



# OmniStream™ Multiview Application Guide

AT-OMNI-111 / WP  
AT-OMNI-121

Atlona Manuals  
**Networked AV**

## Version Information

---

Version	Release Date	Notes
1	May 2023	Initial release

## Sales, Marketing, and Customer Support

---

### Main Office

Atlona Incorporated  
70 Daggett Drive  
San Jose, CA 95134  
United States

Office: +1.408.962.0515

Sales and Customer Service Hours  
Monday - Friday: 6:00 a.m. - 4:30 p.m. (PST)

<https://atlona.com/>

### International Headquarters

Atlona International AG  
Tödistrasse 18  
8002 Zürich  
Switzerland

Office: +41.43.508.4321

Sales and Customer Service Hours  
Monday - Friday: 09:00 - 17:00 (UTC +1)

# Table of Contents

---

<b>Introduction</b>	<b>5</b>
About This Manual	5
<b>Application: Overflow Rooms</b>	<b>6</b>
Overview	6
Planning	7
Configuration	10
Decoder Multiview Configuration	10
Encoder Configuration	13
<b>Programming Recommendations</b>	<b>24</b>
Switching Multiview Layouts	24
Background	24
Changing Multiview Layout	25
Using Velocity	25
Using JSON	30
Atlona Recommendations	34

# Introduction

---

## About This Manual

Multiview functionality in OmniStream 2.0 allows integrators to design systems that display content from multiple streams on a single display without requiring any additional equipment. Proper implementation of multiview systems requires planning during the system design and programming phases to ensure the system operates as expected.

This manual takes readers through the design, configuration, and programming phases of a real-world multiview application. This will introduce the reader to the techniques needed to properly implement a multiview system, which can then be extended to other applications.

The second part of this guide provides programming recommendations for applications where multiview layouts will be switched. Layout changes in OmniStream require several programming steps, and these recommendations provide guidance on the best way to implement multiview layout switches.

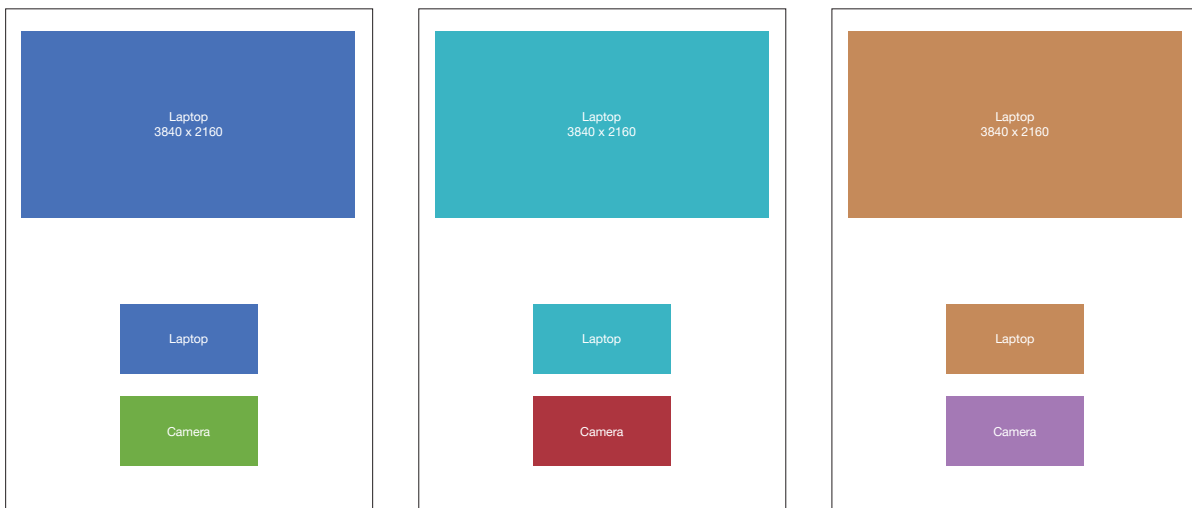
# Application: Overflow Rooms

## Overview

The use of overflow rooms is common in education and other presentation spaces to allow attendees to view a presentation when there is insufficient seating available in the primary space. In order to support overflow, it is necessary that sources in the primary space be available in the overflow space. And, in practical applications where overflow will be supported, this needs to be done flexibly since overflow scheduling (i.e. determining which room is primary and which is overflow) will often be event-dependent.

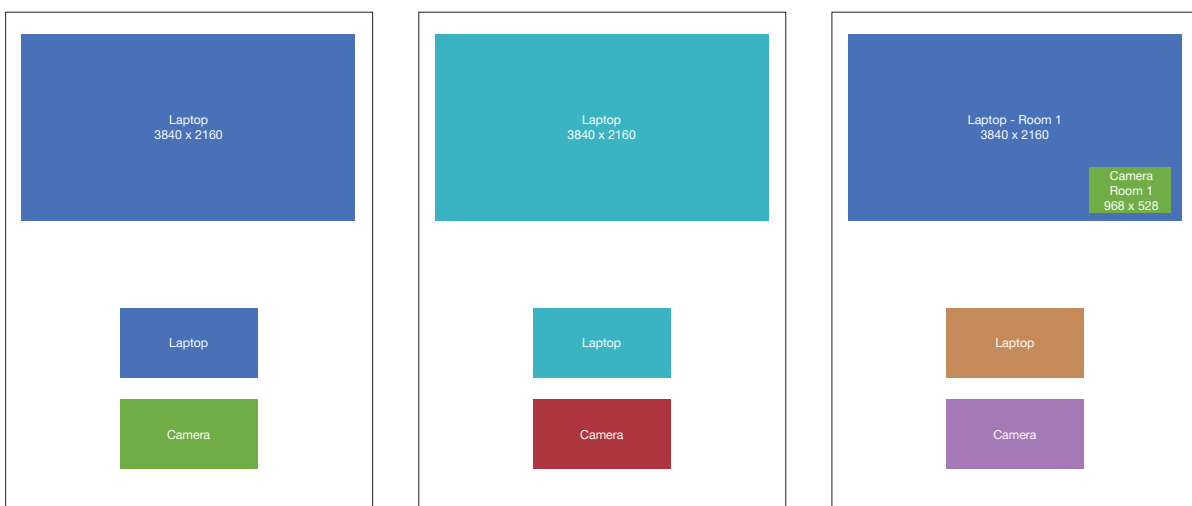
Utilizing OmniStream is perfect for overflow applications since content in one room can be flexibly routed to one or more overflow spaces via a network. To illustrate how this works, *Figure 1.1* shows three rooms where each room is showing only its own content. For purposes of clarity, encoders and decoders are not shown.

*Figure 1.1 - Presentation spaces where each room is showing its own content.*



If a presentation in Room 1 has more attendees than can fit in that room, another room can be used as an overflow space and show both the content and camera from that room. In *Figure 1.2*, Room 3 is used as the overflow space, and shows the content and camera from Room 1 in a PiP format.

*Figure 1.2- Presentation spaces where Rooms 1 and 2 are showing their own content, and Room 3 is used as an overflow for Room 1.*



### Planning

This section will walk you through design and programming considerations for room overflow using OmniStream. Please note that this guide is also applicable for any spaces where users want to show two pieces of content.



