ipGateway:1.57

API Reference

X Platform, X5 HEVC SDI 1.0.2

23/04/2026

appear

TABLE OF CONTENTS

1 Overview

1.57

- **Added**
  - `AutoNegotiation` 1.0
- **Changed**
  - `physicalports` from 1.15 to 1.16;

1.56

- **Changed**
  - `streamType` from 1.0 to 1.1
  - `componentMapping` from 1.3 to 1.4
  - `output` from 1.38 to 1.39
  - `outputMapping` from 1.5 to 1.6
  - `mappingTemplates` from 1.2 to 1.3

1.55

- **Changed**
  - `input` from 1.38 to 1.39
  - `output` from 1.37 to 1.38
  - `srt` from 1.6 to 1.7
  - `status` from 1.24 to 1.25

1.54

- **Changed**
  - `input` from 1.37 to 1.38;

1.53

- **Changed**
  - `outputMapping` from 1.4 to 1.5;
  - `output` from 1.36 to 1.37;

1.52

- **Changed**
  - `physicalports` from 1.14 to 1.15;

1.51

- **Changed**
  - `input` from 1.36 to 1.37

1.50

- **Changed**
  - `ipCardSettings` from 1.1 to 1.2

1.49

- **Changed**
  - `ipCardSettings` from 1.0 to 1.1

1.48

- **Changed**
  - `input` from 1.35 to 1.36;

1.47

- **Changed**
  - `input` from 1.34 to 1.35
  - `output` from 1.35 to 1.36
  - `srt` from 1.5 to 1.6
  - `status` from 1.23 to 1.24

1.46

- **Changed**
  - `status` from 1.22 to 1.23;

1.45

- **Changed**
  - `input` from 1.33 to 1.34;

1.44

- **Changed**
  - `input` from 1.32 to 1.33;
  - `output` from 1.34 to 1.35;
  - `bissDescrambling` from 1.5 to 1.6;
  - `bissTypes` from 1.2 to 1.3;
  - `bissVersion1` from 1.2 to 1.3;
  - `bissVersion2` from 1.5 to 1.6;

1.43

- **Changed**
  - `input` from 1.31 to 1.32
  - `output` from 1.33 to 1.34
  - `srt` from 1.4 to 1.5
  - `srtValidate` from 1.0 to 1.1

1.42

- **Changed**
  - `linkModes` from 1.4 to 1.5
  - `physicalports` from 1.13 to 1.14
  - `ipTriggers` from 1.0 to 1.1

  - input from 1.30 to 1.31

1.41

- **Added**
  - linkModes 1.4
- **Changed**
  - physicalports from 1.12 to 1.13

1.40

- **Added**
  - srtCardSettings 1.0
- **Changed**
  - srt from 1.3 to 1.4
  - input from 1.29 to 1.30
  - output from 1.32 to 1.33

1.39

- **Changed**
  - status from 1.21 to 1.22
  - srt from 1.2 to 1.3
  - input from 1.28 to 1.29
  - output from 1.31 to 1.32

1.38

- **Changed**
  - input from 1.27 to 1.28

1.37

- **Changed**
  - status from 1.20 to 1.21

1.36

- **Changed**
  - input from 1.26 to 1.27

1.35

- **Changed**
  - input from 1.25 to 1.26

1.34

- **Changed**
  - tsstatus from 1.3 to 1.5
  - status from 1.19 to 1.20
  - outputMapping from 1.3 to 1.4

  - componentMapping from 1.2 to 1.3
  - genericDescriptor from 1.1 to 1.2
  - component from 1.4 to 1.5
  - mappingTemplates from 1.1 to 1.2

1.33

- **Changed**
  - ipInterface from 1.10 to 1.11

1.32

- **Changed**
  - input from 1.24 to 1.25
  - output from 1.29 to 1.30
  - outputMapping from 1.2 to 1.3
  - bissScrambling from 1.4 to 1.5
  - bissDescrambling from 1.4 to 1.5
  - bissVersion2 from 1.4 to 1.5

