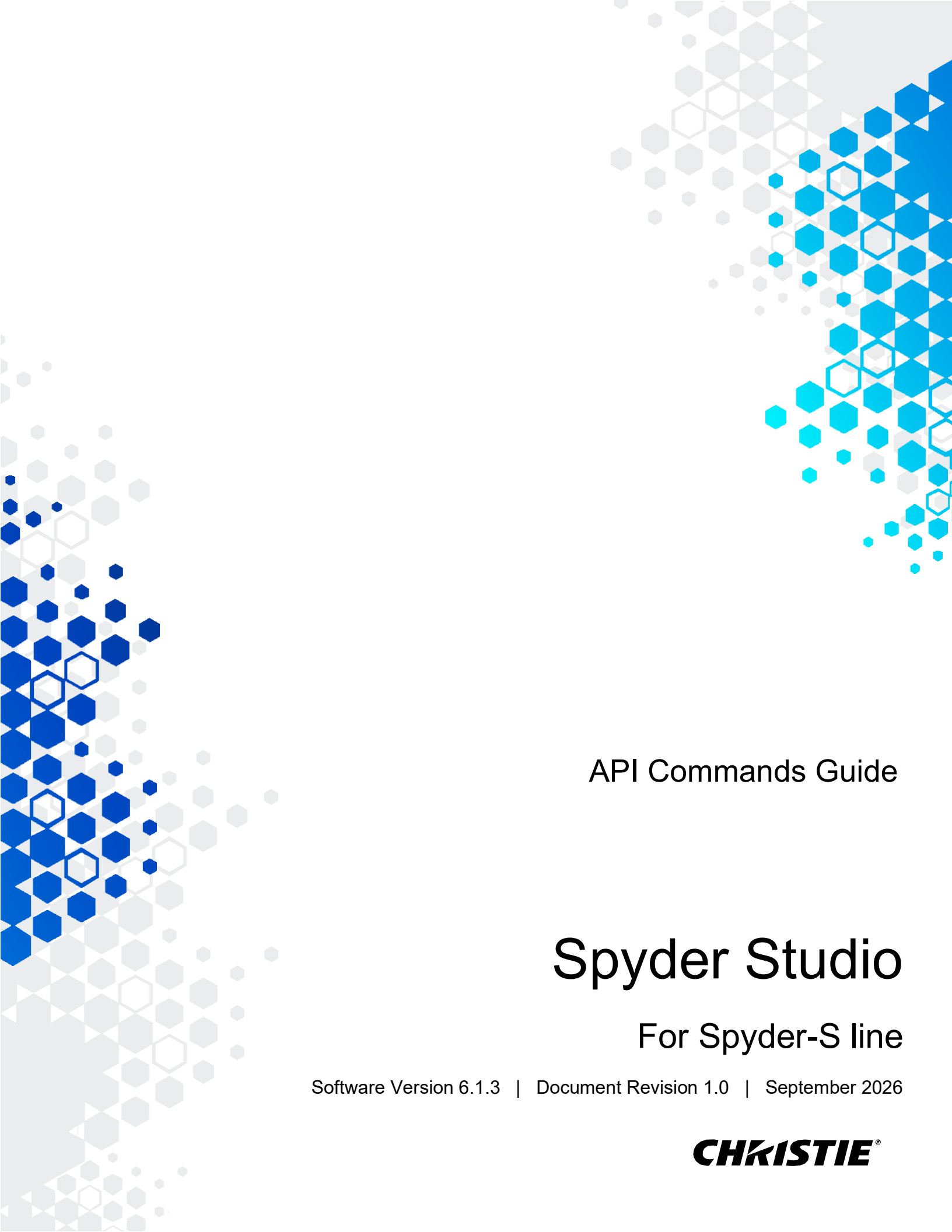




API Commands Guide

Spyder Studio

For Spyder-S line

Software Version 6.1.3 | Document Revision 1.0 | September 2026

CHRISTIE®

NOTICES

COPYRIGHT AND TRADEMARKS

Copyright © 2026 Christie Digital Systems USA, Inc. All rights reserved.

All brand names and product names are trademarks, registered trademarks or trade names of their respective holders.

GENERAL

Every effort has been made to ensure accuracy, however in some cases changes in the products or availability could occur which may not be reflected in this document. Christie reserves the right to make changes to specifications at any time without notice. Performance specifications are typical, but may vary depending on conditions beyond Christie's control such as maintenance of the product in proper working conditions. Performance specifications are based on information available at the time of printing. Christie makes no warranty of any kind with regard to this material, including, but not limited to, implied warranties of fitness for a particular purpose. Christie will not be liable for errors contained herein or for incidental or consequential damages in connection with the performance or use of this material. Our centers of excellence for manufacturing in Kitchener, Ontario, Canada and in Shenzhen, China are ISO 9001:2015 Quality Management System-certified.

Christie is committed to making our documents free from language bias; however, we are not responsible for the language used on any linked or third-party documentation.

For the most current technical documentation and office contact information, visit www.christiedigital.com.

WARRANTY

Products are warranted under Christie's standard limited warranty, the details of which are available at <https://www.christiedigital.com/help-center/warranties/> or by contacting your Christie dealer or Christie.

PREVENTATIVE MAINTENANCE

Preventative maintenance is an important part of the continued and proper operation of your product. Failure to perform maintenance as required, and in accordance with the maintenance schedule specified by Christie, voids the warranty. For preventative maintenance schedules, refer to www.christiedigital.com.

REGULATORY (if applicable)

The product has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the product is operated in a commercial environment. The product generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of the product in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at the user's own expense. Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

CAN ICES-3 (A) / NMB-3 (A)

이 기기는 업무용(A급)으로 전자파적합등록을 한 기기이오니 판매자 또는 사용자는 이점을 주의하시기 바라며, 가정 외의 지역에서 사용하는 것을 목적으로 합니다.

ENVIRONMENTAL



The product is designed and manufactured with high-quality materials and components some of which can be recycled and reused. This symbol  means electrical and electronic equipment, at their end-of-life, should be disposed of separately from regular waste. Please dispose of the product appropriately and according to local regulations. In the European Union, separate collection systems are for used electrical and electronic products.

If printing this document, consider printing only the pages you need and select the double-sided option.

Please help us to conserve the environment we live in!

NOTATION

Learn the hazard and information symbols used in the product documentation.



Danger! If not avoided, the following could result in death or serious injury.



Warning! If not avoided, the following could result in death or serious injury.



Caution! If not avoided, the following could result in minor or moderate injury.



Notice. If not avoided, the following could result in property damage.



Information messages provide additional information, emphasize or provide a useful tip.

Contents

Christie Spyder-S API Commands guide	7
Model names	7
Product documentation	7
Communicating with Spyder	8
Ethernet connectivity	8
Command format notations.....	8
Command considerations.....	8
Command processor responses	10
Mix Effect commands.....	11
MELA – Mix Effect Assign to Layer(s)	11
MELS – Mix Effect List.....	12
MESR – Mix Effect Source	13
METR – Mix Effect Transition	14
Layer commands – content, transition and geometry	15
AIR – Aspect Insensitive Resize	15
ARL – Apply Register to Layer	16
ARO – Aspect Ratio Offset	18
CRP – Crop Layer	19
DCK – Delete Command Key	20
FRZ – Freeze Layer(s).....	21
KBD – Border Adjust.....	22
KPS – Layer Position Change.....	23
KSH – Shadow Adjust	24
KSZ – Layer Size Change	25
KTL – Treatment Learn	26
KTP – Keyframe Transparency Apply	27
KTR – Treatment Recall.....	28
LAC – Layer Alignment Control	29
LAP – Layer Assign Pixelspace	32
LCC – Layer Clone Control	33
LCK – Learn Command Key.....	34
LSP – Layer Size and Position Change.....	35
SRA – Source Apply	36
SWA – Swap Layers.....	37
TRN – Transition Layers	38
ZPA – Zoom/Pan Adjust	39

Source, still and input commands	40
DSL – Detailed Source List.....	40
ICL – Input Config Learn	41
ICR – Input Configuration Recall.....	42
IGP – Input Get Properties.....	43
ILA – Input Level Adjust.....	44
ISP – Input Set Properties	45
RAR – Request Aspect Ratio.....	46
RAS – Request Active Sources.....	47
RLS – Request Layer Source	48
RSN – Request Source Name	49
SCL – Clear Still on Layer	50
SIP – Set Input Properties	51
SLD – Load Still on Layer.....	52
Background commands.....	53
BLD – Background Load Image.....	53
BTR – Background Transition	54
Keyframe and property commands	55
KGP – KeyFrame Get Properties.....	55
KSP – Keyframe Set Properties.....	56
RLK – Request Layer KeyFrame	57
Script, preset and register commands	59
ASC – Advance Script Cue	59
FKR – Function Key Recall	60
RLC – Request Layer Count.....	61
RPN – Request Page Names	62
RRC – Request Register Count.....	63
RRD – Request Register Detail.....	64
RRL – Request Register List	66
RSC – Recall Script Cue.....	67
RSCC – Request Script CueData Count.....	68
RSCD – Request Script CueData Details	69
RSEC – Request Script Element Count	70
SCR – Script Cue Request.....	71
SRS – Stop Running Scripts	72
Output commands.....	73
CSO – Clear Still on Output.....	73
LSO – Load Still on Output.....	74

