HiSLIP-SCPI Server

Aaronia RTSA Suite PRO

In/Out Block

Version: 2022.2

General

The HiSLIP-SCPI Server block provides a TCP/IP server interface with an HiSLIP protocol on top if enabled. The block provides a SCPI command set to control a RTSA Suite Mission remotely.

HiSLIP Protocol

The HiSLIP-SCPI Server block of the RTSA Suite PRO provides an HiSLIP interface running in Overlapped Mode, protocol version 1.0. The HiSLIP protocol specification can be found at the IVI Foundation website.

Here are a few particularities in this HiSLIP implementation:

- The interface returns the Vendor ID “AA” during the connection establishment.
- The message type ***AsyncRemoteLocalControl*** will get a reply but is not implemented. This might be added with future releases.
- When the interface is locked by one or multiple users the synchronous messages of other users are buffered in a software buffer with up to 50 HiSLIP messages. If there is a buffer overflow a custom Fatal Error **128, "Locked Rx queue overflow"** is raised. The IVI interface specification requires the messages to be kept in the hardware buffer, thus the client would get a send timeout instead of the Fatal Error**.** This simplifies our SW and HW independent implementation, if is really required to have this behaviour feel free to let us know within some details.

The HiSLIP protocol transports the SCPI protocol data with the ***Data*** and ***DataEnd*** message types. Additionally, following HiSLIP messages are also directly assosiated with the SCPI interface:

- ***AsyncStatusResponse***: The included Status Byte is the Status Byte Register of the SCPI interface. The MAV bit is set by the HiSLIP interface if the last send message ID does not match the ID received with previous ***AsyncStatusQuery*** message.
- ***Trigger***: This message will call the SCPI command “*TRG”
- ***AsyncServiceRequest:*** This message is triggered by the SCPI interface depending on the Status Enable Registers of the SCPI interface

SCPI Protocol

The SCPI interface is based on the specification 1999 SCPI Syntax & Style also available at the IVI Foundation website.

The HiSLIP protocol can be disabled in the HiSLIP-SCPI Server block configuration and the SCPI interface can be used solely with a raw TCP IP connection and ASCII commands, for example with a terminal program like Putty.

Status Register

Following is the description of the SCPI status registers. Not all flags defined in the SCPI Standard are making sense controlling RTSA Suite PRO blocks which are not implemented and left out here.

Status Byte:

Value	Name	Description
0x04	Error Queue	This bit will be set if the error queue was empty and an error is added to the error queue
0x08	Questionable Status	This bit is set to true if any bit of the Questionable Status Register is set
0x10	MAV	This bit is not set by the SCPI interface. The HiSLIP interface in Overlapped Mode might set this bit in the Async Status Response
0x20	Event Status Register	This bit is set to true if any bit of the Event Status Register is set
0x40	Request for Service	This bit is set if any bit is set which has triggered a service request
0x08	Operational Status	This bit is set to true if any bit of the Operational Status is set

Event Status Register:

Value	Name	Description
0x01	Operation Complete	This bit is set if the command *OPC was called and all previuos started commands are finished
0x20	Command Error	This bit is set if any command processing has generated an error
0x80	Power On	always set to 1

Operational Status Register

Value	Name	Description
0x10	Measureing	This bit is set when data was requested and is available and send to the client. Therefore, this bit can be used to wait for data or a trigger after the *TRG or stream:data? command
0x20	Waiting for Trigger	This bit is set by the command *TRG or stream:data? command

Questionable Status Register

Not used yet.