1.31

- **Changed**
  - srt from 1.1 to 1.2
  - output from 1.28 to 1.29
  - input from 1.23 to 1.24
  - physicalports from 1.11 to 1.12

1.30

- **Changed**
  - status from 1.18 to 1.19
  - input from 1.22 to 1.23

1.29

- **Changed**
  - status from 1.17 to 1.18
  - input from 1.20 to 1.22
  - output from 1.26 to 1.27
  - asiModule from 1.10 to 1.11
  - physicalports from 1.10 to 1.11;
  - outputMapping from 1.1 to 1.2

1.28

- **Changed**
  - inputs from 1.19 to 1.20

  - outputs from 1.25 to 1.26
  - status from 1.16 to 1.17
  - asiModule from 1.9 to 1.10

1.27

- **Added**
- ipTriggers version 1.0
- triggers version 1.0
- baseUuidNamespace version 1.0
- streamType version 1.0
- **Changed**
  - performRecording from 1.0 to 1.1;
  - input from 1.18 to 1.19;
  - output from 1.24 to 1.25;
  - status from 1.15 to 1.16;
  - tsstatus from 1.1 to 1.3;
  - asiModule from 1.8 to 1.9;
  - srt from 1.0 to 1.1;
  - zixi from 1.0 to 1.1;
  - componentMapping from 1.1 to 1.2
  - outputMapping from 1.0 to 1.1
  - component from 1.1 to 1.4
  - mappingTemplates from 1.0 to 1.1

1.26

- **Added**
  - bissCaSettings 1.0;
  - outputMapping 1.0;
- **Changed**
  - output from 1.22 to 1.24;
  - input from 1.17 to 1.18;
  - status from 1.14 to 1.15
  - bissVersion2 from 1.3 to 1.4;
  - bissScrambling from 1.3 to 1.4;
  - bissDescrambling from 1.3 to 1.4;
  - physicalports from 1.9 to 1.10;

1.25

- **Added**

  - bissData 1.2;
  - bissHelpers 1.0;
  - bissScrambling 1.3;
  - bissDescrambling 1.3;
  - bissTypes 1.2;
  - bissSettings 0.0;
  - bissValidate 1.0;
  - bissVersion1 1.2;
  - bissVersion2 1.3;
- **Changed**
  - output from 1.21 to 1.22
  - physicalports from 1.8 to 1.9
  - ipinterface from 1.8 to 1.9
  - input from 1.16 to 1.17
  - status from 1.13 to 1.14
  - asiModule from 1.6 to 1.7

1.24

- **Changed**
  - input from 1.14 to 1.16
  - output from 1.19 to 1.21
  - physicalports from 1.7 to 1.8
  - ipinterface from 1.7 to 1.8
  - status from 1.12 to 1.13
  - asiModule from 1.5 to 1.6
  - tsstatus from 1.0 to 1.1

1.23

- **Changed**
  - input from 1.13 to 1.14
  - output from 1.18 to 1.19
  - status from 1.11 to 1.12
  - asiModule from 1.4 to 1.5

2 asiModule (1.11)

2.1 Overview

Changelog:

1.11

- **Added**
  - Added protocol GetAsiInputStatusProtocol
  - Added protocol GetAsiOutputStatusProtocol
- **Changed**
  - Split asiModule into asiModule and asiStatus.
  - The field 'AsiBitrates' replaced the fields 'totalBitsPerSecond' and 'effBitsPerSecond' in AsiStatus.
  - The field 'AsiBitrates' replaced the fields 'totalBitsPerSecond' and 'effBitsPerSecond' in AsiOutputStatus.

1.10

- **Changed**
  - The imported version of `status` was bumped to 1.17

1.9

- **Changed**
  - The imported version of `status` was bumped to 1.16
  - The imported version of `tsstatus` was bumped to 1.3

1.8

- **Changed**
  - The imported version of `status` was bumped to 1.15

1.7

- **Changed**
  - The imported version of `status` was bumped to 1.14

1.6

- **Changed**
  - The imported version of `status` was bumped to 1.13.
  - The imported version of `tsstatus` was bumped to 1.1.