OCC – Output Config Connection	75
OCF – Output Configuration Format.....	76
OCM – Output Configuration Mode	77
OFZ – Output Freeze	78
OGP – Output Get Properties.....	79
OSP – Output Set Properties	80
Router and device commands	81
DMB – Device Mixer Bus Apply	81
DMT – Device Mixer Transition	82
QRC – Query Router Crosspoint.....	83
RCR – Router Crosspoint Recall	84
PixelSpace and layout commands	85
PPE – Parallel PixelSpace Expansion	85
RPD – Request Pixelspace Definitions	86
RPM – Request Pixelspace Mappings.....	87
SLR – Slide Layout Recall	88
Test pattern commands.....	89
TPC – Test Pattern Clear.....	89
TPL – Test Pattern Load	90
System, status and hardware queries	91
RCM – Recall Continued Message	91
RCS – Request Connection Status	91
RFI – Request Frame Info.....	93
RHI – Request Hardware Information	94
RIF – Request Image File	95
RPS – Request I/O Processor Status.....	96
RSF – Request System FrameRate	97
SAV – Force Server Save.....	98
SDN – Restart Spyder Server	99
Multi-Viewer commands (capability-gated)	100
MVAC – Multi-Viewer Assign Content	100
MVAS – Multi-Viewer Add Single	101
MVCA – Multi-Viewer Clear All	102
MVKF – Multi-Viewer Set Keyframe Properties.....	103
MVPL – Multi-Viewer Preset Learn	104
MVPR – Multi-Viewer Preset Recall	105
MVQO – Multi-Viewer Query Output.....	106

MVST – Multi-Viewer Titling	107
OpMon and tiled output commands (capability-gated)	108
ATO – Apply Theme to Output Tile	108
OMSD – OpMon Set Dimensions	109
OMSL – OpMon Set Large Tile	110
OMSS – OpMon Set Small Tile	110
SRAG – Source Assign Group Tile.....	112
SRAT – Source Apply to Tile.....	113
Hardware data dumps.....	114
RHWC – Request Hardware Config Data Dump	114
RHWS – Request Hardware Status Data Dump	115
Commands not supported in 6.1.3	116

Christie Spyder-S API Commands guide

This guide contains information is designed to coincide with the release of Christie's image processor, the Spyder-S®.

For the latest approved updates to Spyder Studio, contact support at +1-602-943-5700 or techsupport-az@christiedigital.com.

Model names

This guide applies to the following models:

- Christie Spyder-S®

Product documentation

For installation, setup, and user information, see the product documentation available on the Christie website. Read all instructions before using or servicing this product.

1. Access the documentation from the Christie website:
 - a. Go to this URL: bit.ly/3HJtMIo or <https://www.christiedigital.com/products/image-processors/>.
 - b. Scan the QR code using a QR code reader app on a smartphone or tablet.



2. Select the image processor series.
3. On the product page, select the model and switch to the Downloads tab.

Communicating with Spyder

Understand the information and procedures for communicating with Spyder from a remote location. Connectivity is available using an RJ-45 (Ethernet) connection.

Ethernet connectivity

To control the Spyder frame, send the commands listed in this manual in a UDP packet sent to port 11116 on the frame; no configuration is required.

Each message sent to Spyder over UDP must be precluded by a 10-byte message header, provided in the table.

	Character
0	s
1	p
2	y
3	d
4	e
5	r
6	0x00 (hex)
7	0x00 (hex)
8	0x00 (hex)
9	0x00 (hex)

Command format notations

This document presents commands using the following conventional notations:

- Required parameters are enclosed in angle brackets (< >).
- Optional parameters are enclosed in square brackets ([]).
- Spaces separate the name and all arguments.

Example command notation:

```
Command Name <Required Arguments> [Optional Arguments]
```

Command considerations

Note the following about the commands:

Layer identifiers. Layer IDs 0 and 1 are reserved for the system's two background layers, so the first operator layer is layer 2. Commands that address a layer expect this numbering, which is one higher than the layer number shown in Spyder Studio. Sending a layer number that does not exist does not always produce an error — some commands return Success and do nothing — so read the value back with RLS, RLK or KGP when the result matters.

Output identifiers. Outputs are numbered from 0 in the API, which is one lower than the output number shown in Spyder Studio. The output Studio lists as 16 is output 15 in a command. Read an output back with OGP to confirm you have addressed the one you intended — the reply carries the output's name.

- When specifying layer IDs, the IDs start at 2. Layer ID 0 and 1 are reserved for future commands using the two background layers of the system.
- Replace the string argument spaces with the three-character ASCII string %20 to create a valid command, as the space is used as the argument delimiter.
- Do not put an argument delimiter (space character) between the header and the external control command.

Command processor responses

A response is returned for every command sent to the Spyder system.

If multiple values are returned in a single response, the argument delimiter is an ASCII space character and argument values containing a space are converted to the three-character ASCII string: %20.

The first response argument is always the error code for the command, which lets the user know if a command was successfully processed. The table below displays the various responses returned by the Spyder.

Response	Response name	Description
0	Success	The command was successfully processed.
1	Empty	The data requested is not available.
2	Header	An invalid command was specified.
3	Argument count	The command is missing the required minimum number of arguments.
4	Argument value	One or more arguments of the command were invalid.
5	Execution	An error occurred while processing the command. For details, check the Alert Viewer in Spyder Studio.
6	Checksum	Reserved.
7	Success with continuation	The command succeeded and the reply is longer than one message. The first chunk is returned with this code; retrieve the remainder with RCM. Repeat until the reply is complete.

Mix Effect commands

MELA - Mix Effect Assign to Layer(s)

Routes a Mix Effect onto one or more layers on the Program bus, so the layer takes its content from the Mix Effect rather than from a source assigned directly to it. One Mix Effect can drive several layers at once, so a single transition changes every screen it feeds.

Syntax

```
MELA <MixEffectID> <LayerID> [<LayerID> ...]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<MixEffectID>	ID of the Mix Effect to route.	Numeric ME ID, 0-based
<LayerID>	One or more internal layer indices. See the layer ID note below.	Numeric internal layer index, starting at 2

MELS - Mix Effect List

Lists the Mix Effects on the system. The reply gives the number of Mix Effects, then for each one its ID, name, type, and the sources currently loaded into its Top and Bottom inputs. Use it to find Mix Effect IDs before sending MESR or METR. A PixelSpace that has background groups appears here as a Mix Effect of type Background, holding the two background groups as its Top and Bottom inputs.

Syntax

MELS

Response

<Result Code> <Count> then, for each Mix Effect: <ID> <Name> <Type> <TopContent> <BottomContent>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

MESR - Mix Effect Source

Assigns a source to the Top or Bottom input of a Mix Effect. A Mix Effect holds two sources at once — one on air and one ready to come on. Load the incoming source into whichever input is not currently on air, then use METR to transition between them. Source names containing spaces are encoded with %20, so a source named Camera 1 is passed as Camera%201. For example, MESR 3 T Camera%201 loads Camera 1 into the Top input of Mix Effect 3.

Syntax

```
MESR <MixEffectID> <T|B> <SourceName>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<MixEffectID>	ID of the Mix Effect, as reported by MELS.	Numeric ME ID, 0-based
<T B>	Which ME input to load. T = Top, B = Bottom.	T or B
<SourceName>	Name of the source to assign. Encode literal spaces as %20.	Source name string, e.g. Camera%201

METR - Mix Effect Transition

Transitions one or more Mix Effects, swapping the Top and Bottom inputs so the source that was ready becomes the source on air. Add D followed by a frame count to set the transition length; if you omit it, the transition runs over 60 frames. Listing several Mix Effects in one command transitions them together, in sync. To cut rather than mix, use a duration of one frame — METR 3 D1. Backgrounds transition the same way: use the ID of the PixelSpace's Background Mix Effect, which MELS reports. Sending METR again transitions back.

Syntax

```
METR <MixEffectID> [<MixEffectID> ...] [D<Frames>]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<MixEffectID>	One or more Mix Effect IDs to transition together.	Numeric ME ID
D<Frames>	Optional duration in frames. Case-insensitive, clamped to a minimum of 1. Defaults to 60 when omitted.	D followed by a frame count, e.g. D30

Layer commands — content, transition and geometry

AIR - Aspect Insensitive Resize

Changes one dimension of a layer, its width or its height, without changing the other, by adjusting the layer's aspect-ratio offset. A layer that does not exist is skipped, and the command still returns result code 0.

Syntax

```
AIR <Relative> <Dimension> <NewSize> <LayerID> [<LayerID>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Relative>	Sets whether the new size replaces the current size or is added to it.	0 = Set the size to <NewSize> 1 = Add <NewSize> to the current size
<Dimension>	Selects the dimension to change.	0 = Width (Horizontal Size) 1 = Height (Vertical Size)
<NewSize>	The new size in pixels, or the amount to add when <Relative> is 1.	Size in pixels
<LayerID(s)>	One or more layers to apply the sizing change to	Layer ID(s) to set size on

ARL - Apply Register to Layer

Applies a stored register to one or more layers. Give the type of register, the register to apply, and the layers to apply it to. Use RRC to find how many registers of a type exist and RRL to list them; applying a register that does not exist returns an argument error.

This is useful for recalling register data types such as sources or treatments to layers.

For Source-type registers, if a targeted layer's current content is a Mix Effect, the source is routed to the Mix Effect's Top (Program) bus instead of replacing the layer. Other register types (Still, Effect, DDR, Treat, FuncKey) are unaffected. Append a trailing R to force the legacy replace-layer behavior.

Note: register IDs on the wire are zero-based, while Spyder Studio numbers the same registers from 1. RRL reports the true IDs — use those values here. Be aware that KTL does not use the same base: KTL 2 creates the register RRL reports as ID 1, whereas ARL 2 addresses ID 2. Learning a register with KTL n and recalling it with ARL n will therefore address the wrong register; read the IDs back with RRL first.