#### DEFINITIONS

**Content** – Refers to presentation materials that are the primary content in the space. This includes laptop, document cameras, and media players.

**Camera** – Refers to one of more cameras that show the presenter or attendees and is generally considered secondary to the content.

#### Step 1: Decide what content is to be shown in each space.

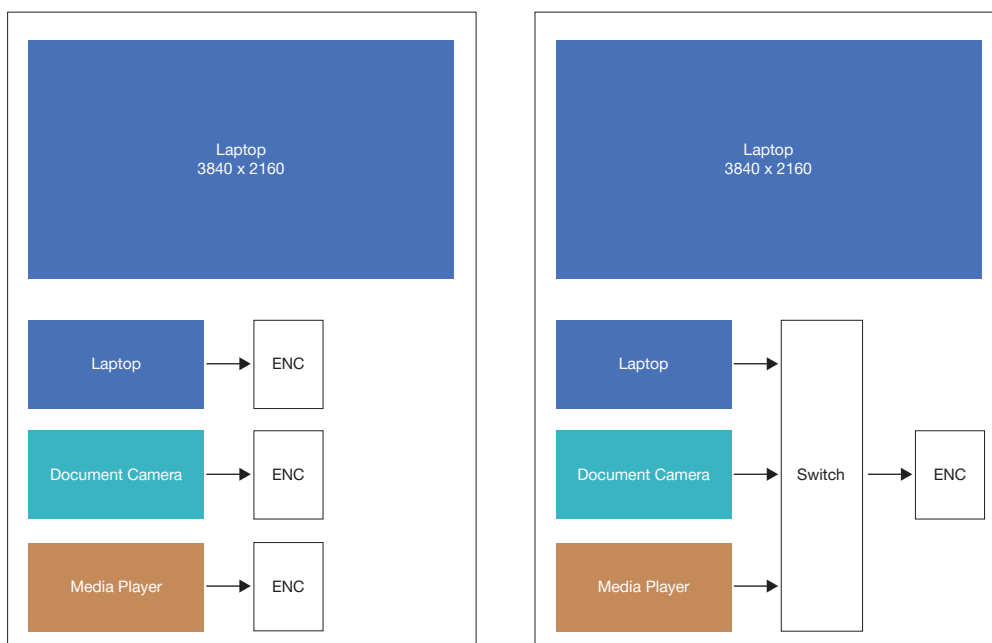
Live presentations requiring overflow support will often have content that is being displayed, which can include a presentation, a document camera, or video content. Additionally, to provide optimal support for overflow, the presentation space should also have a camera. The first step in system design is to consider what content you want to make available in the overflow and primary spaces and use this to determine what encoders are required.

#### Content Sources

Content sources include things like a presenter laptop or document camera. Depending on the design of the room, there are two ways to make these sources part of an OmniStream system:

1. Connect each source to its own OmniStream encoder so that it can be switched to any OmniStream decoder regardless of the state of any other source.
2. Connect each source to a switcher (such as an Atlona Omega switcher), and connect the output of the switcher to an OmniStream encoder. This results in a lower cost room design, but it does mean that overflow spaces will only have access to the content currently switched to in the primary space.

*Figure 1.3 - Two ways to connect content sources to OmniStream. The room on the left has each source connected to its own encoder. The room on the right connects its sources first to a switch, and then the output of that switch connects to an encoder.*



### Cameras

For the best overflow experience there should be at least one camera in the room that is connected to an OmniStream encoder. This allows viewers in the overflow space to view the presenter.

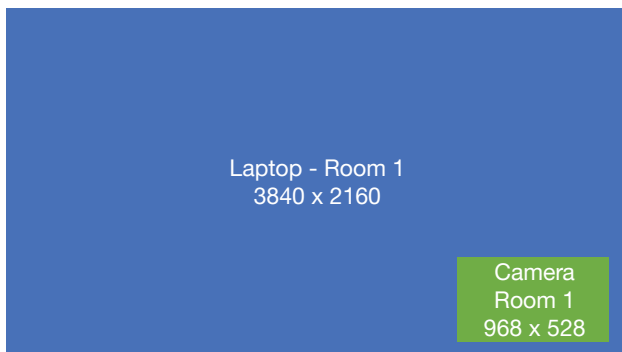
### Step 2: Decide how the content should be distributed in Overflow Spaces

In the overflow rooms, it's always an option to show the content or camera full-screen, but this means that remote viewers will only be able to see one or the other. With the multiview capabilities of OmniStream 2.0, it is possible to composite the content and camera onto a single display so that overflow viewers can see both at the same time.

As part of the system design and implementation, it's important to first consider how overflow rooms will view the content and camera. Some of the most common presentation layouts include:

- **Picture-in-Picture (PiP)** - In PiP applications, the content is shown full screen, and the camera is shown in one of the corners. PiP has the advantage that the content will be shown at maximum size, but comes at the disadvantage that the camera covers part of the content. For this reason, many PiP implementations will allow the PiP content to be moved on the screen.

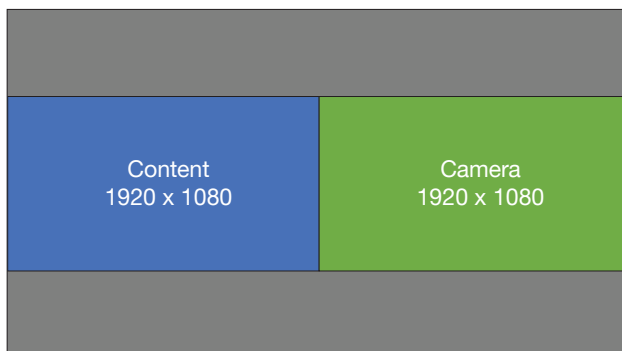
*Figure 1.4 - Example PiP layout. The PiP window can be positioned in any corner to avoid covering critical content.*



- **Side-by-Side (SbS)** - SbS views avoid the problem of the camera covering the content by placing the camera and content side-by-side. In these applications, the content will be scaled down to a resolution less than full screen, and the exact SbS layout will depend on the size expectations for both the content and the camera.

In *Figure 1.5*, both content and camera are shown at the same size. This greatly reduces the size of the content, so this type of layout will typically only be used in overflow applications where the screens are large and where the camera view is determined to be of equal importance to the content.

*Figure 1.5 - Example SbS layout. Note that the content and camera are identical size.*

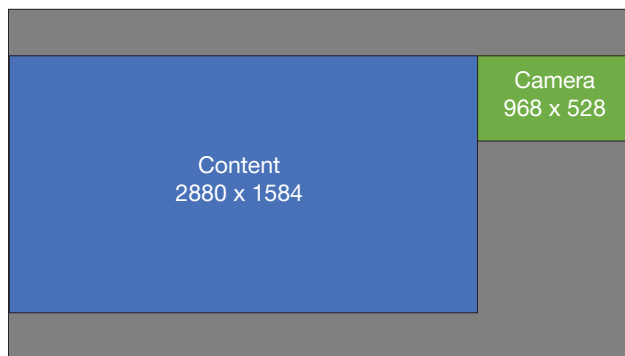




## Application: Overflow Rooms

In those applications where the camera is secondary to the content, users can instead choose a SbS layout where the content occupies most of the display and the camera is scaled down to a small size, much as you would see in the PiP application.