1.5

- **Changed**
  - The imported version of `status` was bumped to 1.12.

2.2 Command Reference

2.2.1 GetAsiPhysicalPorts

- message GetAsiPhysicalPorts.Request
- message GetAsiPhysicalPorts.Response

2.2.2 SetAsiPhysicalPorts

- message SetAsiPhysicalPorts.Request
- message SetAsiPhysicalPorts.Response

2.2.3 ClearCounters

- message ClearCounters.Request
- message ClearCounters.Response

2.3 Type Reference

2.3.1 AsiConfigByteMode

Config enumeration. The ASI byte mode affects the maximum bit rate. Spread mode limits the max bit rate to 72Mbit/s, whereas max bit rate for burst depends on packet size.

enum

ASI_BURST
ASI_SPREAD

2.3.2 AsiConfigPacketSize

Config enumeration. The ASI packet size affects the maximum bit rate when in ASI burst mode.

enum

ASI_PACKET_SIZE_188
ASI_PACKET_SIZE_204

2.3.3 AsiPhysicalPort

AsiPhysicalPort contains the configuration for the physical BNC port on the card.

struct

enabled	**bool** Whether or not the port should be enabled in the FPGA.
portNumber	**int** The port index (0-7) of the port.
label	**string** A label string to associate with this port.
direction	**PortDirection** The configured direction/mode of the port.

2.3.4 AsiTestGenerator

Type to hold the test signal generator information for ASI Input cards.

struct

`enabled`	**bool** When true, the input stream will be replaced with NULL packets of constant bitrate.
`autoMode`	**bool** When true, test generator will only operate upon signal lost.
`cbr`	**int** Constant bitrate of the generated signal, in units of (kbps). Allowed range is 0 - 213 mbps.

2.3.5 AsiTrafficPolicing

Type to hold the traffic policing information for ASI Input cards.

struct

`enabled`	**bool** When true, the input stream will be taken down if the input bitrate exceeds maxBitrate.
`maxBitrate`	**int** Max allowed bitrate for traffic policing, in units of (kbps). Allowed range is 0 - 213 mbps.

2.3.6 AsiTsrFlywheel

Type to hold the TSR flywheel information for ASI Output cards.

struct

`enabled`	**bool** When true, bitrate of the output stream is going to be preserved with NULL packets, in case the input source bitrate is not stable.
`continuous`	**bool** When true, the output stream bitrate will be maintaned continuously.
`duration`	**int** Shows how long the output stream bitrate should be maintained, in case continuous is set to false. In units of milliseconds, allowed range is 0 - 30 seconds.
`maxAllowedPktDrops`	**int** Max number of allowed packet drops before TSR Flywheel kicks in, in range [1, 15]

2.3.7 ClearCounters.Request

ClearCountersRequest

2.3.8 ClearCounters.Response

ClearCountersResponse

2.3.9 ClearCountersRequest

The Clear Counters request has no parameters.

empty **struct**

2.3.10 ClearCountersResponse

The Clear Counters response has no parameters.

empty **struct**

2.3.11 GetAsiPhysicalPorts.Request

The ASI Physical Ports GET request has no parameters.

empty **struct**

2.3.12 GetAsiPhysicalPorts.Response

The ASI Physical Ports GET response returns a map from each port to its corresponding AsiPhysicalPort object.

struct

`data`	**map** from **UUID** to **AsiPhysicalPort** A map from port UUID to the AsiPhysicalPort object

2.3.13 InputAsi

Input object for ASI, which works as a proxy for either seamless mode or single input.

variant

`single`	**InputAsiSettings**

2.3.14 InputAsiSettings

Settings object for ASI Input.

struct

`id`	**UUID** ID of this ASI Input node.
`portId`	**UUID** ID of the physical port which is the source of this Input.
`portNumber`	**int**
`testGenerator`	**AsiTestGenerator**
`trafficPolicing`	**AsiTrafficPolicing**
	int

bufferDelay	Buffer delay to align with other sources to tune the total latency, in units of (ms). Allowed range is 0-1500ms. Default 0ms.