Syntax

```
ARL <RegisterType> <RegisterID> <LayerID> [<LayerID>] ... [R]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RegisterType>	Sets the register type.	0 (0x30) = Effect 1 (0x31) = Playitem 2 (0x32) = Not used 3 (0x33) = Not used 4 (0x34) = Command key/script 5 (0x35) = Treatment 6 (0x36) = Source 7 (0x37) = Function key 8 (0x38) = Not used 9 (0x39) = Not used 10 (0x31 0x30) = Still image
<RegisterID>	Sets the register ID to recall.	—
<Layer1>... <LayerX>	Sets the IDs of the layers to recall.	—
R	Optional. Forces legacy replace-layer behavior for Source-type registers instead of routing to the Mix Effect Top (Program) bus.	R

ARO - Aspect Ratio Offset

Adjusts the aspect ratio of the content on one or more layers. The type is a single letter: t to set the total aspect ratio, o to apply an offset, or a to set the aspect directly.

This command can also be used to set the total aspect ratio for a layer or adjust the KeyFrame aspect ratio offset directly.

Syntax

```
ARO <Type> <Value> <LayerID> [<LayerID>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Type>	Determines the type of aspect ratio being set.	t = Sets the total aspect ratio o = Sets the KeyFrame aspect ratio offset a = Adjusts the existing KeyFrame aspect ratio offset
<FloatingPoint>	Sets the floating-point aspect ratio value.	—
<Layer1>... <LayerX>	Sets the IDs of the layers to apply the aspect ratio offset to.	—

CRP - Crop Layer

Sets the left, right, top, and bottom cropping for the content of the specified layers.

Syntax

```
CRP <Left> <Right> <Top> <Bottom> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Left>	Sets the crop amount to the layer's left side.	0.0 = No crop 1.0 = 100% crop
<Right>	Sets the crop amount to the layer's right side.	
<Top>	Sets the amount to crop from the top of the layer.	
<Bottom>	Sets the amount to crop from the bottom.	
<Layer1>... <LayerX>	Sets the IDs of the layers to crop.	—

DCK - Delete Command Key

Deletes an existing command key. Give the key ID, and add the letter r after it to delete by register ID instead of script ID.

Syntax

```
DCK <ID> [r]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the ID to delete.	—
<Type>	Sets the type of ID. (Optional)	S = ScriptID (Default) R = RegisterID

FRZ - Freeze Layer(s)

Not supported on Spyder-S. Freeze applies to background Mix Effects only. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

```
FRZ <State> <LayerID> [<LayerID> ...]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<State>	Freeze state to apply.	0: unfreeze 1: freeze
<LayerID>	One or more internal layer indices.	Numeric internal layer index

KBD - Border Adjust

Adjusts one or more border properties on a layer. Give the layer first, then the border values.

Syntax

```
KBD <ID> <BorderThickness> <Red> <Green> <Blue> <H-BevelOffset> <V-BevelOffset>
<InsideSoftness>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the ID of the layer to adjust.	—
<BorderThickness>	Determines the thickness of the border. Set a negative number for outside softness.	-255 to 255
<Red>	Sets the red parameter for the RGB color value. If specifying the red parameter, the blue and green parameters must also be set.	0 to 255
<Green>	Sets the green parameter for the RGB color value. If specifying the green parameter, the blue and red parameters must also be set.	
<Blue>	Sets the blue parameter for the RGB color value. If specifying the blue parameter, the red and green parameters must also be set.	
<H-BevelOffset>	Sets the horizontal bevel offset.	
<V-BevelOffset>	Sets the vertical bevel offset.	
<InsideSoftness>	Sets the sharpness of the inside edge of the border.	

KPS - Layer Position Change

Sets the horizontal and vertical position of one or more layers. Note the following about positions: Position is SET in absolute pixels but REPORTED by KGP as a normalised value and by RLK in pixels. The three representations differ; read back with RLK when you need pixel values.

- Positions are mapped in pixels, relative to the top-left pixelspace corner associated with the layer.
- Position changes can be relative to the layer's current position or can be an absolute position setting.

Syntax

```
KPS <Position> <Horizontal> <Vertical> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Position>	Determines if the position is absolute or relative to the current position of the layer.	0 = Absolute position in pixels 1 = Relative position in pixels
<Horizontal>	Sets the horizontal position.	—
<Vertical>	Sets the vertical position.	—
<Layer1>... <LayerX>	Sets the IDs of the layers.	—

KSH - Shadow Adjust

Adjusts the shadow parameters for a layer. Give the layer first, then the shadow values. Setting any shadow value also switches the shadow on.

Syntax

```
KSH <LayerID> <Horizontal> <Vertical> <Size> <Transparency> <Softness>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the ID of the layer to adjust.	—
<Horizontal>	Sets the horizontal position of the shadow.	0 to 255
<Vertical>	Sets the vertical position of the shadow.	
<Size>	Sets the size of the shadow.	
<Transparency>	Sets the transparency of the shadow.	
<Softness>	Sets the outside softness of the shadow.	

KSZ - Layer Size Change

Sets the horizontal size of one or more specified layers.

The vertical size adjusts automatically to ensure the layer's aspect ratio.

Syntax

```
KSZ <Horz> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Horz>	Sets the horizontal size, in pixels, of the specified layers.	—
<Layer1>... <LayerX>	Sets the IDs of the layers to size.	—

KTL - Treatment Learn

Stores a layer's treatment in a register so it can be applied to other layers later. Give the register to write into, the layer to learn from, and a name for the register. Writing to a register that already holds a treatment replaces it. Use RRL to list the treatments already stored.

Syntax

```
KTL <RegisterID> <LayerID> <Name>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RegisterID>	Sets the treatment register to store the layer's treatment in.	Numeric register ID
<LayerID>	Sets the layer to learn the treatment from.	Numeric layer ID
<Name>	Sets the name to store the treatment under.	Text; encode spaces as %20

KTP - Keyframe Transparency Apply

Sets KeyFrame transparency value on layers.

Syntax

KTP <TransparencyValue> [<a|r>] <LayerID(s)>

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<TransparencyValue>	Sets the transparency value.	0 to 255
[<a r>]	Optional. How the value is applied. If omitted, the transparency is set to the value.	a = Set the transparency to the value Any other letter (for example r) = Add the value to the current transparency
<LayerID (s)>	Which layer or layers to apply transparency value.	Layer IDs are 2 based. For example, Layer 1 is ID 2, Layer 5 is ID 6.

KTR - Treatment Recall

Recalls a treatment to one or more layers.

Syntax

```
KTR <TreatmentID> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<TreatmentID>	Sets the ID of treatment to recall.	—
<Layer1> .. <LayerX>	Sets the IDs to recall the treatment to.	—

LAC - Layer Alignment Control

Allows one or more layer KeyFrames to be manipulated by the Spyder Studio to achieve a specified alignment effect. The alignment effects are the same as the ones available from the simulator control of Spyder Studio.

Syntax

```
LAC <EffectID> <RecallDuration> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values	Minimum Layers
<EffectID>	Sets the alignment effect.	0 (Align Bottom) = Repositions specified layers vertically to make the bottom edge of the layers match the first specified layer	2
		1 (Align Center) = Repositions specified layers horizontally to make the center position of all layers match the first specified layer	
		2 (Align Left) = Repositions specified layers horizontally to make the left edge of all layers match the first specified layer	
		3 (Align Middle) = Repositions specified layers vertically to make the center position of all layers match the first specified layer	
		4 (Align Right) = Repositions all specified layers horizontally to make the right edge of all layers match the first specified layer	1
		5 (Align Top) = Repositions specified layers vertically to make the top edge of layers match the first specified layer	
		6 (Center Horizontal) = Repositions all specified layers as a group so that layers are centered horizontally in their current pixelspace	
		7 (Center Vertical) = Repositions all specified layers as a group so that	

		layers are centered vertically in their current pixelspace	
		8 (Horizontal Decrement) = Decreases the horizontal spacing between two or more layers	2
		9 (Horizontal Increment) = Increases the horizontal spacing between two or more layers	
		10 (Make Horizontal Equal) = Makes the horizontal spacing between all specified layers equal to the spacing between the first and second specified layers	3
		11 (Make Same Height) = Makes all specified layers the same height as the first specified layer, maintaining aspect ratio in all layers	2
		12 (Make Same Width) = Makes all specified layers the same height as the first specified layer, maintaining aspect ratio in all layers	
		13 (Make Vertically Equal) = Makes the vertical spacing between all specified layers equal to the spacing between the first and second specified layers	3
		14 (Remove Horizontal Spacing) = Removes spacing between specified layers, causing them to be horizontally stacked in the order specified	2
		15 (Remove Vertical Spacing) = Removes spacing between specified layers, causing them to be vertically stacked in the order specified	
		16 (Size to Display Height) = Resizes specified layers to fill their respective pixelspace vertically; no adjustment is made to the horizontal position	1
		17 (Size to Display Width) = Resizes specified layers to fill their respective pixelspace horizontally, and centers the input vertically	
		18 (Snap to Bottom) = Repositions specified layers vertically to align bottom edges of layers with the bottom of their pixelspace	
		19 (Snap Left) = Repositions the specified layers horizontally to align	

		left edges of layers with the left edge of their pixelspace	
		20 (Snap Right) = Repositions the specified layers horizontally to align right edges of layers with the right edge of their pixelspace	
		21 (Snap Top) = Repositions the specified layers vertically to align top edges of the layers with the top edge of their pixelspace	
		22 (Stack Horizontal) = Makes all layers the same height, centers them vertically with the first specified layer, and positions them in a horizontal array running to the right of the first layer	2
		23 (Stack Vertical) = Makes all layers the same width, centers them horizontally with the first specified layer, and positions them in a vertical array running downward starting at the first layer's position	
		24 (Swap Windows) = Swaps horizontal position and size between the two specified layers	
<RecallDuration>	Sets the duration for the alignment recall, in frames per second.	—	—
<Layer1> .. <LayerX>	Sets the IDs of the layers to crop.	—	—

LAP - Layer Assign Pixelspace

Assigns one or more layers to a PixelSpace. Give the PixelSpace, then 1 to make the layers visible on it or 0 to leave them hidden, then the layers to move.