*Figure 1.6 - Example SbS layout. Note that the content is given priority on this display and is shown at a larger size*



### Step 3: Selecting the equipment for a Multiview System

To successfully implement an OmniStream Multiview System, you will need the following:

- 1 x AT-OMNI-111 encoder per content source. This can be designed so that each content source, such as a laptop, document camera, or media player is connected to the encoder. Or, the sources can all be connected to a video switch, and the output of that switch can be connected to the encoder.
- 1 x AT-OMNI-121 decoder per display.
- Multicast-capable network, that connects all spaces that will have overflow capability.
- Control System, such as Atlona Velocity™, that allows the user to make easy content and layout changes.

### Step 4: Configuring the Device

The way the encoders and decoders are configured in a system will depend in large part on the multiview layout or layouts that will be used in overflow rooms. The instructions below are designed to allow users to have a mix of single stream, PiP, and SbS layouts.

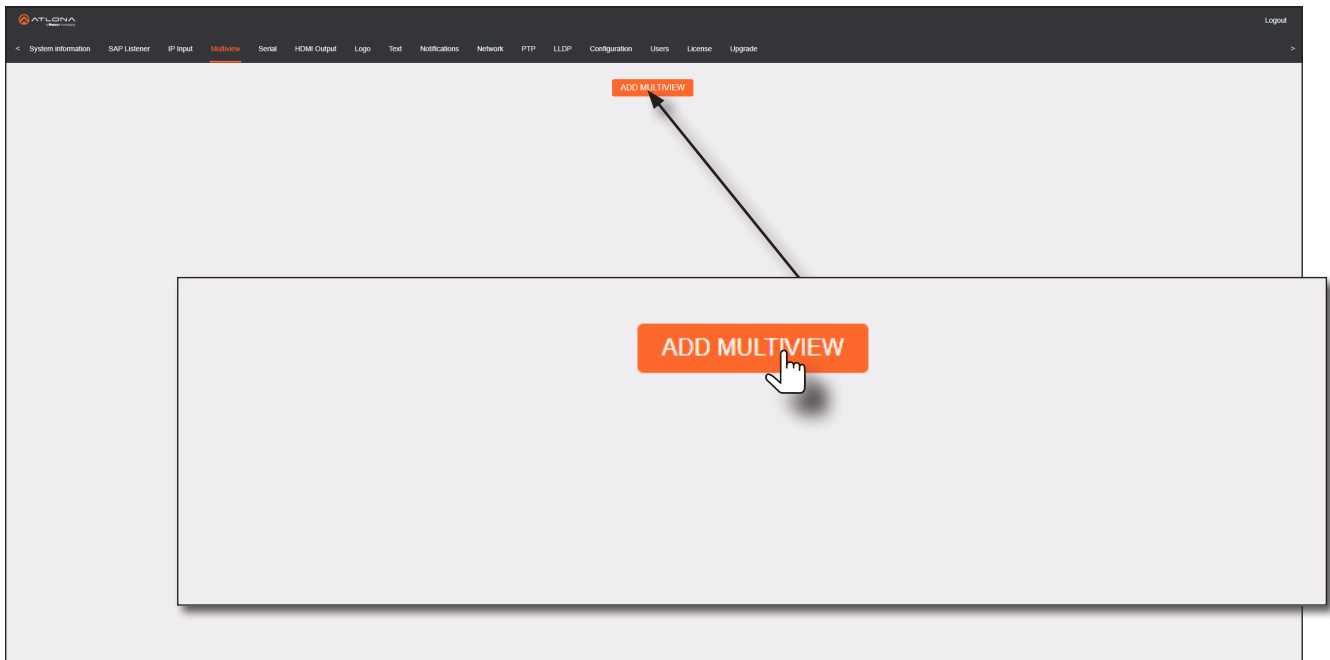
## Configuration

### Decoder Multiview Configuration

Based on the desired multiview layout (as described above), each decoder will need to be configured to display that layout. OmniStream has numerous multiview preset layouts, which can be customized if needed.

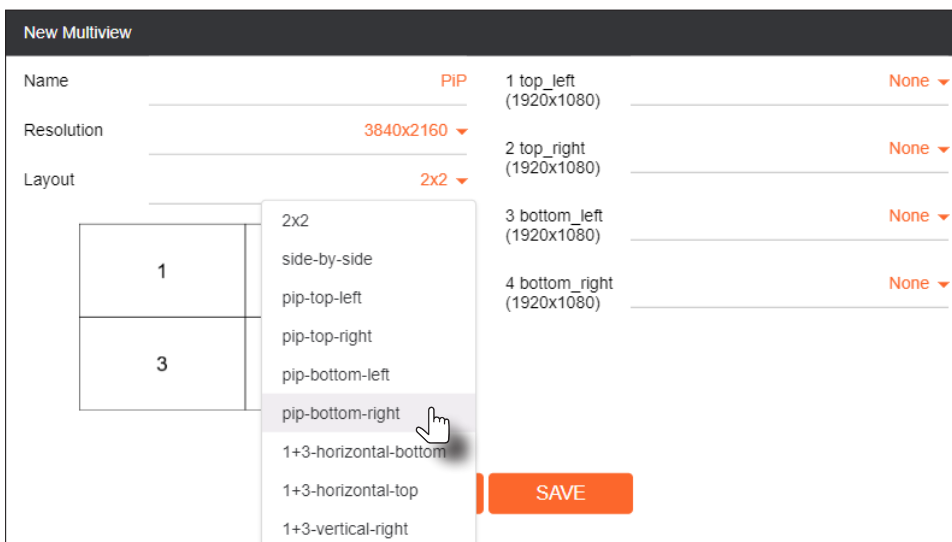
Creating a new layout on a decoder is as simple as going to the **Multiview** tab on the decoder web page and clicking on the **ADD MULTIVIEW** button.

Figure 1.7 - OmniStream decoder web page showing the Multiview tab before any Multiviews have been created.



A pop-up will appear, allowing the user to define a new multiview. Give the multiview a name, select a canvas resolution (for most applications, this will be the default of 3840x2160), and select a layout.