2.3.15 InputAsiTransportSettings

Top level ASI input configuration type, which is used in the transportSettings level of the generic Input objects.

struct

input	**InputAsi**

2.3.16 OutputAsi

ASI output variant that selects either cloned or single output mode.

variant

single	**OutputAsiSettings** Configuration for the only port used by this output.

2.3.17 OutputAsiSettings

Output settings for ASI outputs.

struct

id	**UUID** ID of this ASI Output node.
portId	**UUID** ID of the physical port which is the destination of this Output.
portNumber	**int**
byteMode	**AsiConfigByteMode** Byte mode can be either burst or spread.
packetSize	**AsiConfigPacketSize** Packet size can be either 188 or 204 bytes.
tsrFlywheel	**AsiTsrFlywheel**

2.3.18 OutputAsiTransportSettings

Transport settings for an ASI output.

struct

output	**OutputAsi** Output settings for ASI, which can contain either single or cloned port.

2.3.19 PortDirection

ASI physical port directions can be configured to either input, output, passthrough (monitoring, input only) or cloned (output only).

enum

INPUT
OUTPUT
PASSTHROUGH
CLONED

2.3.20 SetAsiPhysicalPorts.Request

The ASI Physical Ports SET request requires a map from each port to its corresponding AsiPhysicalPort object.

struct

data	**map** from **UUID** to **AsiPhysicalPort** A map from port UUID to the AsiPhysicalPort object

2.3.21 SetAsiPhysicalPorts.Response

The ASI Physical Ports SET response has no parameters.

empty **struct**

3 asiValidate (1.0)

3.1 Type Reference

3.1.1 FloatRange

Float Range

struct

`min`	**float** Minimum value of the range
`max`	**float** Maximum value of the range

3.1.2 IntRange

Integer Range

struct

`min`	**int** Minimum value of the range
`max`	**int** Maximum value of the range

3.1.3 Result

Validation result

struct

`result`	**bool** True if config is valid and false otherwise.
`msg`	**optional string** An error message, which may be given if the result is invalid.

4 autoNegotiation (1.0)

4.1 Overview

Changelog:

1.0

- **Added**
- Initial version

4.2 Type Reference

4.2.1 AutoNegotiation

Auto-Negotiation settings for physical ports. Auto-negotiation is a feature that allows a port to communicate with the device/switch on the other end of the link to determine the optimal speed for the connection.

enum

OFF	Disable auto-negotiation on this physical port.
ON	Enable auto-negotiation on this physical port.

5 baseUuidNamespace (1.0)

6 bissCaSettings (1.0)

6.1 Type Reference

6.1.1 BissCaEcmInterval

BISS CA ECM Repetition Interval

Predefined set of possible ECM repetition interval values, in milliseconds.

Values lower than 100 ms are outside of the BISS CA specification. These values may still be used if the parameter `extendedOptions` is set to true in the `BissCaSettings` struct.

enum

`I6`	5.7 ms (Value outside BISS CA spec)
`I15`	14.7 ms (Value outside BISS CA spec)
`I50`	50 ms (Value outside BISS CA spec)
`I100`	100 ms
`I125`	125 ms
`I150`	150 ms
`I175`	175 ms
`I200`	200 ms
`I250`	250 ms
`I300`	300 ms

6.1.2 BissCaEntitlementFlags

BISS CA Entitlement Flags

The entitlement flags are parameters for how the entitled receivers shall behave. These values are mandatory part of the stream, and is inserted into the session data, which is the encrypted part of the EMM table.

struct

`preventDescrambledForward`	**bool** Prevent descrambling forward
`preventDecodedForward`	**bool** Prevent decode forward
`insertWatermark`	**bool** Insert watermark

6.1.3 BissCaSettings

BISS CA Specific Settings

struct

`cryptoPeriod`	**int** Crypto period (in seconds).
`emmBitrate`	**int** EMM bitrate (kbps).
	BissCaEcmInterval