Syntax

```
LAP <PixelSpaceID> <Visible> <LayerID> [<LayerID>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the pixelspace to associate the layer with.	—
<Visibility>	Hides or shows the layers at the time this command is applied. Making the layer visible allows additional commands to be sent to configure the layer before it is transitioned	0 = Hides the layers (Default) 1 = Makes the layers visible
<Layer1>... <LayerX>	Sets the IDs of the layers to associate with the pixelspace.	—

Related information

TRN—Transition Layers (**page 37**)

LCC - Layer Clone Control

Adjusts the clone property on the KeyFrame of a specified layer.

Syntax

```
LCC <LayerID> <Mode> [<Offset>] <OffsetType>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LayerID>	Sets the ID of the layer that the KeyFrame is on.	—
<Mode>	Sets the clone mode. If the mode is specified with no offset value, the clone offset value stored in the layer's current KeyFrame is used.	0 = Off 1 = Offset 2 = Mirror
<Offset>	Sets the offset distance in relative coordinates or the pixel value for the clone position. (Optional)	—
<OffsetType>	Determines the type of clone offset.	0 = Absolute (in pixels) 1 = Relative coordinate (Default)

LCK - Learn Command Key

Creates a command key from the current state, storing it under the name given. The reply returns the page and key position the new command key was placed at. Use RSCC and RSCD to inspect the script it captured.

Syntax

LCK <KeyID> <Name> <Argument3> <Argument4> <Argument5>

Response

<Result Code> <CommandKey ID> <Script ID>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LearnAs>	Determines if the command key is learned as absolute or relative.	0 = Absolute 1 = Relative
<Name>	Sets the name of the command key to be learned.	—
<RegisterID>	Sets the ID of the register to learn the command to.	—
<LearnFrom>	Determines what the command key is learned from.	1 = Preview only 2 = Program only 3 = Both
<LearnMixers>	Determines if the command key is learned as a mixer.	0 = False 1 = True

LSP - Layer Size and Position Change

Sets the size and horizontal and vertical position of one or more layers.

In cases where the size and position are being modified simultaneously for an application, Christie recommends using this command instead of the individual layer size and position commands (KPS and KSZ).

Syntax

```
LSP <Position> <Horizontal> <Vertical> <Size> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Position>	Determines the position.	0 = Absolute position 1 = Relative position to the current position of the layer
<Horizontal>	Sets the horizontal position in pixels.	—
<Vertical>	Sets the vertical position in pixels.	—
<Size>	Sets the horizontal layer size in pixels.	—
<Layer1>... <LayerX>	Sets the IDs of the layers.	—

Related information

KPS – Layer Position Change (**page 22**)

KSZ – Layer Size Change (**page 24**)

SRA - Source Apply

Applies an existing source to one or more specified layers.

A source is defined as an input configuration and a router input. To create sources and define connected routers, use Spyder Studio.

If a targeted layer's current content is a Mix Effect, the source is applied to the Mix Effect's Top (Program) bus so it is immediately visible. Top is the default for external control because external control does not operate on preview layers. Append a trailing R to force the legacy replace-layer behavior.

Syntax

```
SRA <Name> <Layer1> [<Layer2>] ... [R]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Name>	Provides the name of the source.	—
<Layer1>... <LayerX>	Sets the IDs of the layers to apply the source to.	—
R	Optional. Forces legacy replace-layer behavior instead of routing the source to the Mix Effect Top (Program) bus when the layer holds a Mix Effect.	R

SWA - Swap Layers

Swaps the size, position and keyframe settings of two layers. Both layers must have content; naming an empty layer returns an argument error. The sources assigned to the layers stay where they are — it is the layer geometry that exchanges.

Note: Both layers must be visible on screen when the command is executed, or an ArgValue response will be returned.

Syntax

```
SWA <LayerID> <LayerID>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<First Layer ID>	First layer of the two-layer swap.	Numeric layer ID
<Second Layer ID>	Second layer of the two-layer swap.	Numeric layer ID

TRN - Transition Layers

Transitions layers on and off their currently assigned PixelSpace. TRN does not toggle: sending the same transition a second time does not return the layer to where it was, it takes the layer off both Program and Preview. To bring a layer back, reassign its content.

Syntax

```
TRN <Mix> <Duration> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Mix>	Transitions layers on and off their currently assigned pixelspace.	0 = Mix off 1 = Mix on
<Duration>	Sets the transition duration in number of frames per second.	—
<Layer1> ... <LayerX>	Sets the IDs of the layers to transition.	—

ZPA - Zoom/Pan Adjust

Sets the zoom and pan of a layer. Give the mode, the zoom value, the horizontal pan, the vertical pan, and then the layer. Zoom is limited to a maximum of 20; a larger value is clamped rather than rejected. Read the result back with RLK.

Syntax

ZPA <Mode> <Zoom> <PanH> <PanV> <LayerID>

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RecallMode>	Determines if adjustments are relative to the existing zoom/pan settings of the layer or can be set to specific values directly.	0 = Absolute values 1 = Relative adjustment
<Zoom>	Sets the zoom value.	0.0 to 20.0
<HorizontalPan>	Sets the horizontal pan value.	-2048 to 2048
<VerticalPan>	Sets the vertical pan value.	-2048 to 2048
<LayerID>	Sets the layer of the keyframe to adjust.	—

Source, still and input commands

DSL - Detailed Source List

Returns a detailed source list.

Syntax

DSL

Response

Lists count and six information fields for each source:

<Count> <RegisterID> <SourceName> <HActive> <VActive> <Connector> <Thumbnail>

Arguments

None.

ICL - Input Config Learn

Saves the input configuration for a layer ID to a configuration ID for later recall.

Syntax

```
ICL <Input> <LayerID>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Input>	Sets the input configuration ID to save to.	—
<LayerID>	Determines the layer ID to save the input configuration from.	—

ICR - Input Configuration Recall

Applies a stored input configuration to a layer. A configuration ID below zero is accepted and does nothing, which lets a control system send the command unconditionally.

Syntax

```
ICR <ConfigurationID> <LayerID>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ConfigurationID>	Sets the input configuration to apply.	Numeric configuration ID
<LayerID>	Sets the layer to apply the input configuration to.	Numeric layer ID

IGP - Input Get Properties

Returns every property of the input configuration used by the source on a layer, formatted as a list. The reply includes the connector, active resolution, refresh rate, colour space and related settings.

Syntax

```
IGP <LayerID>
```

Response

```
<Property Name> <Property Type> <Current Value>
```

Arguments

Argument	Description	Values
<LayerID>	Sets the layer whose input configuration is returned.	Numeric internal layer index, starting at 2

ILA - Input Level Adjust

Adjusts the image levels of an input. On Spyder-S the Gamma value is applied to the hardware; the remaining level controls are accepted for compatibility with earlier Spyder systems but do not change the image.

Syntax

```
ILA <InputID> <Gamma> <Brightness> <Contrast> <Saturation>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<InputID>	ID of the input to adjust.	Numeric input ID
<Gamma>	Gamma level. This is the only level applied to Spyder-S hardware.	Numeric gamma value
<Brightness>	Brightness level. Accepted but not applied on Spyder-S.	Numeric
<Contrast>	Contrast level. Accepted but not applied on Spyder-S.	Numeric
<Saturation>	Saturation level. Accepted but not applied on Spyder-S.	Numeric

ISP - Input Set Properties

Sets the named properties on the input config object for the specified layer.

Syntax

```
ISP <LayerID> <PropertyName> <PropertyValue> [<PropertyName> <PropertyValue>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LayerID>	Sets the layer whose input configuration is changed.	Numeric layer ID
<PropertyName>	Sets the name of the property to change. At least one name and value pair is required; further pairs may be added to change several properties in one command.	Property name, as reported by IGP
<PropertyValue>	Sets the new value for the property.	Value matching the property's type

RAR - Request Aspect Ratio

Requests the aspect ratio of a defined source.

Syntax

RAR <Layer>

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
RAR <Layer>	Sets the ID of the layer or the source name to request the aspect ratio	—

RAS - Request Active Sources

Allows for lightweight status request for sources visible in program and/or preview pixelspaces. Useful for showing active status for Spyder sources in a third-party user interface, or it can be used with an external system to trigger a tally device.

Syntax

```
RAS [<PixelSpace ID> ...]
```

Response

```
<Result Code> <Count> [<Source Name> <Bus> ...]
```

Arguments

Argument	Description	Values
<PixelSpace ID>	Sets the PixelSpace ID(s) to query the active sources.	Default = all pixelspaces

Returned values

Response	Description	Values
Source	String name of source.	
Bus	Indicates if the specified source is visible on a program pixelspace, a preview pixelspace, or both.	0 = Preview 1 = Program 2 = Both

RLS - Request Layer Source

Retrieves the current source name and associated source register ID loaded on a specified layer. Note the following about returned results:

- If no source is currently assigned to the specified layer, an empty result code is returned with no parameters.
- If a source is assigned to the specified layer but no corresponding register can be found, 1 is returned for the register ID.

Syntax

```
RLS <LayerID>
```

Response

<Result Code> <Source Name> <Source Register ID>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LayerID>	Specifies the ID of the layer to retrieve the information for.	—

RSN - Request Source Name

Retrieves a list of sources defined within Spyder.