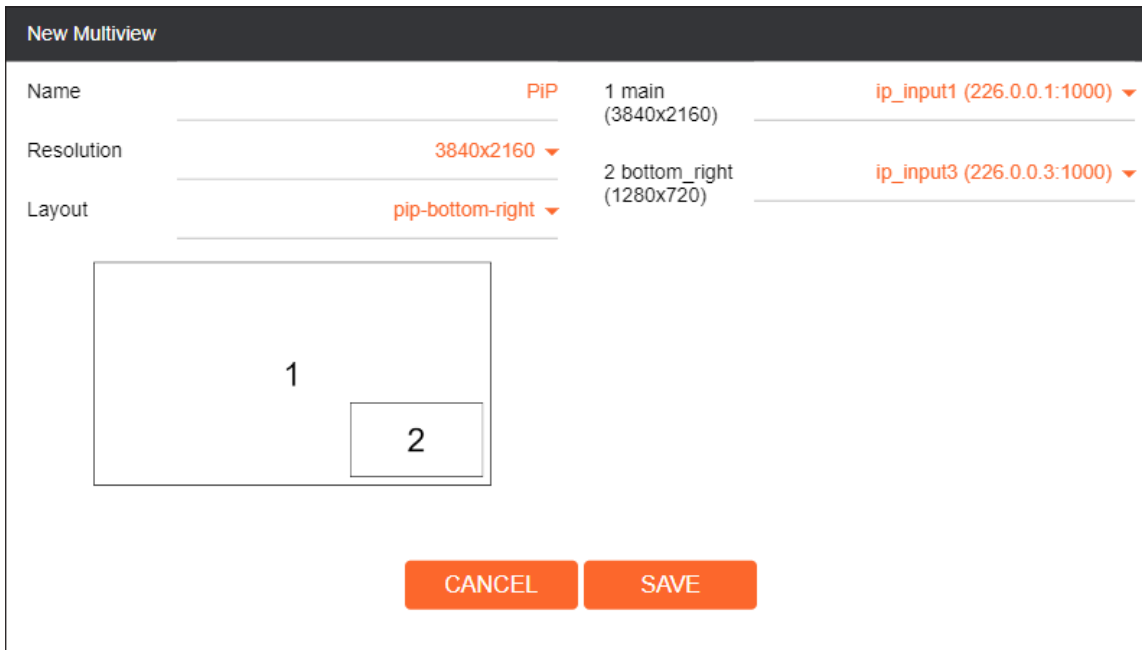
Figure 1.8 - Multiview creation pane showing the various built-in layouts.



## Application: Overflow Rooms

For PiP applications, OmniStream decoders include four different presets, depending on the position of the PiP window. This is just a starting point, since the PiP window can be moved anywhere on the screen using the API. Users should pick the layout they think will most likely be used as a starting point.

Figure 1.9 - Multiview creation pane with one of the PiP layouts selected.



**New Multiview**

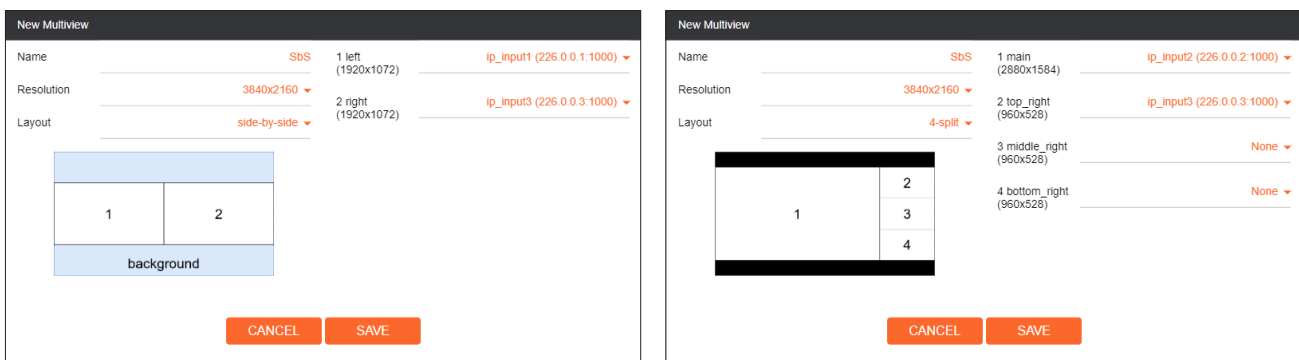
Name	PIP	1 main (3840x2160)	ip_input1 (226.0.0.1:1000) ▼
Resolution	3840x2160 ▼	2 bottom_right (1280x720)	ip_input3 (226.0.0.3:1000) ▼
Layout	pip-bottom-right ▼		

Visual diagram showing a large rectangle labeled '1' and a smaller rectangle labeled '2' positioned in the bottom right corner of '1'.

CANCEL SAVE

For SbS applications, OmniStream decoders make a side-by-side layout available, where each of the two pieces of content is the same size. If customers desire to have the content window much larger, they should select the 4-split layout and delete the two subframes that will not be used.

Figure 1.10 - Multiview creation pane showing two different options for SbS layouts. The layout on the left has both content and camera the same size. The layout on the right allows the content to occupy most of the screen. Note that only 1 of the three panes on the right is needed for the camera.



**New Multiview (Left)**

Name	SbS	1 left (1920x1072)	ip_input1 (226.0.0.1:1000) ▼
Resolution	3840x2160 ▼	2 right (1920x1072)	ip_input3 (226.0.0.3:1000) ▼
Layout	side-by-side ▼		

Visual diagram showing two equal-sized rectangles labeled '1' and '2' side-by-side, with a 'background' area below them.

CANCEL SAVE

**New Multiview (Right)**

Name	SbS	1 main (2880x1584)	ip_input2 (226.0.0.2:1000) ▼
Resolution	3840x2160 ▼	2 top_right (960x528)	ip_input3 (226.0.0.3:1000) ▼
Layout	4-split ▼	3 middle_right (960x528)	None ▼
		4 bottom_right (960x528)	None ▼

Visual diagram showing a large rectangle labeled '1' and three smaller rectangles labeled '2', '3', and '4' stacked vertically to its right.

CANCEL SAVE

When configuring the decoder multiview, you will need to assign ip\_inputs to subframes. For maximum flexibility, this guide assumes ip\_inputs will be used as follows:

- ip\_input1: native resolution source content or scaled content for large SbS.
- ip\_input12: scaled content for SbS.
- ip\_input13: scaled camera for PiP and SbS.

## Application: Overflow Rooms

These ip\_inputs can be assigned when setting up a new multiview according to the following table:

Decoder Layout Type	Default Layout Options	ip_input Assignments
PiP	pip-top-left pip-top-right pip-bottom-left pip-bottom-right	1: ip_input1 2: ip_input13
SbS (content and camera same size)	side-by-side	1: ip_input12 2: ip_input13
SbS (content larger than camera)	4-split	1: ip_input1 2: ip_input13 3: None 4: None

### Encoder Configuration

When configuring encoders for multiview applications, the Encoding and Session configuration will depend on the selected layout. This section will describe those settings; the reader should reference the AT-OMNI-111 documentation for information about other encoder configuration options.

#### Content Encoder

Encoders that are connected to content sources will be assumed to be the primary content shown in the multiview. If this content will ever be secondary, you can configure this encoder in the same manner as a camera encoder.

In the Encoding tab of the AT-OMNI-111, there will be two encoders that can be configured, including scalers for each encoder. The recommended settings are shown in the table below:

Setting	PiP	SbS Using side-by-side template in decoder	SbS* Using 4-split template in decoder
Encoder 1	Input: hdmi_input1 Max bit rate: 750 Mb/s Scaler: 3840x2160	Input: hdmi_input1 Max bit rate: 700 Mb/s Scaler: 3840x2160	Input: hdmi_input1 Max bit rate: 700 Mb/s Scaler: 2880x1584
Encoder 2	Input: Not used Max bit rate: 150 Mb/s Scaler: 1920x1080	Input: hdmi_input1 Max bit rate: 200 Mb/s Scaler: 1920x1080	Input: Not used Max bit rate: 200 Mb/s Scaler: 1920x1080

\*Because the content resolution in the 4-split layout is greater than supported by the Encoder 2 scaler, Encoder 1 will be used. This means that 4K sources will be scaled down in the encoder, which could result in some loss of fidelity in rooms that are not using multiview. In cases where the content is not being used in overflow scenarios, the scaler can be reprogrammed to 3840x2160, but it will need to be changed to 2880x1584 when an overflow room is used.