`ecmInterval`	ECM repetition interval. Note that changes to the repetition interval will not take effect until the next crypto period (the next time the ECM content is updated).
`emmPreferredPid`	**int** EMM preferred PID value. If the PID value is not available, the first larger PID value available will be used instead. If no larger PID value is available, then the first smaller available value will be used.
`ecmPreferredPid`	**int** ECM preferred PID value. If the PID value is not available, the first larger PID value available will be used instead. If no larger PID value is available, then the first smaller available value will be used.
`entitlementFlags`	**BissCaEntitlementFlags** BISS CA entitlement flags. Controls how the entitled receivers shall behave.
`extendedOptions`	**bool** Allow for extended options. This means allowing violation of the BISS CA specification. If set to true, then the following requirements of the BISS CA specification are *not* enforced: - EMM repetition interval must be at least 200 ms - ECM repetition interval must be at least 100 ms Note that the requirement that EMM interval must be larger than the ECM interval is still enforced.

7 bissDescrambling (1.6)

7.1 Type Reference

7.1.1 DescramblerSettings

- Descrambler Settings
-
- Settings defining how to descramble the transport stream.
- By default, there is a single descrambling settings parameter, where these
- settings are applied to all services in the transport stream (TS).
-
-
-

struct

`enabled`	**bool** Descrambling on/off. •
`method`	**DescramblingMethod** Descrambling settings.

7.1.2 DescramblingMethod

DescramblingMethod

variant

`fixedKeyRaw`	**bissTypes.FixedKeyRawDescrambling** Fixed key decryption.
`biss1`	**bissVersion1.Descrambling** BISS version 1 decryption.
`biss2`	**bissVersion2.Descrambling** BISS version 2 decryption.

8 bissHelpers (1.0)

9 bissScrambling (1.6)

9.1 Type Reference

9.1.1 ScramblerSettings

- Scrambler Settings
-
- Settings defining how to scramble the transport stream.
- By default, there is a single scrambling settings parameter, where these
- settings are applied to all services in the transport stream (TS).
-
- Optionally, however, one or more extra settings may be configured, where the
- different settings are applied to different services in the TS. The different
- settings are referenced by integer indecies. These indecies are
- cross-references to the transport stream configuration. For example, a single
- service may refer to one of these indecies, meaning that the service will be
- scrambled with the corresponding scrambling settings.
-
- Requirements:
-
  - All extra scrambling setting *must* contain the same scrambling method as
- the main settings.
-
- If these requirements are not met, the configuration is not valid.
-
-

struct

`enabled`	**bool** Scrambling on/off •
`main`	**ScramblingMethod** The main (default) scrambling settings •
`extra`	**map** from **int** to **ScramblingMethod** Additional scrambling settings, each with its own index.

9.1.2 ScramblingMethod

ScramblingMethod

variant

`fixedKeyRaw`	**bissTypes.FixedKeyRawScrambling** Fixed key encryption
`biss1`	**bissVersion1.Scrambling** BISS version 1 encryption
`biss2`	**bissVersion2.Scrambling** BISS version 2 encryption

10 bissSettings (1.0)

10.1 Type Reference

10.1.1 AllowedExtraSettings

Allowed Num of Extra Scrambling Settings

enum

EXTRA_15_ECMS	15 extra settings
EXTRA_7_ECMS	7 extra settings
EXTRA_3_ECMS	3 extra settings

10.1.2 AllowedMode

Allowed (De)Scrambling modes

enum

ALL	CSA and AES algorithms allowed
AES_ONLY	Only AES allowed
CSA_ONLY	Only CSA allowed

10.1.3 BissModuleSettings

BISS Module Setup for a given product (not configurable)

struct

allowedMode	**AllowedMode** Algorithms supported by product
numExtraPerOutput	**AllowedExtraSettings** Number of extra settings allowed per output
maxNumOutputs	**int** Maximum number of outputs
totalEmmBitrateLimit	**optional int** Set an upper limit for the total EMM playout bitrate (sum of the bitrate all EMMs). Given in units of kbps.