Syntax

RSN

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
RSN	Returns a list of sources names separated by spaces.	—

SCL - Clear Still on Layer

Not supported on Spyder-S. Stills are a background feature; they cannot be placed on a layer or an output. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

```
SCL <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Layer1>... <LayerX>	Sets the IDs of the layers.	—

SIP - Set Input Properties

Sets a property on an input. Give the input ID, the property name, and the value. On Spyder-S the ColorSpace property is applied to the hardware.

Syntax

```
SIP <InputID> <PropertyName> <PropertyValue>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<InputID>	ID of the input.	Numeric input ID
<PropertyName>	Name of the property to set.	Property name string
<PropertyValue>	Value to apply.	Value appropriate to the property

SLD - Load Still on Layer

Not supported on Spyder-S. Stills are a background feature; they cannot be placed on a layer or an output. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

```
SLD <Filename> <Layer1> [<Layer2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Filename>	Sets the name of the file to load.	—
<Layer1>... <LayerX>	Sets the IDs of the layers.	—

Background commands

BLD - Background Load Image

Loads a background image into a PixelSpace. Backgrounds are configured in the Configuration Manager; a PixelSpace with two background groups can transition between them with METR.

Syntax

```
BLD <PixelSpaceID> <Argument2> <Argument3>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<PixelSpaceID>	ID of the target PixelSpace.	Numeric PixelSpace ID
<Argument2>	Background selector.	Numeric
<Argument3>	Background selector.	Numeric

BTR - Background Transition

Transitions the background of a PixelSpace. To transition a background, use METR with the ID of the PixelSpace's Background Mix Effect, which MELS reports.

Syntax

```
BTR <PixelSpaceID> [D<Frames>]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<PixelSpaceID>	ID of the target PixelSpace.	Numeric PixelSpace ID
D<Frames>	Optional duration in frames.	D followed by a frame count, e.g. D30

Keyframe and property commands

KGP - KeyFrame Get Properties

Returns all KeyFrame properties for the specified object. The properties are formatted in a list.

Syntax

KGP <LayerID>

Response

<Property Name> <Property Type> <Current Value>

Arguments

Argument	Description	Values
<Layer ID>	Determines the properties of the source that the layer is using.	2 - 26

KSP - Keyframe Set Properties

Sets the named properties on the Keyframe object for a specified layer.

Syntax

```
KSP <LayerID> <PropertyName> <PropertyValue> [<PropertyName> <PropertyValue>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LayerID>	Sets the layer whose KeyFrame is changed.	Numeric ID
<PropertyName>	Sets the name of the property to change. At least one name and value pair is required; further pairs may be added to change several properties in one command.	Property name, as reported by KGP
<PropertyValue>	Sets the new value for the property.	Value matching the property's type

RLK - Request Layer KeyFrame

Retrieves the KeyFrame values for a layer ID.

Christie recommends implementing the code to handle additional values which may be appended to the end of the responses in future versions.

Syntax

RLK <LayerID> [n]

Response

<Result Code> <Relative HPosition> <Relative VPosition> <X Position> <Y Position>

<Width> <Height> <Border Thickness> <Border Red> <Border Green> <Border Blue> <Border HBezel Offset> <Border VBezel Offset> <Border Inside Softness> <Border Outside Softness> <OutSide Edges> <Shadow HOffset> <Shadow VOffset> <Shadow HSize> <Shadow Softness> <Shadow Transparency> <Clone Mode> <Clone Offset> <Left Crop> <Right Crop>

<Top Crop> <Bottom Crop> <Crop Anchor> <AR Offset><Zoom> <HPan> <VPan>

<PixelSpace ID> <Transparency>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<LayerID>	Specifies the ID of the layer to retrieve the KeyFrame values	—
[n]	Optional. For a layer on a Preview PixelSpace, returns its geometry scaled to the Program PixelSpace. Has no effect on a Program layer.	n or normalize

Returned values

Response	Values
<Outside Edges>	Provided as a hexadecimal value, with the lower four-bits indicating specific edges enabled. • 0x01—Top enabled • 0x02—Bottom enabled • 0x04—Left enabled • 0x08—Right enabled
<Clone Mode>	0 = Off 1 = Offset 2 = Mirror
<Crop Anchor>	0 = Input center 1 = Window center

Script, preset and register commands

ASC - Advance Script Cue

Advances the current script cue by steps.

Syntax

ASC [<Steps>]

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Steps>	Number of steps (optional)	<i>None (no number)</i> – advances the current script cue by one. <i>n</i> - to go forward <i>-n</i> - to go backward (default is 1)

FKR - Function Key Recall

Recalls a single function key defined in Spyder.

Note: To access registers on pages above the first page, add the page number * 1000 to the register ID to be recalled. For example, to recall a function key at register ID 2 on page 3, send a register ID of 3002.

Syntax

```
FKR <ID> [<Layer1>] [<LayerX>] [<IDType>]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the ID of the function key.	—
<Layer1> .. <LayerX>	Sets the IDs of the layers for the relative function keys. (Optional)	—
<IDType>	Defines the ID type being recalled. If this argument is not specified, the function key ID is used. (Optional)	F = Function key ID (Default) R = RegisterID

RLC - Request Layer Count

Retrieves the logical layer count within the connected Spyder system.

Syntax

RLC

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
RLC	Retrieves the logical layer count within the connected Spyder system. The logical layer count includes two background layers in Spyder. In most cases, applications should subtract two from the response.	—

RPN - Request Page Names

Returns the names and page numbers for a specified register list. Command key and function key lists return data.

Syntax

RPN <RegisterType>

Response

<Result Code> <Register Count> [<Register Page ID> <Register Page Name>]...

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RegisterType>	Sets the register type.	0 (0x30) = Effect 1 (0x31) = Playitem 2 (0x32) = Not used 3 (0x33) = Not used 4 (0x34) = Command key/script 5 (0x35) = Treatment 6 (0x36) = Source 7 (0x37) = Function key 8 (0x38) = Not used 9 (0x39) = Not used 10 (0x31 0x30) = Still image

RRC - Request Register Count

Returns how many registers of a given type are defined. Use it before RRL to size the reply, or to check whether a register type holds anything at all.

Syntax

```
RRC <RegisterType>
```

Response

```
<Result Code> <Register Count>
```

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RegisterType>	Sets the type of register to count.	0 5 = Treatments 6 = Sources

RRD - Request Register Detail

Returns detailed information specific to the data stored within a specified register type and ID. The format of the response data is dependent on the register type requested, and the individual response formats register types supported under this command are documented below. Data associated with register types may be enhanced in the future, and callers supporting this function should implement their code to handle additional information which may be appended to the end of the responses in future versions of the Spyder server.

Syntax

```
RRD <RegisterType> <Register ID>
```

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Response (Cmd Keys)

```
<Result Code> <Script ID> <0=Absolute/1= Relative> <Cue Count>
```

Response (Treatments)

```
<Result Code> <Size Enabled (1/0) > <Position Enabled (1/0)> <Border Enabled (1/0)>  
<Shadow Enabled (1/0)> <Clone Enabled (1/0)> <Crop Enabled (1/0)> <A/R Enabled (1/0)>  
<Pan/Zoom Enabled (1/0)> <Duration (1/0)>
```

Response (Source)

```
<Result Code> <Source Name> <Router ID> <Router Input> <Input Config ID>  
<Preferred Treatment ID> <Preferred Layer ID> <HActive> <VActive> <Frame  
Rate>
```

Response (Function Key)

```
<Result Code> <Function Key ID> <Function Key Type (string name)> <User Defined Name>
```

Response (Still)

```
<Result Code> <FileName> <Exists at server (0/1)> <Width> <Height> <File  
Size (bytes)>
```

Argument	Description	Values
----------	-------------	--------

<RegisterType>	Sets the register type	0 (0x30) = Effect 1 (0x31) = Playitem 2 (0x32) = Not used 3 (0x33) = Not used 4 (0x34) = Command key/script 5 (0x35) = Treatment 6 (0x36) = Source 7 (0x37) = Function key 8 (0x38) = Not used 9 (0x39) = Not used 10 (0x31 0x30) = Still image
<RegisterID>		

RRL - Request Register List

Lists the registers of a given type. The reply gives the number of registers, then each register's ID and name. Treatments and sources are held in separate register types, so query the type you need.

As register lists can be long, additional arguments are available to request portions of the list at a time.

Syntax

```
RRL <RegisterType>
```

Response

```
<Result Code> <Return Count> [<Register1 ID> <Register1 Name>]...
```

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RegisterType>	Sets the type of register to list.	0 5 = Treatments 6 = Sources

RSC - Recall Script Cue

Recalls an existing script at a specific cue. To build scripts, use Spyder Studio.

Syntax

RSC <Script> <Cue> [<Type>]

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Script>	Sets the ID of the script being recalled.	—
<Cue>	Sets what script cue to recall.	—
[Type]	Defines the ID type being recalled being recalled in the <Script> argument. (Optional) To access registers on pages above the first page, add the page number * 1000 to the register ID to be recalled.	S = ScriptID (Default) R = RegisterID

RSCC - Request Script CueData Count

Returns the number of cues in a script. Allow extra time for the reply; this command takes noticeably longer than most.

Syntax

```
RSCC <ScriptID> [R]
```

Response

<Number of cues in the requested script>

Arguments

Argument	Description	Values
<ScriptID>	Sets the script whose cues are counted.	Numeric script ID
[R]	Optional. Treats the ID as a command key's register ID instead of a script ID.	R

RSCD - Request Script CueData Details

Returns the detail of one cue in a script: the cue type, its indexes, its timecode, and its state. Use RSCC first to find how many cues the script has.

Syntax

```
RSCD <ScriptID> <CueIndex> [R]
```

Response

<Name> <JumpType> <JumpCue> <TriggerType> <TriggerTime> <Function Key Count> [<Function Key Register ID>...] <PlayItemType> [<PlayItem Register ID> <PlayItem Command>...]

Response	Codes and Format
JumpType	None=0 Direct=1
TriggerType	None=0 Wait=1 WaitX=2 Time=3 Time Of Day=4
TriggerTime	HH:MM:SS.FF (Hours:Minutes:Seconds.Frames)
PlayItem	Cue=0 Play=1 Stop=2

Arguments

Argument	Description	Values
<ScriptID>	Sets the script holding the cue.	Numeric script ID
<CueIndex>	Sets the cue within the script to return.	Numeric index
[R]	Optional. Treats the ID as a command key's register ID instead of a script ID.	R

RSEC - Request Script Element Count

Returns the number of elements in a specified script.

Syntax

```
RSEC <ScriptID> [R]
```

Response

<Number of elements in the requested script>

Arguments

Argument	Description	Values
<ScriptID>	Sets the ID of the script to count the elements of.	Numeric script ID
[R]	Optional. Treats the ID as a command key's register ID instead of a script ID.	R

SCR - Script Cue Request

Requests the current execution cue of a specified script ID.

Returns a single integer value representing the current cue of the specified script or the command returns -1 if the specified script is not being executed on any layer.

Syntax

```
SCR <ID> [<Type>]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ID>	Sets the script ID to request the status for.	—
<Type>	Defines the ID type for the specified script.	S = ScriptID R = RegisterID

SRS - Stop Running Scripts

Allows for either stopping running scripts (cue trigger countdowns) or disconnects scripts from layers.