Basic SCPI Commands

Command	Parameter	Return Value	Server Action
*IDN?	No parameter	String	Returns the Identification String. Example: "Aaronia AG, Aaronia RTSA Suite PRO HiSLIP-SCPI Server,2022.1"
*CLS	No parameter	No return value	This command will clear the Error Queue and all registers, except the enable registers. All running commands are cleared or chanceled.
*RST	No parameter	No return value	This command will do the same as the *CLS command and also resets all enable registers and stream configuration related parameters. The RTSA Suite PRO mission and block parameters are not touched here.
*OPC	No parameter	No return value	When all commands are completed the Operation Complete bit in the Event Status Register will be set.
*OPC?	No parameter	{ 1 }	This command will wait for all commands to be completed and will then answer with "1" as a reply.
*ESE	Integer {0 - 255}		Sets the Event Status Enable Register
*ESE?	{No parameter | MIN | MAX}	Integer {0 - 255}	Returns the Event Status Enable Register
*ESR?	{No parameter | MIN | MAX}	Integer {0 - 255}	Returns the Event Status Enable
*SRE	Integer {0 - 255}		Sets the Status Byte Enable Register
*SRE?	{No parameter | MIN | MAX}	Integer {0 - 255}	Returns the Status Byte Enable Register
*STB?	{No parameter | MIN | MAX}	Integer {0 - 255}	Returns the Status Byte Register
STATus:PREset	No parameter	No return value	Reset the Operational and Questionable Status and the related Enable Registers.
STATus:OPERation?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Read the Operational Status Register and resets all its bits
STATus:OPERation:ENABle	Integer {0 - 65535}	No return value	Sets the Operational Status Enable Register

STATus:OPERation:ENABle?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Returns the Operational Status Enable Register
STATus:OPERation:Condition?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Read the Operational Status Register
STATus:QUEStionable?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Read the Questionable Status Register and resets all its bits
STATus:QUEStionable:ENABle	Integer {0 - 65535}	No return value	Sets the Questionable Status Enable Register
STATus:QUEStionable:ENABle?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Returns the Questionable Status Enable Register
STATus:QUEStionable:Condition?	{No parameter | MIN | MAX}	Integer {0 - 65535}	Read the Questionable Status Register
*TST?	No parameter	No return value	not implemented
*WAIt	{ No parameter | Integer {0-65535}	No return value	This command will wait for all previous commands to be completed before following commands are executed. The Parameter will set the timeout time in milliseconds. Without parameter it will wait for a maximum of 10 seconds
*SLEep	Integer {0 - 65535}	No return value	This command will delay the execution of all following SCPI commands by the given time.
*TRG	No parameter	{JSON Header } + {binary data}	Same as the command **streaming:data?** (see below)
SYSTem:ERRor?	No parameter	Error code, “Error String“ For example: **0,"No error"**	This command will return the oldest error in the error queue. If read the error is removed from the error queue. The queue gets cleared with the device clear operation