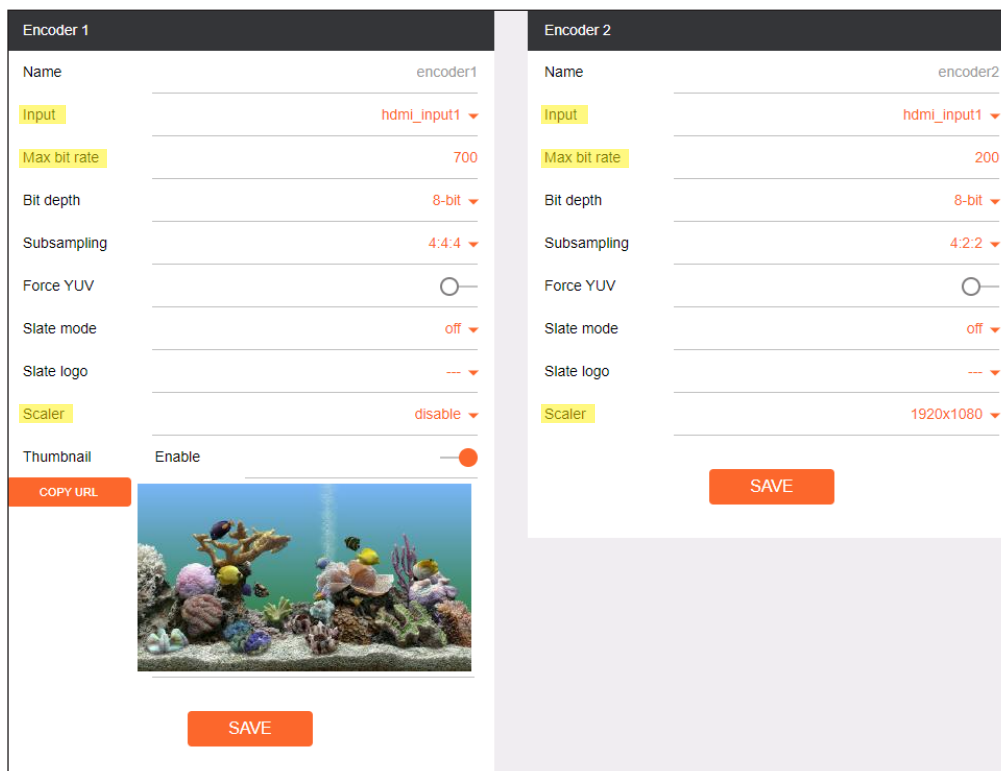


Figure 1.11 - Encoding tab on the AT-OMNI-111 with the fields highlighted that need to be changed based on the target overflow application.

## Application: Overflow Rooms

The Session tab contains the stream settings for each encoder. PiP only requires the use of Session 1 on Content Encoders, whereas SbS requires the use of both Sessions. The configuration of the two sessions will be virtually identical, but the reader is encouraged to adhere to the following guidelines:

- The scrambling key should be enabled (to support HDCP content) and set to the same streaming key as all other encoders that will be used in the overflow application.
- Session 1 Video should use encoder1 and Session 2 Video should use encoder2.
- The Destination IP address and UDP port for each Session should be assigned based on a scheme defined by the end user. One recommendation is to pick a range of multicast addresses for the encoders, and assign odd multicast addresses to Session 1 and even multicast addresses to Session 2. *It is critical that no two encoders in a system have the same multicast addresses.*
- Only Session 1 needs to have Audio enabled and a unique multicast address assigned to it, since that one stream can be used in overflow rooms regardless of overflow configuration.

Figure 1.12 - Session tab on the AT-OMNI-111 with the fields highlighted that need to be changed based on the target overflow application.

Session 1		Session 2	
Name	session1	Name	session2
Interface	eth1	Interface	eth1
EncGroup	Enable <input type="radio"/>	EncGroup	Enable <input type="radio"/>
SAP Listener	Enable <input checked="" type="radio"/>	SAP Listener	Enable <input checked="" type="radio"/>
Interval	10	Interval	10
Name	Roku	Name	RokuHD
Description	N/A	Description	N/A
Originator	-	Originator	-
Categorisation	Atlona	Categorisation	Atlona
Scrambling	Enable <input checked="" type="radio"/>	Scrambling	Enable <input checked="" type="radio"/>
Key	scrambling	Key	scrambling
Video	Encoder: encoder1	Video	Encoder: encoder2
Enable	Enable <input checked="" type="radio"/>	Enable	Enable <input type="radio"/>
Destination IP address	225.0.0.5	Destination IP address	225.0.0.7
Destination UDP port	1000	Destination UDP port	1020
TTL	255	TTL	255
DSCP	Best effort	DSCP	Best effort
FEC enable	<input type="radio"/>	FEC enable	<input type="radio"/>
FEC rows	15	FEC rows	15
FEC columns	15	FEC columns	15
	<a href="#">Link test</a>		<a href="#">Link test</a>
Audio	Source: Not used	Audio	Source: Not used
AUX	Source: Commands	AUX	Source: Commands
Enable	Enable <input type="radio"/>	Enable	Enable <input type="radio"/>
	<a href="#">SAVE</a>		<a href="#">SAVE</a>

### Camera Encoder

Encoders that are connected to camera sources will be assumed to be the secondary content shown in the multiview. If this content will ever be primary, you can configure this encoder in the same manner as a content encoder.

In the Encoding tab of the AT-OMNI-111, there will be two encoders that can be configured, including scalers for each encoder. The recommended settings are shown in the table below:

Setting	PiP	SbS Using side-by-side template in decoder	SbS Using 4-split template in decoder
Encoder 1	Input: hdmi_input1 Max bit rate: 750 Mb/s Scaler: disable	Input: hdmi_input1 Max bit rate: 650 Mb/s Scaler: disable	Input: hdmi_input1 Max bit rate: 750 Mb/s Scaler: disable
Encoder 2	Input: hdmi_input1 Max bit rate: 150 Mb/s Scaler: 960x528*	Input: hdmi_input1 Max bit rate: 250 Mb/s Scaler: 1920x1080	Input: hdmi_input1 Max bit rate: 150 Mb/s Scaler: 960x528

*\*This will determine the size of the PiP window and can be made larger or smaller depending on the target PiP size.*

The Session tab contains the stream settings for each encoder. The configuration of the two sessions will be virtually identical, but the reader is encouraged to adhere to the following guidelines:

- The scrambling key should be enabled (to support HDCP content) and set to the same streaming key as all other encoders that will be used in the overflow application.
- Session 1 Video should use encoder1 and Session 2 Video should use encoder2.
- The Destination IP address and UDP port for each Session should be assigned based on a scheme defined by the end user. One recommendation is to pick a range of multicast addresses for the encoders, and assign odd multicast addresses to Session 1 and even multicast addresses to Session 2. *It is critical that no two encoders in a system have the same multicast addresses.*
- Only Session 1 needs to have Audio enabled and a unique multicast address assigned to it, since that one stream can be used in overflow rooms regardless of overflow configuration.