Syntax

SRS [<Operation Type>]

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Operation Type>	Specifies whether to cancel cue triggers or to disconnect layers from scripts.	D: Disconnect layers from scripts S: Cancel pending cue triggers

Output commands

CSO - Clear Still on Output

Not supported on Spyder-S. Stills are a background feature; they cannot be placed on a layer or an output. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

CSO <OutputID>

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the output to clear.	Numeric output ID

LSO - Load Still on Output

Not supported on Spyder-S. Stills are a background feature; they cannot be placed on a layer or an output. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

```
LSO <OutputID> <StillID>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the target output.	Numeric output ID
<StillID>	Identifier of the still to load.	Numeric still ID

OCC - Output Config Connection

Sets the connection configuration of an output. The output must already have an output configuration.

Syntax

OCC <OutputID> <Argument2> <Argument3>

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the output.	Numeric output ID
<Arguments>	Connection parameters.	Numeric

OCF - Output Configuration Format

Sets the output format for a specified output. Changing the format also updates the blanking and bit depth that belong to it; setting the original refresh rate again does not restore those values. Use OSP to set them individually if you need to return to a previous configuration.

Currently this command supports configuring VESA output formats only. Additionally, Spyder outputs support specific output refresh rates, and in cases where an unsupported refresh rate is specified, the closest available refresh value is used.

Syntax

```
OCF <OutputID> <HActive> <VActive> <RefreshRate> <Interlaced> [<Timing>]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the output ID.	Zero-based
<HActive>	Sets the horizontal active resolution.	—
<VActive>	Sets the vertical active resolution.	—
<RefreshRate>	Sets the refresh rate. Supports floating point	—
<Interlaced>	Sets the interlaced format.	0 = Non-interlaced 1 = Interlaced
<Timing>	Sets the reduced blanking timing.	0 = Off 1 = Reduced Level 1 2 = Reduced Level 2

OCM - Output Configuration Mode

Sets the output mode for a specified output. The mode is given by name: normal, unused, scaled, opmon, sourcemon, aux, unscaledaux or multiviewer. The arguments that follow depend on the mode being applied.

The arguments for this command vary depending on the mode being applied. Not every mode is available on every output. OpMon mode requires an output whose hardware supports it, and returns an argument error on an output that does not. Use OGP to check an output's current mode before changing it.

Syntax for Normal Mode

```
OCM <OutputID> <Mode> [<HStart> <VStart>]
```

Response for Normal Mode

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments for Normal mode

Argument	Description	Values
<OutputID>	Sets the ID of the output.	Zero-based
<Mode>	Sets the output mode to Normal.	Normal
<HStart>	Sets the horizontal starting position for the output.	—
<VStart>	Sets the vertical starting position for the output.	—

Syntax for OpMon Mode

```
OCM <OutputID> <Mode> [<PixelspaceID>]
```

Response for OpMon Mode

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments for OpMon Mode

Argument	Description	Values
<OutputID>	Sets the ID of the output.	Zero-based
<Mode>	Sets the output mode to OpMon (X20) or Multiviewer with OpMon preset type (X80).	OpMon
<PixelspaceID>	Optional. Sets the program pixelspace to focus OpMon. If a preview ID is supplied, its associated program pixelspace is applied. Default: ALL PGM/PVW.	—

Syntax for UnscaledAux Mode

OCM <OutputID> unscaledaux <MixEffectID>

Response for UnscaledAux Mode

<Result Code>

Arguments for UnscaledAux Mode

Argument	Description	Values
<OutputID>	ID of the output to configure. The output must report OutputCapability.UnscaledAuxMode.	
unscaledaux	Sets the output to UnscaledAux mode driven by a Mix Effect (OutputMode.UnscaledAux, UnscaledAuxSourceType = MixEffect).	
<MixEffectID>	ID of the Mix Effect that drives the output (UnscaledAuxSourceId).	

OFZ - Output Freeze

Not supported on Spyder-S. Freeze applies to background Mix Effects only. The command is accepted for compatibility with earlier Spyder systems but has no effect.

Syntax

OFZ <OutputID> <State>

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the output.	Numeric output ID
<State>	Freeze state to apply.	0: unfreeze 1: freeze

OGP - Output Get Properties

Returns the properties of an output, formatted as a list. The reply includes the name, connector, active resolution, timing, colour settings and the output mode. For an output in OpMon or Multi-Viewer mode the reply also carries the theme identifier and the Full, Halves and Quarters tile group definitions.

Syntax

OGP <Output ID>

Response

<Property Name> <Property Type> <Current Value>

Arguments

Argument	Description	Values
<Output ID>	Determines the output config object ID.	-0 - 15

OSP - Output Set Properties

Sets named properties on the output config object for the specified output.

The property names accepted here are the ones OGP returns for the same output — for example Name, Mode, HActive, VActive, RefreshRate, HStart and VStart. At least one name/value pair is required; sending the output ID alone returns result code 3.

Syntax

```
OSP <Output ID> <PropertyName> <PropertyValue> [<PropertyName> <PropertyValue>]...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Output ID>	Determines the output config object ID.	0 - 15

Router and device commands

DMB - Device Mixer Bus Apply

Allows the active bus to be selected for one or more control devices defined in the system. This command is intended for devices configured as mixers.

Syntax

```
DMB <Duration> <Bus> <DeviceIndex1> [<DeviceIndex2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Duration>	Sets the transition duration to be applied to layers coming on or off the program, in frames per second.	—
<Bus>	Sets the bus state to apply to the device.	OFF = Both layers for a device are off the screen PVW = The program layer of the device is in preview and the preview layer is off the screen PGM = Both the program and preview layers are visible in their respective pixelspaces.
<DeviceIndex1> ... <DeviceIndexX>	Indicates the target device indexes.	Zero-based

DMT - Device Mixer Transition

Sets the automatic transition for a mixer one or more devices. This command is intended for devices configured as mixers.

Syntax

```
DMT <Duration> <DeviceID1> [<DeviceID2>] ...
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Duration>	Sets the transition duration to be applied, in frames per second.	1 = Forces a cut transition
<DeviceID1> ... <DeviceIDX>	Indicates the target device indexes.	Zero-based

QRC - Query Router Crosspoint

Queries one or all router outputs for their currently connected input. Note the following about inputs and outputs:

- For level-controlled routers, the input returned is the input connected to the output on the level configured to be controlled by Spyder.
- If no specific output is supplied, the inputs for all outputs are returned.
- Inputs and outputs returned are zero indexed. For example, output 1 on the router is returned as a zero (0).
- An output with no input connected (disconnected) returns -1 for the input.

Syntax

```
QRC <RouterID> [<OutputID>]
```

Response

```
<Result Code> <Router ID> <Output>:<Input> [<Output>:<Input>]
```

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RouterID>	Sets the router ID to query.	—
<OutputID>	Sets the router output to query the status for.	—

RCR - Router Crosspoint Recall

Switches crosspoints on a router connected to the Spyder frame.

The <output> and <input> arguments can be repeated as many times as required to stack a series of switches into a single command. If the router and the Spyder control protocol for the router support stack and trigger switching, Spyder uses this functionality automatically when sending a command with multiple switch assignments.

Syntax

```
RCR <RouterID> [<Switch>] <Output> <Input>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<RouterID>	Sets ID of the router to switch.	—
<Switch>	Determines which patch is switched. If this argument is not specified, the logical patch is switched. (Optional) A logical output refers to the router patch configured from Spyder Studio, stored as part	L = Switch logical Output (Default) P = Switch Physical Outputs
<Output>	Sets the output to switch.	Zero-based
<Input>	Sets the input to switch.	Zero-based

PixelSpace and layout commands

PPE - Parallel PixelSpace Expansion

Applies a configuration change that has already been staged, expanding PixelSpaces across frames in parallel. The single argument is the time the transition takes, in milliseconds. If no configuration change is waiting to be applied, the command returns an execution error.