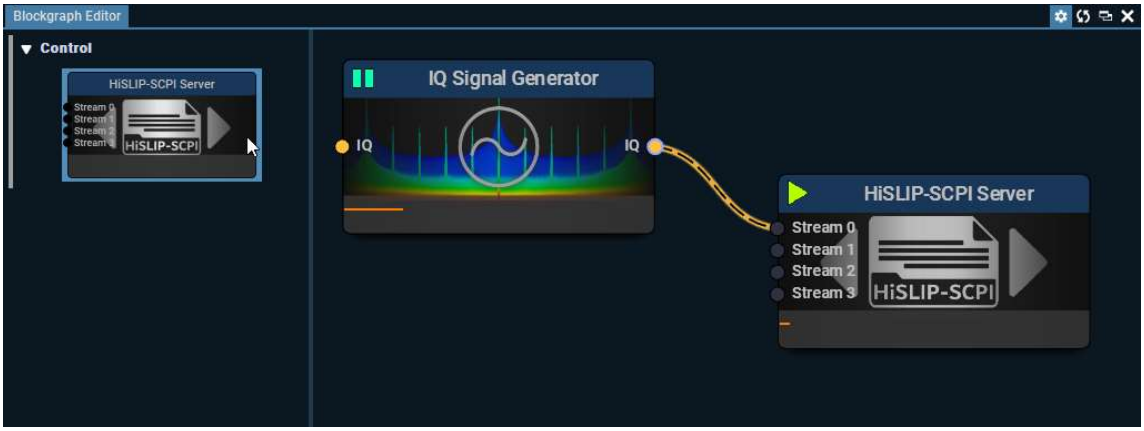
Data Streaming related SCPI Commands

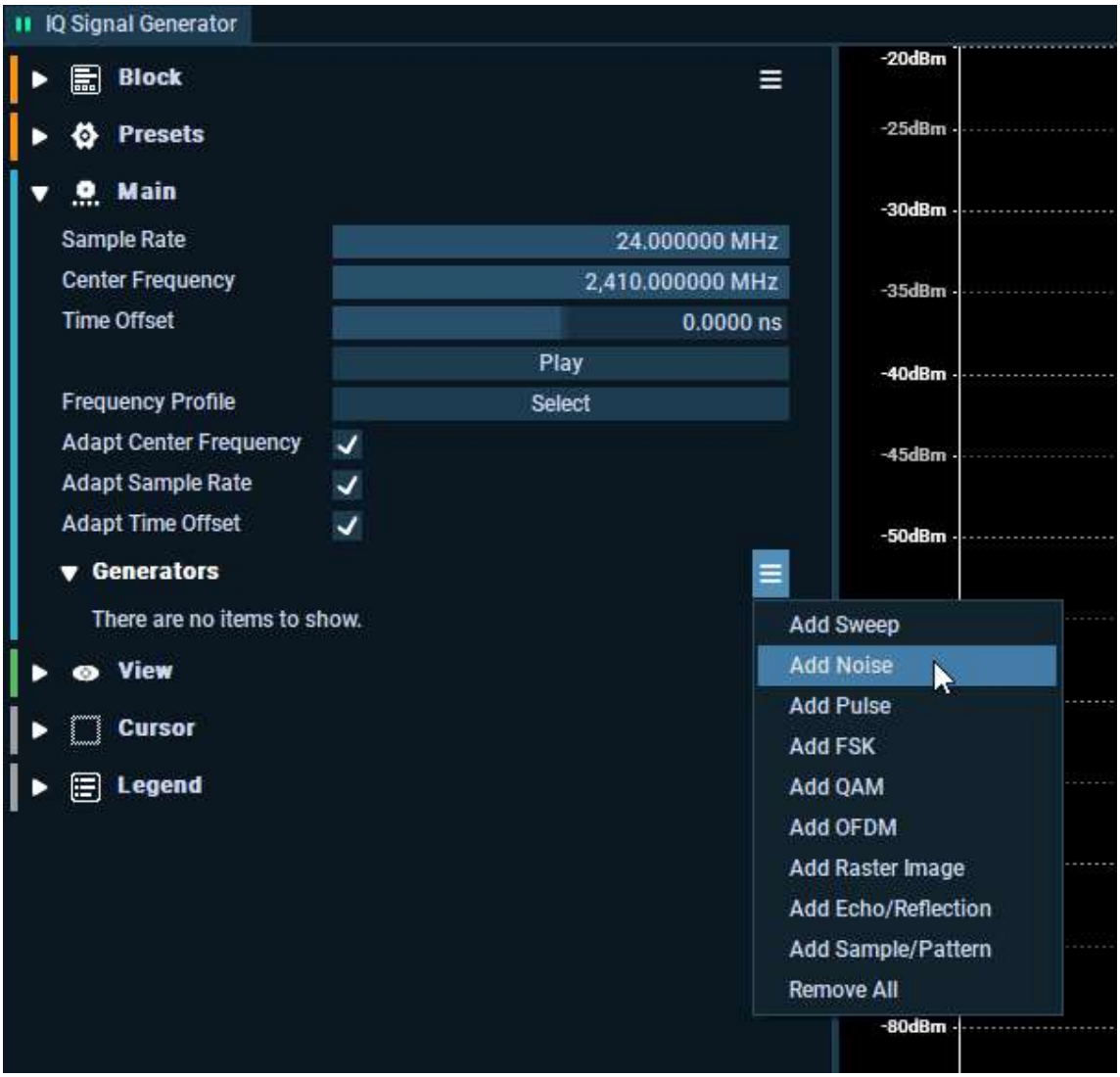
Command	Parameter	Return Value	Server Action
STREAMing:STArt	No parameter	No return value	This send a start streaming command to all Blocks connected to the HiSILIP[SCPI] block. (directly or indirectly)
STREAMing:STARTOPC	No parameter	No return value	This command will execute the streaming:start command and enables the use of the OPC synchronization. Some devices need some time to start a stream the first time.
STREAMing:STOp	No parameter	No return value	This send a stop streaming command to all Blocks connected the inputs of the HiSILIP-SCPI block.
STREAMing:INput	Integer { 0-3 } Default: 0	No return value	Input switch to select the active stream input
STREAMing:INput?	{No parameter | MIN | MAX}	Integer { 0 - 3 }	returns current selected stream input
STREAMing:COUnt	Sign. Integer {-1 - 65535} Default: 1	No return value	This will set the number of data packets received by the commands ***TRG** or **streaming:data?** The value "-1" will start an endless data stream until the command ***abort,*** ***clr** or ***rst** was called.
STREAMing:COUnt?	{No parameter | MIN | MAX}	Integer {-1 - 65535}	Returns the value of the **stream:count** configuration item
STREAMing:HEADer:ENABle	Boolean {OFF|0|ON|1} Default: ON	No return value	This will enable/disable the JSON Header in the data packets
STREAMing:HEADer:ENABle?	No parameter	Boolean {OFF|0|ON|1}	Returns if the data header is enabled
STREAMing:DATa?	No parameter	{JSON Header if enabled + #<digits size><size><binary data>}	Returns one or multiple data packets as configured with streaming:count. This command will also wait an unlimited time until data gets received. To synchronize with the SCPI server the Operational Status Flag Measuring is raised as soon as the first data packet is available.
CONFig?	No parameter	String	This will return the complete configuration tree of the current RTSA Suite Mission configuration
ABort	No parameter	No return value	This will abort the *TRG or Streaming:data? command