### Decoder Stream Configuration

Now that the encoders and decoders have been configured, the one remaining step will be to assign the correct multicast address and port numbers to the ip\_inputs used within the multiview. Depending on the layout selected, the multicast address and port number for the desired stream should be entered into the ip\_inputs assigned to the multiview.

Figure 1.13 - Decoder tab on the AT-OMNI-121 where multicast addresses will be assigned to ip\_inputs.

Input 1		Input 2	
Name	ip_input1	Name	ip_input2
Enable	<input checked="" type="checkbox"/>	Enable	<input checked="" type="checkbox"/>
Interface	eth1	Interface	eth1
Multicast address	225.0.0.1	Multicast address	225.0.0.5
Port	1000	Port	1000
Multicast filter (IGMPv3)	Mode	Multicast filter (IGMPv3)	Mode
	Addresses*		Addresses*
	exclude ▼		exclude ▼
	N/A		N/A
*Separate multiple IP addresses with a comma.		*Separate multiple IP addresses with a comma.	
<input type="button" value="SAVE"/>		<input type="button" value="SAVE"/>	



### Step 5: Programming the System

In any of the rooms, you will want to give the user the ability to select what content is shown in that room. This could be full-screen content from within the room, full-screen content from another room, or a multiview layout where the content within the layout can also be changed.

Most of the time, the user will select the content that is shown using a touch panel. This section provides examples of the programming required to change the content on the decoder. Examples are provided both using Atлона Velocity macros and JSON.

### Showing Content Full-Screen

In the case where content will be shown full screen, regardless of whether the content comes from within the room or from another room, the programming will command the decoder to show ip\_input1 and not a multiview.

Steps in programming:

1. Set the desired multicast address and port in ip\_input1 on the decoder.
2. Switch the HDMI output on the decoder to ip\_input1.

### Using Velocity

**Left Display to BluRay** 🏠 ▶ + ▾

Hide In:  Drawer  Preset Page

**Command 1** Custom Value:

Device:  ▾

Command:  ▾

---

IP Input:  ▾ <>

Multicast IP Address:  ▾ <>

Multicast Port:  ▾ <>

Enabled:  ▾ <>

Repeat:  ▾ Interval:  ms Delay (After):  ms ⋮

---

**Command 2** Custom Value:

Device:  ▾

Command:  ▾

---

IP Input Start:  ▾ <>

Or Multi View Name:  ▾ <>

Repeat:  ▾ Interval:  ms Delay (After):  ms ⋮

### Using JSON

The following two commands need to be sent to the decoder where the items in **orange** need to be changed for the specific stream that will be viewed.

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [{
      "multicast": {
        "address": "225.0.0.1"
      },
      "name": "ip_input1",
      "port": 1000
    }]
  }
}
```

Changing the HDMI output to ip\_input1:

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "hdmi_output",
    "config": [{
      "name": "hdmi_output1",
      "video": {
        "input": "ip_input1"
      },
      "number": 1
    }]
  }
}
```

### Showing Multiview in Overflow Rooms

Showing multiview in overflow rooms is a matter of assigning which content and camera will be shown by setting the correct ip\_inputs on the decoder and then switching to that multiview. As described in a previous section, if the content will be shown in a PiP layout or in a SbS layout with large content, the content will be assigned to ip\_input1, and if the content will be shown in a SbS layout, the content will be assigned to ip\_input12.

Steps in programming:

1. Assign the content multicast address and port number to ip\_input1 (PiP or large SbS) or ip\_input12 (SbS).
2. Assign the camera multicast address and port number to ip\_input13.
3. Switch the decoder to the desired multiview layout.

### Using Velocity

**Left Display to BluRay PiP** ⌵ ▶ + ▾

Hide In:  Drawer  Preset Page

---

**Command 1** Custom Value:

Device: Device: Left Display Decoder (10.1.0.19) ▾

Command: IP Input Set All ▾

---

IP Input: 1 ▾ <>

Multicast IP Address: 225.0.0.1 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: on ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

**Command 2** Custom Value:

Device: Device: Left Display Decoder (10.1.0.19) ▾

Command: IP Input Set All ▾

---

IP Input: 13 ▾ <>

Multicast IP Address: 225.0.0.11 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: on ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

**Command 3** Custom Value:

Device: Device: Left Display Decoder (10.1.0.19) ▾

Command: HDMI Output - Video Input ▾

---

IP Input Start: None ▾ <>

Or Multi View Name: multiviewPiP ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

### Using JSON

The following three commands need to be sent to the decoder where the items in **orange** need to be changed for the specific streams and multiview that will be used.

Changing the ip\_input1 multicast address and port:

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [{
      "multicast": {
        "address": "225.0.0.1"
      },
      "name": "ip_input1",
      "port": 1000
    }]
  }
}
```

Changing the ip\_input13 multicast address and port:

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [{
      "multicast": {
        "address": "225.0.0.11"
      },
      "name": "ip_input1",
      "port": 1000
    }]
  }
}
```

Changing the HDMI output to the PiP multiview:

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "hdmi_output",
    "config": [{
      "name": "hdmi_output1",
      "video": {
        "input": "multiviewPiP"
      },
      "number": 1
    }]
  }
}
```

### Changing the Source within a Multiview

Showing multiview in overflow rooms is a matter of assigning which content and camera will be shown the correct. While showing a multiview, the user might want to change the content or camera shown. In this case, the user only needs to change the multicast address and port number of the ip\_input that will be changed.

Steps in the programming:

1. Assign the multicast address and port number to the corresponding ip\_input.

### Using Velocity

**Left Display PiP Content to Roku** ⌵ ▶ + ▾

Hide In:  Drawer  Preset Page

Command 1  Custom Value:

Device: Device: Left Display Decoder (10.1.0.19) ▾

Command: IP Input Set All ▾

---

IP Input: 1 ▾ <>

Multicast IP Address: 225.0.0.5 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: on ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay (After): 0 ms ⋮

### Using JSON

The following command needs to be sent to the decoder where the items in **orange** need to be changed for the specific stream that will be used:

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [{
      "multicast": {
        "address": "225.0.0.5"
      },
      "name": "ip_input1",
      "port": 1000
    }]
  }
}
```

### Moving the PiP Window

In PiP applications, the user may want to move the location of the PiP window. This can be done by changing the location of the PiP subframe. Recommended settings for the different PiP window locations are shown below.

PIP Window Location	Anchor Point	x	y
Upper Left	top left	32	32
Upper Right	top right	3808	32
Upper Left	bottom left	32	2128
Lower Right	bottom right	3808	2128

Steps in the programming:

1. Change the anchor point and the x and y coordinates of the PiP subframe in the multiview.

### Using Velocity

**Left Display PiP Upper Left** ⌵ ▶ + ▾

Hide In:  Drawer  Preset Page

**Command 1** Custom Value:

Device:  ▾

Command:  ▾

---

Multiview Name:  ▾ <>

Subframe Name:  ▾ <>

Anchor:  ▾ <>

x:  ▾ <>

y:  ▾ <>

Z-Order:  ▾ <>

Repeat:  ▾ Interval:  ms Delay (After):  ms ⋮

### Using JSON

The following command needs to be sent to the decoder where the items in **orange** need to be changed based on the desired PiP window location and the item in **blue** is the name of the subframe containing the PiP content:

```
{
  "id": "multiview-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "multiview",
    "config": [{
      "name": "multiviewPiP",
      "subframes": [{
        "anchor": "top left",
        "name": "bottom_right",
        "priority": 2,
        "x": 32,
        "y": 32
      }]
    }]
  }
}
```

# Programming Recommendations

---

## Switching Multiview Layouts

The Multiview functionality of OmniStream 2.0 allows dealers to composite multiple streams inside a decoder to show, for example, PiP, side-by-side, and quad-view content. Because this content utilizes more than one stream, and because users may want to dynamically change multiviews, there are some programming steps that can be used to simplify multiview changes.