11 bissTypes (1.3)

11.1 Type Reference

11.1.1 Algorithm

De/Scrambling Algorithm

The DVB-CSA algorithm uses a 64-bit key, while all the AES algorithms use a 128-bit key.

enum

DVB_CSA	DVB-CSA
AES_ATIS_IDSA	AES ATIS IDSA
AES_ECBT_ECB1	AES-ECBT/AES-ECB1 (Alticast)
AES_ECBL_ECB2	AES-ECBL/AES-ECB2 (Latens)
AES_CBCT_CBC1	AES CBCT CBC1
AES_DVB_CISSA	AES DVB-CISSA
AES_IRDETO_CBCT	AES CBCT Irdeto
AES_CBCT_IV	AES CBCT w/specified Initialization Vector

11.1.2 FixedKeyRawDescrambling

Descrambling Method: Fixed Key Raw

struct

algorithm	**Algorithm** Scrambling algorithm
key	**string** Scrambling key
iv	**optional string** Scrambling Initialization vector (only for AES_CBCT_IV)
serviceMappings	**list** of **FixedKeyServiceMapping** A list of service mappings.

11.1.3 FixedKeyRawScrambling

Scrambling Method: Fixed Key Raw

struct

algorithm	**Algorithm** Scrambling algorithm
key	**string** Scrambling key
iv	**optional string** Scrambling Initialization vector (only for AES_CBCT_IV)

11.1.4 FixedKeyServiceMapping

Fixed key descrambling for a list of services

struct

key	**string** Scrambling key
iv	**optional string** Scrambling Initialization vector (only for AES_CBCT_IV)
serviceIds	**list** of **int** A list of services that the key applies to.

12 bissValidate (1.0)

12.1 Type Reference

12.1.1 FloatRange

Float Range

struct

`min`	**float** Minimum value of the range
`max`	**float** Maximum value of the range

12.1.2 IntRange

Integer Range

struct

`min`	**int** Minimum value of the range
`max`	**int** Maximum value of the range

12.1.3 Result

Validation result

struct

`result`	**bool** True if config is valid and false otherwise.
`msg`	**optional string** An error message, which may be given if the result is invalid.

13 bissVersion1 (1.3)

13.1 Type Reference

13.1.1 CsaKeyExpansion

Method of CSA Key Expansion from 48 to 64 bit

enum

CHECKSUM	Session Word is expanded using checksum.
ZERO	Session Word is expanded using zeros.

13.1.2 Descrambling

BISS version 1 Descrambling

variant

mode1	**Mode1Descrambling** BISS version 1, Mode 1.
modeE	**ModeEDescrambling** BISS version 1, Mode E.

13.1.3 Mode1Descrambling

BISS version 1 Descrambling, Mode 1

struct

expansion	**CsaKeyExpansion** Expansion method.
sessionWord	**string** Session Word, 48-bit hex string.
serviceMappings	**list** of **Mode1ServiceMapping** A list of service mappings.

13.1.4 Mode1Scrambling

BISS version 1 Scrambling, Mode 1

struct

expansion	**CsaKeyExpansion** Expansion method.
sessionWord	**string** Session Word, 48-bit hex string.

13.1.5 Mode1ServiceMapping

BISS version 1 Descrambling, Mode 1 for a list of services

struct

`sessionWord`	**string** Session Word, 48-bit hex string.
`serviceIds`	**list** of **int** A list of services that the session word applies to.

13.1.6 ModeEDescrambling

BISS version 1 Descrambling, Mode E

struct

`activeId`	**UUID** Active session ID.
`expansion`	**CsaKeyExpansion** Expansion method.
`encSessionWord`	**UUID** Encrypted Session Word, 128-bit. Reference to BissData UUID.
`serviceMappings`	**list** of **ModeEServiceMapping** A list of service mappings.

13.1.7 ModeEScrambling

BISS version 1 Scrambling, Mode E

struct

`expansion`	**CsaKeyExpansion** Expansion method.
`sessionWord`	**UUID** Session Word, 128-bit. Reference to BissData UUID.