PRESet	No parameter	No return value	This will reset all RTSA Suite Blocks to the factory default configuration. It will execute for each block following SCPI command: **[blockname]:presets:resettopreset 1** Take care this will override the Mission configuration.

Mission and Block Configuration related SCPI Commands

The configuration tree returned by the command **config?** are depending on the RTSA block graph configuration. Here is an example mission for this chapter, an IQ Signal Generator block connected to the HiSLIP-SCPI Server block input. In the IQ Signal Generator block a Noise generator was added:





The response of the ***CONFIG?*** command contains a list of all configuration items a user can configure via the user interface in the RTSA Suite Mission. There are a lot of configuration items per block. The most important group should be the **[blockname]:main:** group. The **[blockname]:info** and **[blockname]:health** groups if supported are only available for blocks directly or indirectly connected to one of the HiSLIP-SCPI Server block inputs.

The RTSA Suite block configuration items are set asynchronous in the RTSA Suite blocks, therefore it may need some time to take effect. For all following commands synchronisation can be achieved by the ***wait** and ***OPC?** command or by the ***OPC** command and further Status Byte polling or waiting for the ***AsyncServiceRequest*** message of the HiSLIP interface.

Following is the ***CONFIG?*** response of this RTSA Suite Mission:

iqsignalgenerator_0:boverprotcfgwidget { OFF | 0 | ON | 1 }

iqsignalgenerator_0:boolshowconfigurationwidget { OFF | 0 | ON | 1 }

iqsignalgenerator_0:boolshowviewwidget { OFF | 0 | ON | 1 }

iqsignalgenerator_0:functionhandler:haspauseblock { OFF | 0 | ON | 1 }

iqsignalgenerator_0:functionhandler:mpauseblock { OFF | 0 | ON | 1 } | Descr.: Pause Streaming

iqsignalgenerator_0:functionhandler:hasexportblock { OFF | 0 | ON | 1 }

iqsignalgenerator_0:functionhandler:mexportblock | Descr.: Export

iqsignalgenerator_0:functionhandler:hasautosetblock { OFF | 0 | ON | 1 }

iqsignalgenerator_0:functionhandler:mautosetblock | Descr.: Autoset

iqsignalgenerator_0:settings:groupcolor

iqsignalgenerator_0:settings:title { "ASCII String" } | Descr.: Title

iqsignalgenerator_0:settings:desc { "ASCII String" } | Descr.: Description

iqsignalgenerator_0:settings:shortdesc { "ASCII String" } | Descr.: Short Description

iqsignalgenerator_0:settings:ribbonbarappearancestickOnMain | followDock } | Descr.: Ribbon Bar Appearance

iqsignalgenerator_0:settings:ribbonbarvisibilitysameAsDock | stayVisible } | Descr.: Ribbon Bar Visibility