### Background

The OmniStream decoders support a maximum of 16 defined ip\_input slots. These slots are used to pre-define multicast addresses and ports and switch between them. In a reasonably-sized OmniStream system, there may be far more than 16 streams that can be selected from to display in a multiview. If audio will be used with the multiview, then the audio stream should be assigned to ip\_input5.

Multiviews can show a maximum of four streams, so the recommended best practice is to utilize the same four ip\_inputs for all multiviews (for example ip\_input1 through ip\_input4).

The rest of this document shows two ways to implement the programming for this recommendation: one way uses Velocity™, the other way uses JSON over a websocket connection.

At the end of the document the reader can find some additional recommendations to provide the best overall user experience when switching multiviews.



### Changing Multiview Layout

The following is a recommendation for the best way to programmatically change multiview layout. If the user only needs to change a stream or streams within the layout already shown, the program only needs to change the addresses assigned to the ip\_inputs used within the layout. If the user needs to change the entire layout, Atlona recommends the following:

1. Disable the four ip\_inputs used for multiview.
2. *Optional:* change the encoder resolution and bit rate as needed to accommodate the target multiview.
3. Change the multicast addresses for the four ip\_inputs to the addresses that should be shown after the switch.



**NOTE:** Each multiview can have 1 to 4 subframes, so the program will only need to set the addresses that will be shown.

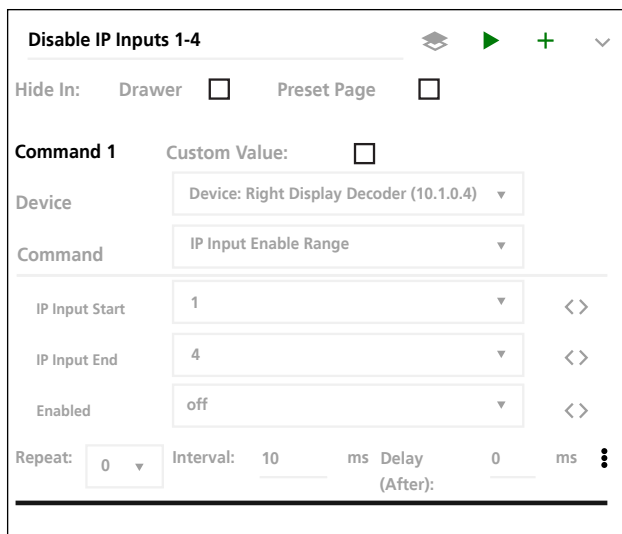
4. Tell the decoder to switch multiview layout.
5. Re-enable the four ip\_inputs used for multiview.

### Using Velocity

To aid in programming, the following are provided as examples in Velocity that should be used for each of the steps above. Please note that these examples require Velocity 2.5.3 or greater.

1. Disable the four ip\_inputs used for multiview. The following example utilizes ip\_inputs 1-4, which should work for most applications. However, if an installation requires use of other ip\_inputs, the **IP Input Start** and **IP Input End** fields should be changed to match the ip\_inputs that will be used with a multiview.

As long as the ip\_inputs being disabled are contiguous (e.g. 1-4), you can use the `IP Input Enable Range` command to easily disable all inputs



The screenshot shows a configuration window titled "Disable IP Inputs 1-4". At the top, there are icons for a folder, a play button, a plus sign, and a minus sign. Below the title, there are two checkboxes: "Hide In: Drawer" and "Preset Page", both of which are currently unchecked. The main configuration area is labeled "Command 1" and includes a "Custom Value" checkbox, which is also unchecked. The configuration fields are as follows:

- Device:** A dropdown menu showing "Device: Right Display Decoder (10.1.0.4)".
- Command:** A dropdown menu showing "IP Input Enable Range".
- IP Input Start:** A dropdown menu showing "1" with left and right arrow icons.
- IP Input End:** A dropdown menu showing "4" with left and right arrow icons.
- Enabled:** A dropdown menu showing "off" with left and right arrow icons.
- Repeat:** A dropdown menu showing "0".
- Interval:** A text input field showing "10" followed by "ms".
- Delay (After):** A text input field showing "0" followed by "ms".

## Programming Recommendations

If the ip\_inputs are not contiguous, you will need to disable each ip\_input one-by-one using the IP Input Enable command as shown below

**Disable IP Inputs** 🏠 ▶ + ▾

Hide In:  Drawer  Preset Page

---

**Command 1** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▾

Command: IP Input Enable ▾

IP Input Start: 1 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

**Command 2** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▾

Command: IP Input Enable ▾

IP Input Start: 2 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

**Command 3** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▾

Command: IP Input Enable ▾

IP Input Start: 3 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

**Command 4** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▾

Command: IP Input Enable ▾

IP Input Start: 4 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After):

---

## Programming Recommendations

2. *Optional:* change the encoder resolution and bit rate as needed to accommodate the target multiview. This change should be made with care as any other decoder accessing this stream will be affected by the bit rate and resolution change.

**Change Encoder Resolution and Bit Rate** 🏠 ▶ + ▼

Hide In:  Drawer  Preset Page

---

**Command 1** Custom Value:

Device:

Command:

---

Resolution:  <>

Encoder:  <>

Repeat:  Interval:  ms Delay  ms (After):

---

**Command 2** Custom Value:

Device:

Command:

---

Mb/s:  <>

Encoder:  <>

Repeat:  Interval:  ms Delay  ms (After):

3. Change the multicast addresses for the four ip\_inputs to the addresses that should be shown after the switch.



**NOTE:** Each multiview can have 1 to 4 subframes, so the program will only need to set the addresses that will be shown.

The **Multicast IP Address** and **Multicast Port** fields should be changed to match the multicast addresses and port numbers that will be shown in the multiview. The **IP Input** field only needs to be changed if the installation is using something other than ip\_inputs 1-4. See the next page for an illustration.

## Programming Recommendations

**Quad View Layout 1** ⌵ ▶ + ▾

Hide In:  Drawer  Preset Page

---

**Command 1** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▾

Command: IP Input Set All ▾

IP Input: 1 ▾ <>

Multicast IP Address: 226.0.0.1 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After) ⋮

---

**Command 2** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▾

Command: IP Input Set All ▾

IP Input: 2 ▾ <>

Multicast IP Address: 226.0.0.2 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After) ⋮

**Command 3** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▾

Command: IP Input Set All ▾

IP Input: 3 ▾ <>

Multicast IP Address: 226.0.0.3 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After) ⋮

---

**Command 4** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▾

Command: IP Input Set All ▾

IP Input: 4 ▾ <>

Multicast IP Address: 226.0.0.4 ▾ <>

Multicast Port: 1000 ▾ <>

Enabled: off ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After) ⋮

- Tell the decoder to switch multiview layout. Note that the **Multi View Name** field will need to be changed to match the name of the multiview that is being switched to.