Syntax

```
PPE <DurationMs>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<DurationMs>	Sets the time the transition takes to complete, in milliseconds.	Numeric; 0 applies the change immediately

RPD - Request PixelSpace Definitions

Returns a list of all the pixelspaces currently defined in the system.

Syntax

RPD

Response

<Result Code> <Count> [<ID> <Name> <Current Background> <Next Background> <Xpos> <Ypos> <Width>
<Height> <RenewalGroup ID>]...

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
RPD	Returns a list of all the pixelspaces currently defined.	—

RPM - Request PixelSpace Mappings

This command returns the mappings between preview and program pixelspace IDs. Pixelspaces with no associated preview/program pixelspace will not be included in the response.

Syntax

RPM

Response

<Result Code> <Count> [<ProgramID> <PreviewID> <Preview Scale Ratio>]...

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

None.

SLR - Slide Layout Recall

Runs a slide-style transition of layers onto a PixelSpace, in the manner of a presentation program. Layers are chosen automatically as the slide runs; any layer you want left alone — a background layer, for example — is listed as a reserved layer and will not be used. Give the PixelSpace, whether unused layers are cleared, how many reserved layers follow, then the reserved layers, then the slide entries.

Syntax

```
SLR <PixelSpaceID> <ClearLayers> <ReservedLayerCount> [<ReservedLayer> ...]
[<SlideLayoutEntry> ...]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<PixelSpaceID>	ID of the PixelSpace on which to run the slide operation.	Numeric PixelSpace ID
<ClearLayers>	Determines whether unused layers are removed.	0: leave unused layers 1: remove unused layers
<ReservedLayerCount>	Number of reserved layers listed in the arguments that follow.	Numeric count
<ReservedLayer>	Zero or more layer IDs that must not be used for slide operations.	Numeric layer ID
<SlideLayoutEntry>	Zero or more slide operations, each using its own tilde (~) separated sub-syntax.	Tilde-separated entry

Test pattern commands

TPC - Test Pattern Clear

Clears a test pattern from an output, returning it to its normal content.

Syntax

TPC <Target> <Argument2>

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Target>	Target to clear.	Numeric target ID
<Argument2>	Pattern selector.	Numeric

TPL - Test Pattern Load

Places a test pattern directly on a physical output, which is useful for confirming cabling or identifying a port before content is up. Use TPC to clear it. The first test pattern loaded after the system starts can take several seconds to appear while the pattern is prepared; later ones appear immediately. If your controller uses a short reply timeout, allow extra time for the first test pattern of a session.

Syntax

```
TPL <Target> <Argument2> <Argument3>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Target>	Target for the test pattern.	Numeric target ID
<Argument2>	Pattern selector.	Numeric
<Argument3>	Pattern selector.	Numeric

System, status and hardware queries

RCM - Recall Continued Message

Collects the remainder of a reply that was too long to send in one message. When a query returns result code 7, send RCM with the continuation ID from that reply to receive the next part, and repeat until the reply is complete.

Syntax

RCM <ContinuationID>

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<ContinuationID>	Sets the continuation returned with result code 7. Send 0 to collect the next part of the message.	Numeric; 0 collects the next part

RCS - Request Connection Status

Returns the connector type and connection status of the input used by a layer.

The connection status in the reply is always 2 (Unknown) in 6.1.3: the command does not poll the connector.

Syntax

RCS <LayerID>

Response

<Result Code> <Layer ID> <Current Connector Type> <Connection Status>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
----------	-------------	--------

<LayerID>	Sets the layer whose input is checked.	Numeric internal layer index, starting at 2
-----------	--	---

Returned values

Response	Values
Connection status	0 = Disconnected 1 = Connected 2 = Unknown
Current connector type	0 = HDMI 1 = Display Port 2 = SDI

RFI - Request Frame Info

Lists the frames in the system. For each one the reply gives its ID, model, whether it is enabled, its input, output and layer counts, its IP address and its software version. Use it to confirm the shape of a multi-frame system before addressing it.

Syntax

RFI

Response

<Result Code> For each node: <ID> <GUID> <Model> <Enabled> <Inputs> <Outputs> <Layers> <IPAddress>
<FirmwareVersion>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

RHI - Request Hardware Information

Returns identifying information for the connected frame, including its MAC address and model name.

Syntax

RHI

Response

<Result Code> <GUID> <MACAddress> <ModelName> <CapabilityFlags>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

RIF - Request Image File

Allows an image file to be returned from the internal still image folder on the Spyder system. Image files are always normalized to a .PNG format, regardless of their original format. The dimensions of the image may additionally be downscaled to a specified maximum width or height.

Notes:

- The byte formatting returned from this command is an ASCII encoding of hexadecimal numbers. For example, the following four bytes '0x00 0xAB 0xAC 0xFF' would be returned as the string literal '00ABACFF'.
- This command should be used sparingly, and it is recommended that the client device cache images locally to avoid regular re-downloading when possible. Heavy use of this command, particularly when image scaling is in use, may have a performance effect on the Spyder server.

Syntax

```
RIF <Image File Name (No Path)> [Max Width or Height]
```

Response

<ASCII encoded hexadecimal characters>

Arguments

Argument	Description	Values
<FileName>	File name of the image being requested. Do not include file path.	String filename
[Maximum width or height]	Optionally, specify maximum width or height of the image. Image scaling will maintain aspect ratio. Useful for grabbing thumbnails.	Numeric value

RPS - Request I/O Processor Status

Queries the current state of the I/O processor.

Asynchronous operations, such as still loading, are processed serially as background tasks in Spyder. You can query the current state of the I/O processor to use in external state logic such as needing to wait for the image load commands to complete.

The status returned includes a numeric value from 0 to 101 giving the percent progress as well as an ASCII string with a generic description of the task being processed. When idle, this request returns a progress/message response of 0/<empty string> or 101/Ready.

Syntax

```
RPS [<Chars>]
```

Response

```
<Result Code> <Progress> <Status Message>
```

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Chars>	Sets the number of characters to truncate names to.	—

RSF - Request System FrameRate

Allows the system to be interrogated for its current frame rate setting. The number returned from this command is the Frame Rate that is used in Frame Setup of the Configuration Manager.

Syntax

RSF

Response

<System Frame Rate ID>

Frame Rate	ID
FR_60	0
NTSC	1
PAL	2
FR_48	3
FR_30	4
FR_29_97	5
FR_25	6
FR_24	7
FR_23_98	8

Arguments

None.

SAV - Force Server Save

Forces the remote Spyder frame to save all configuration and user data changes to non-volatile storage.

Syntax

SAV

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
SAV	Forces the remote frame to save all configuration and user changes to non-volatile storage.	—

SDN - Restart Spyder Server

Restarts the Spyder server, its system manager, or the Frame. Give the restart mode, and optionally 0 to restart without saving first. The server is unavailable while it restarts.

Syntax

SDN <Mode> [0]

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Mode>	Selects what is restarted.	1 = Restart the Spyder server 2 = Restart the system manager in place, without saving 3 = Reboot the Frame (embedded systems only; otherwise result code 4)
[0]	Optional. Restart without saving first. Applies to modes 1 and 3; mode 2 never saves.	0

Multi-Viewer commands (capability-gated)

MVAC - Multi-Viewer Assign Content

Assigns an input to a tile of an OpMon output. Give the output, the tile index as reported by MVQO, and the input to place there. The tile then shows whatever source is currently routed to that input. The input must belong to the same frame as the output; an input on another frame is rejected. Tiles that carry a PixelSpace view are set from the OpMon Configuration tab rather than by command.

Syntax

```
MVAC <OutputID> <TileIndex> <InputID>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the OpMon output holding the tile.	Numeric output ID
<TileIndex>	Sets the tile to assign the input to. Use MVQO to list the tiles an output holds.	Numeric tile index
<InputID>	Sets the input to show in the tile. The input must be on the same frame as the output.	Numeric input ID

MVAS - Multi-Viewer Add Single

Adds a single tile to an OpMon output. Seven arguments are required, all numeric. Four more may be added to show a cropped section of the content instead of the whole of it. The view type selects what the tile shows, and the third argument is read according to that choice. Inputs and outputs shown in a tile must be on the same frame as the output the tile is added to. The tile is also checked against the tiles the output already carries, so a call can be refused on an output that is already full. Use MVQO first to see the tiles an output holds.

Syntax

```
MVAS <OutputID> <ViewType> <ContentID> <Left> <Top> <Width> <Height> [<SectionLeft>
<SectionTop> <SectionWidth> <SectionHeight>]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the OpMon output the tile is added to.	Numeric output ID
<ViewType>	Selects what the tile shows.	0 = Program PixelSpace 1 = Preview PixelSpace 2 = Input 3 = Output
<ContentID>	Identifies the content to show. Read as a PixelSpace ID, an input ID or an output ID, according to the view type.	PixelSpace ID when the view type is 0 or 1; -1 binds no PixelSpace Input ID when the view type is 2 Output ID when the view type is 3
<Left>	Sets the horizontal position of the tile, in pixels.	Numeric
<Top>	Sets the vertical position of the tile, in pixels.	Numeric
<Width>	Sets the width of the tile, in pixels.	Numeric
<Height>	Sets the height of the tile, in pixels.	Numeric
<SectionLeft> <SectionTop> <SectionWidth> <SectionHeight>	Optional. Shows only the given section of the content in the tile. Omit all four to show the whole of it.	Numeric; the section must fall within the frame's display area

MVCA - Multi-Viewer Clear All

Clears every tile from a multi-tile output, leaving it empty. Use MVQO to confirm the result. Recalling a preset with MVPR restores a tile layout.

Syntax

MVCA <OutputID>

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the Multi-Viewer output.	Numeric output ID

MVKF - Multi-Viewer Set Keyframe Properties

Sets a keyframe property on a single tile of a multi-tile output. Give the output, the tile index as reported by MVQO, and the value to apply. Read the result back with MVQO.

Syntax

```
MVKF <OutputID> <TileIndex> <Left> <Top> <Width> <Height> <BorderThickness> [<Red>  
<Green> <Blue>]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the OpMon output holding the tile.	Numeric output ID
<TileIndex>	Sets the tile to adjust.	Numeric tile index
<Left>	Sets the horizontal position of the tile, in pixels.	Numeric
<Top>	Sets the vertical position of the tile, in pixels.	Numeric
<Width>	Sets the width of the tile, in pixels.	Numeric
<Height>	Sets the height of the tile, in pixels.	Numeric
<BorderThickness>	Sets the thickness of the tile border, in pixels.	Numeric
<Red> <Green> <Blue>	Optional. Sets the colour of the tile border.	0 to 255 for each component

MVPL - Multi-Viewer Preset Learn

Saves the current tile layout of a multi-tile output as a preset.