iqsignalgenerator_0:presets:groupcolor

iqsignalgenerator_0:presets:resettopreset | Descr.: Preset

iqsignalgenerator_0:presets:resettouserdefaults | Descr.: Custom Presets

iqsignalgenerator_0:main:groupcolor

iqsignalgenerator_0:main:samplerate { 1000-20000000000 } | Descr.: Sample Rate

iqsignalgenerator_0:main:centerfreq { 0-20000000000 } | Descr.: Center Frequency

iqsignalgenerator_0:main:timeoffset { -0.2-0.2 } | Descr.: Time Offset

iqsignalgenerator_0:main:playbutton { OFF | 0 | ON | 1 } | Descr.: Play

iqsignalgenerator_0:main:frequencyprofile | Descr.: Frequency Profile

iqsignalgenerator_0:main:adaptcenter { OFF | 0 | ON | 1 } | Descr.: Adapt Center Frequency

iqsignalgenerator_0:main:adaptsamplerate { OFF | 0 | ON | 1 } | Descr.: Adapt Sample Rate

iqsignalgenerator_0:main:adapttimeoffset { OFF | 0 | ON | 1 } | Descr.: Adapt Time Offset

iqsignalgenerator_0:main:generators:generator#iqnoisegenerator#847585596:power { -200-0 } | Descr.: Power at 1Hz

iqsignalgenerator_0:view:groupcolor

iqsignalgenerator_0:view:background | Descr.: Background

iqsignalgenerator_0:view:powerunitdBm | dBW | dBmW | dBµW | dBV | dBmV | dBµV | dBA | dBmA | dBµA | Watt | Volt | Ampere } | Descr.: Unit

iqsignalgenerator_0:view:fontsize { 6-24 } | Descr.: Font Size

iqsignalgenerator_0:view:frequencystart { 0-20000000000 } | Descr.: Start Frequency

iqsignalgenerator_0:view:frequencyspan { 0-20000000000 } | Descr.: Span Frequency

iqsignalgenerator_0:view:gridmodenone | solid | light | dotted | cross | cross dot } | Descr.: Grid Mode

iqsignalgenerator_0:view:gridcolor | Descr.: Grid Color

iqsignalgenerator_0:view:axiscolor | Descr.: Axis Color

iqsignalgenerator_0:view:axiswidth { 1-3 } | Descr.: Axis Width

iqsignalgenerator_0:view:axisscale { 1-4 } | Descr.: Axis Scale

iqsignalgenerator_0:view:chartcolor | Descr.: Chart Area Color

iqsignalgenerator_0:view:absolutetime { OFF | 0 | ON | 1 } | Descr.: Absolute Time

iqsignalgenerator_0:view:axislabelsoutside | inside | box outside | box inside | box both } | Descr.: Axis Labels

iqsignalgenerator_0:view:title:placementhidden | top-left | top-center | top-right | left | center | right | bottom-left | bottom-center | bottom-right | custom } | Descr.: Placement

iqsignalgenerator_0:view:title:placementx { 0-1 } | Descr.: Placement X

iqsignalgenerator_0:view:title:placementy { 0-1 } | Descr.: Placement Y

iqsignalgenerator_0:view:title:bordersize { 0-8 } | Descr.: Border Size

iqsignalgenerator_0:view:title:bordercolor | Descr.: Border Color

iqsignalgenerator_0:view:title:backgroundcolor | Descr.: Background Color

iqsignalgenerator_0:view:title:title { "ASCII String" } | Descr.: Title

iqsignalgenerator_0:view:title:info { "ASCII String" } | Descr.: Information

iqsignalgenerator_0:view:title:titlescale { 1-10 } | Descr.: Title Scale

iqsignalgenerator_0:view:title:infoscale { 1-10 } | Descr.: Info Scale

iqsignalgenerator_0:view:title:titlecolor | Descr.: Title Color

iqsignalgenerator_0:view:title:infocolor | Descr.: Info Color

iqsignalgenerator_0:view:intraces:name { "ASCII String" } | Descr.: Name

iqsignalgenerator_0:view:intraces:visible { OFF | 0 | ON | 1 } | Descr.: Visible

iqsignalgenerator_0:view:intraces:color | Descr.: Color

iqsignalgenerator_0:view:intraces:linewidth { 0.2-6 } | Descr.: Line Width

iqsignalgenerator_0:view:intraces:opacity { 0-1 } | Descr.: Opacity

iqsignalgenerator_0:view:intraces:persistence { 0.001-5 } | Descr.: Persistence

iqsignalgenerator_0:view:intraces:limit { 1-2000 } | Descr.: Limit

iqsignalgenerator_0:cursor:mousemode | Descr.: Action

iqsignalgenerator_0:cursor:showoverlay2none | small | detailed } | Descr.: Information Overlay