13.1.8 ModeEServiceMapping

BISS version 1 Descrambling, Mode E for a list of services

struct

`activeId`	**UUID** Active session ID. Reference to BissData UUID.
`encSessionWord`	**UUID** Encrypted Session Word, 128-bit. Reference to BissData UUID.
`serviceIds`	**list** of **int** A list of services that the session ID and encrypted session word applies to.

13.1.9 Scrambling

BISS version 1 Scrambling

variant

mode1	**Mode1Scrambling** BISS version 1, Mode 1.
modeE	**ModeEScrambling** BISS version 1, Mode E.

14 bissVersion2 (1.6)

14.1 Type Reference

14.1.1 Descrambling

BISS version 2 Decrambling

variant

mode1	**Mode1Descrambling** BISS version 2, Mode 1.
modeE	**ModeEDescrambling** BISS version 2, Mode E.
modeCA	**ModeCADescrambling** BISS version 2, Mode CA.

14.1.2 Mode1Descrambling

BISS version 2 Descrambling, Mode 1

struct

sessionWord	**string** Session Word, 128-bit hex string.
serviceMappings	**list** of **Mode1ServiceMapping** A list of service mappings.

14.1.3 Mode1Scrambling

BISS version 2 Scrambling, Mode 1

struct

sessionWord	**string** Session Word, 128-bit hex string.

14.1.4 Mode1ServiceMapping

BISS version 2 Descrambling, Mode 1 for a list of services

struct

sessionWord	**string** Session Word, 128-bit hex string.
serviceIds	**list** of **int** A list of services that the session word applies to.

14.1.5 ModeCADescrambling

BISS version 2 Decrambling, Mode CA

struct

key	**optional UUID** Private RSA key. If UUID value is given, this references a Biss-Data UUID. If optional is empty, the buried RSA private key is used.

14.1.6 ModeCAScrambling

BISS version 2 Scrambling, Mode CA

struct

receivers	**UUID** Public RSA keys of receivers. Reference to BissData CSV File UUID or BissDataGroup UUID.
esId	**string** Entitlement session ID (16-bit hex string).
settings	**bissCaSettings.BissCaSettings** ModeCa settings.

14.1.7 ModeEDescrambling

BISS version 2 Descrambling, Mode E

struct

activeId	**UUID** Active session ID. Reference to BissData UUID.
encSessionWord	**UUID** Encrypted Session Word, 128-bit. Reference to BissData UUID.
serviceMappings	**list** of **ModeEServiceMapping** A list of service mappings.

14.1.8 ModeEScrambling

BISS version 2 Scrambling, Mode E

struct

sessionWord	**UUID** Session Word, 128-bit. Reference to BissData UUID.

14.1.9 ModeEServiceMapping

BISS version 2 Descrambling, Mode E for a list of services

struct

activeId	**UUID** Active session ID. Reference to BissData UUID.
encSessionWord	**UUID** Encrypted Session Word, 128-bit. Reference to BissData UUID.

`serviceIds`	**list** of **int** A list of services that the session ID and encrypted session word applies to.

14.1.10 Scrambling

BISS version 2 Scrambling

variant

`mode1`	**Mode1Scrambling** BISS version 2, Mode 1.
`modeE`	**ModeEScrambling** BISS version 2, Mode E.
`modeCA`	**ModeCAScrambling** BISS version 2, Mode CA.

15 commonTypes (1.0)

15.1 Type Reference

15.1.1 NmosVersionNumber

Generic version number type for NMOS APIs

enum

V1_2
V1_3

16 component (1.5)

16.1 Overview

Changelog:

1.5

- **Added**
  - SCTE27_SUB to ComponentType enum and respective mappings.