Syntax

```
MVPL <OutputID> <PresetName>
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
OutputID	Sets the ID of the output.	3 = output 4 7 = output 8 11 = output 12 15 = output 16
<PresetName>	The name of the desired preset to learn.	A unique preset name.

MVPR - Multi-Viewer Preset Recall

Recalls a tile layout preset onto a multi-tile output: the OpMon or SourceMon layout, or a custom preset recalled by name.

Syntax

```
MVPR <OutputID> <PresetType> [<PresetName>]
```

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the ID of the output.	3 = output 4 7 = output 8 11 = output 12 15 = output 16
<PresetType>	Type of Multiviewer preset.	0 = OpMon 1 = SourceMon 2 = Custom
<PresetName>	Name of the custom preset to recall. Required when <PresetType> is 2 (Custom).	—

MVQO - Multi-Viewer Query Output

Returns the tile map of an OpMon output. The reply lists each tile in turn: index, type, the input assigned to it, then the tile's X, Y, width and height, its border width, and its border colour as red, green and blue. The value reported for a tile's content is an input identifier, not the source name shown on screen. Use MVQO to discover tile indexes before assigning content, and to confirm the result.

Syntax

```
MVQO <OutputID>
```

Response

<Result Code> Multi-Viewer configuration data for the output.

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the Multi-Viewer output.	Numeric output ID

MVST - Multi-Viewer Titling

Sets how a tile on an OpMon output is titled. Give the output, the tile index as reported by MVQO, and the titling mode as a number. The mode is a number, not the text of a caption.

Syntax

```
MVST <OutputID> <TileIndex> <TitleMode>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the OpMon output holding the tile.	Numeric output ID
<TileIndex>	Sets the tile to adjust.	Numeric tile index
<TitleMode>	Sets how the tile is labelled. This is a mode number, not caption text.	Numeric mode

OpMon and tiled output commands (capability-gated)

ATO - Apply Theme to Output Tile

Applies a display theme to the tiles of an OpMon output. Give the output, OM for an OpMon output, and the theme to apply. Use -1 to apply no theme.

Syntax

```
ATO <OutputID> <MV|OM> <ThemeID>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	Sets the output to apply the theme to.	Numeric output ID
<MV OM>	Selects which surface the theme applies to.	MV = Multi-Viewer OM = OpMon
<ThemeID>	Sets the on-screen display theme to apply.	Numeric theme ID; -1 applies no theme

OMSD - OpMon Set Dimensions

Sets the tile layout of an OpMon output. Give the output, the tile row, then the position and size of the tile rectangle, then the sources to place in it. The sources argument is required whenever a row other than 0 is given. Use MVQO to read the resulting tile map.

Syntax

```
OMSD <OutputID> <Row> <X> <Y> <Width> <Height> <SourceIDs>
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputID>	ID of the OpMon output.	Numeric output ID
<Arguments>	Dimension parameters.	Numeric

OMSL - OpMon Set Large Tile

Sets the source shown on the large tiles of an OpMon output. Give the output, the tile start index, then the source to place.

Syntax

```
OMSL <OutputID> <StartIndex> <SourceID>
```

Response

<Property Name> <Property Type> <Current Value>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Output ID>	Determines the output config object ID.	–0 – 15
<StartIndex>	The position to start the SourceIDs	0, 1, etc.
<SourceIDs>	The numerical ID of a given source. Note: A given source's input number is one minus it's assigned location, e.g. Input 1 is InputId 0 and Input 8 is InputId 7.	0, 1, etc.

OMSS - OpMon Set Small Tile

Sets the source shown on the small tiles of an OpMon output. Give the output first, then the tile start index, then the sources to place.

Syntax

```
OMSS <OutputId> <StartIndex> <SourceIds>
```

Response

<Property Name> <Property Type> <Current Value>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Output ID>	Determines the output config object ID.	–0 – 15
<StartIndex>	The position to start the SourceIDs	0, 1, etc.
<SourceIDs>	The numerical ID of a given source.	0, 1, etc.

	Note: A given source’s input number is one minus it’s assigned location, e.g. Input 1 is InputId 0 and Input 8 is InputId 7.	
--	--	--

SRAG - Source Assign Group Tile

Syntax

SRAG <OutputId> <TileSize> <TileIndex> <SourceIdList> <Duration>

Response

<Result Code>

Refer to **Command processor responses, page 10**, for additional information on Result Codes.

Arguments

Argument	Description	Values
<OutputId>	The numerical ID of a given output. Note: A given output's number is one minus it's assigned location, e.g. Output 1 is OutputId0 and Output 8 is OutputId.	0, 1, etc.
<TileSize>	The division of the tiles on the screen. Full is one, half is four, quarter is 16.	F: Full H: Half Q: Quarter
<TileIndex>	The indexing is based off the number of tiles.	0-15
<SourceIdList>	The numerical ID of a given source. Note: A given source's input number is one minus it's assigned location, e.g. Input 1 is InputId 0 and Input 8 is InputId 7.	0, 1, etc.
<Duration>	Transition measured in frames.	D24, D30, D60, etc.

SRAT - Source Apply to Tile

Places a source onto a tile of a tiled output. Give the source, then the tile group size as a single letter — Q for quarter, H for half, F for full or C for custom — then the output and the tile position within that group. Add D followed by a frame count to transition the change rather than cut to it. The output must be configured for tiled operation.

Syntax

```
SRAT <SourceID> <Q|H|F|C> <OutputID> <GroupIndex> [D<Frames>]
```

Response

<Result Code>

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<SourceID>	Sets the source to route to the tile group.	Numeric source ID
<Q H F C>	Selects the tile group to route the source to.	Q = Quarters H = Halves F = Full C = Custom
<OutputID>	Sets the output carrying the tile group.	Numeric output ID
<GroupIndex>	Sets the tile within the group.	Numeric index
[D<Frames>]	Optional. Delays the change by the given number of frames.	D followed by a frame count, for example D30

Hardware data dumps

RHWC - Request Hardware Config Data Dump

Requests a dump of hardware configuration data. Select which buffer to report with a single letter: F, H or B. The data is returned over the system's monitoring channel rather than in the command reply.

Syntax

RHWC <F|H|B>

Response

<Result Code> None on the command channel.

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<F H B>	Selects the buffer to dump.	F = Full H = Half B = Both

RHWS - Request Hardware Status Data Dump

Requests a dump of hardware status data. The data is returned over the system's monitoring channel rather than in the command reply.

Syntax

RHWS <Arguments>

Response

<Result Code> None on the command channel.

Refer to Command processor responses, page 10, for additional information on Result Codes.

Arguments

Argument	Description	Values
<Arguments>	Dump selector.	Numeric

Commands not supported in 6.1.3

The commands listed here appear in earlier Spyder documentation and in controllers written for legacy Spyder systems. None of them is supported for Spyder-S operation, but they do not all behave the same way. Two of them, OCS and OCU, are no longer present and return result code 2, Header — an invalid command was specified. Most of the rest are still present and accept a command rather than rejecting it, so a controller can appear to succeed even though the underlying feature is not available on Spyder-S. The three Basic Preset commands need particular care and are covered in the notice below. They are all listed so that a controller carried over from an earlier system can be identified and updated.

Notice: Basic Presets was removed from Spyder Studio in 6.1.3 and is not available on Spyder-S. The three tokens below are documented only so that a controller written for an earlier Spyder system can be checked for them and corrected; they are not commands to build against. Unlike the inert tokens in this section they were never taken out of the command processor, so a controller that still sends BPR will change what is on air, and one that still sends BPL will silently store state — with nothing shown in Spyder Studio to explain either. Nothing stored this way can be removed: Basic Presets are held outside the register types, so RRL and RRD cannot reach them, and DCK reports success without deleting anything. Audit any controller carried over from an earlier system and remove these three commands from it before going on air.

Command	Name	Status on Spyder-S 6.1.3
BPL	Basic Preset Learn	Removed in 6.1.3 — not for use. The Basic Presets interface was removed from Spyder Studio and the feature is not available on Spyder-S. The token was left in the command processor and still responds, so a legacy controller can store state unintentionally. Stored presets cannot be deleted.
BPR	Basic Preset Recall	Removed in 6.1.3 — not for use. The Basic Presets interface was removed from Spyder Studio and the feature is not available on Spyder-S. The token was left in the command processor and still responds: a legacy controller sending it will change what is on air, with no indication in Spyder Studio.
RBL	Request Basic Preset List	Removed in 6.1.3 — not for use. The Basic Presets interface was removed from Spyder Studio and the feature is not available on Spyder-S. The token was left in the command processor and still responds.
CII	Capture Input Image	Image capture is not implemented on Spyder-S (E4K-BBC) hardware.
CLI	Capture Layer Image	Image capture is not implemented on Spyder-S (E4K-BBC) hardware.
COI	Capture Output Image	Image capture is not implemented on Spyder-S (E4K-BBC) hardware.
ESSC	Execute Special System Command	Not implemented on Spyder-S.
ICK	Input Color Key	Accepted by the command processor, but the Spyder-S layer provides cut (linear) keying only; colour keying is not available.
ILK	Input Luminance Key	Accepted by the command processor, but the

		Spyder-S layer provides cut (linear) keying only; luminance keying is not available.
OCB	Output Config Blend	Edge blending is not reachable through the external control API.
OCR	Output Config Rotation	Output rotation is not implemented on Spyder-S.
OCS	Output Configuration Save	Not present in 6.1.3; returns result code 2. Retired — the output configuration command group was reworked.
OCU	Output Config Undo/Cancel	Not present in 6.1.3; returns result code 2. Retired — the output configuration command group was reworked.

Note: some keyer settings remain visible in an input's property list even though the keying commands are not supported. Those settings have no effect on Spyder-S.