iqsignalgenerator_0:cursor:axislabels { OFF | 0 | ON | 1 } | Descr.: Axis Labels

iqsignalgenerator_0:cursor:centered { OFF | 0 | ON | 1 } | Descr.: Center Cross

iqsignalgenerator_0:cursor:crosswidth { 1-4 } | Descr.: Line Width

iqsignalgenerator_0:cursor:borderwidth { 0-4 } | Descr.: Border Width

iqsignalgenerator_0:cursor:outsideblend { 0-1 } | Descr.: Outside Opacity

iqsignalgenerator_0:cursor:backcolor | Descr.: Line Background

iqsignalgenerator_0:cursor:frontcolor | Descr.: Line Color

iqsignalgenerator_0:cursor:bordercolor | Descr.: Border Color

iqsignalgenerator_0:cursor:outsidecolor | Descr.: Outside Color

iqsignalgenerator_0:cursor:axistextcolor | Descr.: Axis Text Color

iqsignalgenerator_0:cursor:axisbackcolor | Descr.: Axis Back Color

iqsignalgenerator_0:legend:placementhidden | top-left | top-center | top-right | left | center | right | bottom-left | bottom-center | bottom-right | custom } | Descr.: Placement

iqsignalgenerator_0:legend:placementx { 0-1 } | Descr.: Placement X

iqsignalgenerator_0:legend:placementy { 0-1 } | Descr.: Placement Y

iqsignalgenerator_0:legend:bordersize { 0-8 } | Descr.: Border Size

iqsignalgenerator_0:legend:bordercolor | Descr.: Border Color

iqsignalgenerator_0:legend:backgroundcolor | Descr.: Background Color

iqsignalgenerator_0:legend:color | Descr.: Text Color

iqsignalgenerator_0:legend:scale { 1-2 } | Descr.: Scale

iqsignalgenerator_0:legend:showtitle { OFF | 0 | ON | 1 } | Descr.: Show Title

iqsignalgenerator_0:legend:showdate { OFF | 0 | ON | 1 } | Descr.: Show Date

iqsignalgenerator_0:legend:showtype { OFF | 0 | ON | 1 } | Descr.: Show Type

iqsignalgenerator_0:legend:showcenter { OFF | 0 | ON | 1 } | Descr.: Show Center

iqsignalgenerator_0:legend:showspan { OFF | 0 | ON | 1 } | Descr.: Show Span

iqsignalgenerator_0:legend:showrbw { OFF | 0 | ON | 1 } | Descr.: Show RBW

iqsignalgenerator_0:legend:showbins { OFF | 0 | ON | 1 } | Descr.: Show Bins

iqsignalgenerator_0:legend:showlocation { OFF | 0 | ON | 1 } | Descr.: Show Geolocation

iqsignalgenerator_0:legend:showantenna { OFF | 0 | ON | 1 } | Descr.: Show Antenna

iqsignalgenerator_0:legend:showrate { OFF | 0 | ON | 1 } | Descr.: Show Rate

iqsignalgenerator_0:legend:showfftwindow { OFF | 0 | ON | 1 } | Descr.: Show FFTs

iqsignalgenerator_0:legend:showlatency { OFF | 0 | ON | 1 } | Descr.: Show Latency

iqsignalgenerator_0:legend:showpoi { OFF | 0 | ON | 1 } | Descr.: Show POI 100%

'

Data Request Example

Following command set will start the signal generator and get one IQ data packet:

Command	Server Action
***cls\n**	Device clear
iqsignalgenerator_0:main:centerfreq 2000000000\n	Set center frequency of the signal generator
***wait;**	Wait for the center frequency to be set
stream:start\n	Start the signal generator
stream:data?;	Wait for data to be forwarded, if available data is send
***wait;**	Pause processing of further commands until all requested data packets are send
stream:stop\n	Stop the signal generator