**Switch Multiview Layout** ⌵ ▶ + ▾

Hide In:  Drawer  Preset Page

---

**Command 1** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▾

Command: HDMI Output - Video Input ▾

IP Input: None ▾ <>

Or Multi View Name: multiviewQuad ▾ <>

Repeat: 0 ▾ Interval: 10 ms Delay 0 ms (After) ⋮

## Programming Recommendations

- Re-enable the four ip\_inputs used for multiview. The following example utilizes ip\_inputs 1-4, which should work for most applications. However, if an installation requires use of other ip\_inputs, the **IP Input Start** and **IP Input End** fields should be changed to match the ip\_inputs that will be used with a multiview.

As long as the ip\_inputs being disabled are contiguous (e.g. 1-4), you can use the IP Input Enable Range command to easily disable all inputs.

**Enable IP Inputs 1-4** 🏠 ▶ + ▼

Hide In:  Drawer  Preset Page

**Command 1** Custom Value:

Device: Device: Right Display Decoder (10.1.0.4) ▼

Command: IP Input Enable Range ▼

IP Input Start: 1 ▼ <>

IP Input End: 4 ▼ <>

Enabled: on ▼ <>

Repeat: 0 ▼ Interval: 10 ms Delay 0 ms (After):

If the ip\_inputs are not contiguous, you will need to enable each ip\_input one-by-one using the IP Input Enable command as shown below.

**Disable IP Inputs** 🏠 ▶ + ▼

Hide In:  Drawer  Preset Page

**Command 1** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▼

Command: IP Input Enable ▼

IP Input Start: 1 ▼ <>

Enabled: off ▼ <>

Repeat: 0 ▼ Interval: 10 ms Delay 0 ms (After):

---

**Command 2** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▼

Command: IP Input Enable ▼

IP Input Start: 2 ▼ <>

Enabled: off ▼ <>

Repeat: 0 ▼ Interval: 10 ms Delay 0 ms (After):

**Command 3** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▼

Command: IP Input Enable ▼

IP Input Start: 3 ▼ <>

Enabled: off ▼ <>

Repeat: 0 ▼ Interval: 10 ms Delay 0 ms (After):

---

**Command 4** Custom Value:

Device: Device Right Display Decoder (10.1.0.4) ▼

Command: IP Input Enable ▼

IP Input Start: 4 ▼ <>

Enabled: off ▼ <>

Repeat: 0 ▼ Interval: 10 ms Delay 0 ms (After):

## Programming Recommendations

### Using JSON

The following is a recommendation for the best way to programmatically change multiview layout using JSON over WebSockets. If the user only needs to change a stream or streams within the layout already shown, the program only needs to change the addresses assigned to the ip\_inputs used within the layout. If the user needs to change the entire layout, Atlona recommends the following.

1. Disable the four ip\_inputs used for multiview.
2. Optional: change the encoder resolution and bit rate as needed to accommodate the target multiview.
3. Change the multicast addresses for the four ip\_inputs to the addresses that should be shown after the switch.



**NOTE:** Each multiview can have 1 to 4 subframes, so the program will only need to set the addresses that will be shown.

4. Tell the decoder to switch multiview layout.
5. Re-enable the four ip\_inputs used for multiview.

To aid in programming, the following are provided as examples of the JSON that should be used for each of the steps above.

1. Disable the four ip\_inputs used for multiview. The following example utilizes ip\_inputs 1-4, which should work for most applications. However, if an installation requires use of other ip\_inputs, the text in **orange** should be changed to match the ip\_inputs that will be used with a multiview.

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [{
      "enabled": false,
      "name": "ip_input1"
    },
    {
      "enabled": false,
      "name": "ip_input2"
    },
    {
      "enabled": false,
      "name": "ip_input3"
    },
    {
      "enabled": false,
      "name": "ip_input4"
    }
  ]
}
```

## Programming Recommendations

- Optional: change the encoder resolution and bit rate as needed to accommodate the target multiview. This change should be made with care as any other decoder accessing this stream will be affected by the bit rate and resolution change. The number in **orange** should be changed to the desired bit rate of the stream (in Mb/s) and the numbers in **blue** should be changed to the desired resolution of the encoded video (in pixels).

```
{
  "id": "vc2-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "vc2",
    "config": [
      {
        "name": "vc2_encoder2",
        "bitrate": 150,
        "scaler": {
          "height": 720,
          "width": 1280
        }
      }
    ]
  }
}
```

- Change the multicast addresses for the four ip\_inputs to the addresses that should be shown after the switch. Note: a multiview can have 1-4 subframes, so the program will only need to set the addresses that will be shown.

The text in **orange** should be changed to match the multicast addresses and port numbers that will be shown in the multiview. The text in **blue** only needs to be changed if the installation is using something other than ip\_inputs 1-4.

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [
      {
        "multicast": {
          "address": "226.0.0.1"
        },
        "name": "ip_input1",
        "port": 1000
      },
      {
        "multicast": {
          "address": "226.0.0.2"
        },
        "name": "ip_input2",
        "port": 1000
      }
    ]
  }
}
```

(continued on next page)

```

    {
      "multicast": {
        "address": "226.0.0.3"
      },
      "name": "ip_input3",
      "port": 1000
    },
    {
      "multicast": {
        "address": "226.0.0.4"
      },
      "name": "ip_input4",
      "port": 1000
    }
  ]
}

```

4. Tell the decoder to switch multiview layout. Note that the text in **orange** will need to be changed to match the name of the multiview that is being switched to.

```

{
  "id": "hdmi_output-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "hdmi_output",
    "config": [
      {
        "name": "hdmi_output1",
        "video": {
          "input": "multiviewQuad"
        }
      }
    ]
  }
}

```



## Programming Recommendations

5. Re-enable the four ip\_inputs used for multiview. Note that the following example utilizes ip\_inputs 1-4, which should work for most applications. However, if an installation requires use of other ip\_inputs, the text in **orange** should be changed to match the ip\_inputs that will be used with a multiview.

```
{
  "id": "ip_input-set",
  "username": "admin",
  "password": "Atlona",
  "config_set": {
    "name": "ip_input",
    "config": [
      {
        "enabled": true,
        "name": "ip_input1"
      },
      {
        "enabled": true,
        "name": "ip_input2"
      },
      {
        "enabled": true,
        "name": "ip_input3"
      },
      {
        "enabled": true,
        "name": "ip_input4"
      }
    ]
  }
}
```

## Programming Recommendations

---

### Atlona Recommendations

Based on experience programming multiviews, Atlona recommends the following to improve the experience of the viewer:

- If the control system allows users to move the location of subframes on a display, it may be advised to reset the subframe location the next time the multiview is selected to provide a consistent experience across uses.
- If multiple displays in a room will change multiview at the same time, it is advised to disable the ip\_inputs across all decoders so that the screens blank together before making the switch. This will also avoid any issues from appearing on a display if encoder settings are also changed.