If there is something wrong processing or sending the data, the server will run into the maximum timeout for the ****wait*** command. If by the end of the timeout not all packets have been sent an error is raised and put into the error queue. The client can limit the timeout by a parameter added to the ****wait*** command. Here is a request for 100 data packets, followed by the ****wait*** command with a timeout of one second which should be more than enough:

```
*cls\n
stream:count 100; stream:start\n
stream:data?;*wait 1000;stream:stop\n
```

Data Packet Format

Here is one IQ data packet of a **stream:data?** reply:

{"startTime": 1645717251.136874,"depth": 1,"samples": 200,"payload": "iq","stepFrequency": 10000000,"unit": "volt","endTime": 1645717251.156874,"startFrequency": 2409995000,"maxValue": 2,"minValue": -2,"endFrequency": 2410005000,"size": 2}

#41600ÌÇt¸(³ªÀ(ÈÄ,(Âs(/àÕ(.{G)DFd)Í
¢)õð); @* 5¬*i5+Ý++}¹,T.Ñ¹e5´9æ1µ¹=Zµ9Ü,µ¹1Ö°¹×9×¾¹\K9u9e¹«¹99ýÂ9fÛ¹í$¹ã»9!>´¹^s¹ðû°¹§W9à&9 WÄ±9D<¹¤·9Å³½9µ¹u¤¹Ù¯¹±¹õ9

µ9:¯¹5´¹Wº½¹ÏY·¹uç9ü9K§º9ö#±9 Ìµ9rj¹£Ó½9T²9³j¯¹!b9©¹ÿ¶¹%Ì¹ã¹{X·¹Êø¹æ±¹XÚ¹&Ê´98´¹cb¹,º¹)92º¾9Mu¹ «µ¹+s·¹ß¹KJ·9RB9é;97Ü9bµ9¹ÅT¹G»9Nw9ì9Ñ96½¶¹2$µ¹SE9Nó¹|>9Ó_¹rq²¹½¹ÖJ·9 ·9h¿¾¹Â¦¹-_9Xµ9&«¹·»¹¯û¹µ9+µ9,±¹±ú¼9ë9¸¹=¾¹¢Ï¹¢9çV¹*v9JÃ´9Õñ¹«à9ãü9Û(¸¹R¿9²tµ¹C2¸9¢¸9ëµ¹£¸¹M´9d¾¹À9-9t9^ê9:W·9Ää´¹è¹)¹¹n¹87¹ 9¶ì99ó 9±¨®¹Î9;@¾¹IBµ¹Á9?µ9Þ3¸¹Ü¹ä¹ä¹õ*¾9XÌ9 \±9Ð¹£´¹Kê º¹áF¹½©º¹ì/¹Ì±¹Áå¹É º¹w`´9û¹þ¶¹c±¹it º¹´¹9dµ9*9¦¹Áh¹°9C¹·9ÏÎ9¡999 w9ôÎ9«·¹c6µ¹æò9·¹2¾¹¥¾¹·Ù°9Ph9Ò"·9Üu¹¹`¹S;¾¹= ·¹

First part if enabled is the JSON Header closed by a new line(\n) character. Followed by the SCPI binary data prefix **#< digits of size><size><binary data>**. In this example the prefix #41600 specify 4 digits for the size which results in the size of 1600 bytes for the binary data. The binary data is a list of 32bit floating point IEEE754 values in little endian.

The JSON Header should/may contain following fields:

size	Number	Size of a single sample (2 for IQ)
depth	Number	Depth of a single sample (e.g. bins in a histogram)
payload	String	Name of the payload type
unit	String	Name of the sample units
startTime	Number	Start time in seconds since the epoch
endTime	Number	End Time in seconds since the epoch
startFrequency	Number	Start of frequency range
endFrequency	Number	End of frequency range
stepFrequency	Number	IQ Sample Rate / Size of FFT Spectrum bin
minValue	Number	Minimum value in range
maxValue	Number	Maximum value in range
samples	Number	Number of Samples

To disable the data header, use this command:

stream:header:enable OFF\n

Take care on the network data throughput and the CPU/memory usage of the RTSA Suite PRO. High data rates might not be able to be send in real-time and are buffered by the software TCP/IP buffer in the RTSA Suite. Even if the measurement was already stopped data packets might still be in the TCP/IP buffers and sent to the client.