16.2 Type Reference

16.2.1 AnyComponent

empty **struct**

16.2.2 ComponentGroup

```
ComponentGroup: Grouping of ComponentType, used for filtering
```

enum

UNDEFINED
AUDIO
VIDEO
SUBTITLE
DATA

16.2.3 ComponentLanguage

string

16.2.4 ComponentSelector

```
Component selector.

@param pid          Selects a component using a specific (source) PID.
@param language     Selects a component by a three character language string.
@param type         Selects a component by its type.
@param group        Selects a component by group membership.
@param any          Selects any component.
```

variant

pid	**Pid**
language	**ComponentLanguage**
componentType	**ComponentType**
componentGroup	**ComponentGroup**
any	**AnyComponent**

16.2.5 ComponentType

A ComponentType is typically identified by a stream type (see ISO 13818-1 Table 2-29), often distinguished with another header field such as a descriptor tag.

enum

MPEG_VIDEO
MPEG_AUDIO
TTX
DVBSUB
ECM
PCR
PRIVATE
AC_3
H264
VBI
AAC
VC_1
SCS_EMM
DPI_CUE
EAC3
AIT
HBBTV_CAROUSEL
DATA_CAROUSEL
H265
S302M
JPEG2000
S2038
DVBTTMLSUB
JPEG_XS_VIDEO
SCTE27_SUB

16.2.6 Pid

int

16.2.7 SubtitleFormat

enum

DVB
DVB_HARD_OF_HEARING
EBU
EBU_HARD_OF_HEARING

17 componentMapping (1.4)

17.1 Overview

Changelog:

1.4

- **Changed**
- `streamtype` bumped from 1.0 to 1.1

1.3

- **Changed**
- `component` bumped from 1.4 to 1.5

1.2

- **Changed**
- Moved the StreamType definitions to dedicated IDCL module, `streamtype`
- Now using the ComponentType definition found in IDCL module `component`, as the one found in componentMapping version 1.1 was redundant.

17.2 Type Reference

17.2.1 ComponentAction

enum

STOP
PASSTHROUGH
REMAP

17.2.2 ComponentGeneration

A pmt entry generation rule. Usually used to indicate that some external PID will Appear at the specified PID.

struct

`streamType`	**streamtype.StreamType** Streamtype as specified in the standards.
`pid`	**int** PID number to signal the component on.

17.2.3 ComponentLayoutMap

A collection of a priority list of component mapping rules, and one default action to handle unmatched components.

struct

`defaultAction`	**ComponentMapDefaultAction** Default action to perform on any unmatched component

layout	**list** of **ComponentMapping** Priority list of mapping rules - each rule matches exactly one source component

17.2.4 ComponentMapDefaultAction

enum

PASSTHROUGH
STOP

17.2.5 ComponentMapping

The ComponentMapping object describes a rule that will match one - and only one - incoming component, and perform an action on it. I.e, the selection of type MPEG_AUDIO with language "nor" with an action set to "STOP", will stop one "instance" of mpeg audio with the norwegian language. Subsequent components matching the same filter will fall through to the next lower priority rule matching them.

There are some extra considerations in regards to pids containing PCR. It is not allowed to STOP the PCR content. If such a rule is applied, the payload of the component will be stopped, but the PCR content will still be mapped out.

It is possible to remap only the PCR content. To do this, componentType must be specified as PCR. Any other rule matching the same pid will remap all the content, not just the PCR.

struct

componentType	**optional component.ComponentType** Component type selector.
language	**optional string** Language selector.
pid	**optional int** Pid selector.
action	**ComponentAction** What to do with the matched component.
targetPid	**optional int** Optional. Must be specified when action is REMAP.

18 constants (1.1)

19 dnsConfig (1.0)

19.1 Type Reference

19.1.1 DnsConfig

Configuration for DNS

struct

`nameResolutionEnabled`	**bool** Enable name resolution
`nameServers`	**list** of **string** List of name servers

20 firewallTypes (1.0)

20.1 Type Reference

20.1.1 IpAddress

string

20.1.2 PaginatedRequest

```
Instructs the server to return only a subset of the data found on the server

@param pageSize number of entries returned per request
@param pageNumber what page of the set to returned, determined by the slice [pageSize*pageNumber
```

struct

pageSize	**int**
pageNumber	**int**

20.1.3 PaginationInfo

struct

totalEntries	**int** the total number of entries on the server, before matching

20.1.4 SocketAddress

Tuple with IP address and port

struct

address	**IpAddress** IP address, either IPv4 or IPv6
port	**int**

20.1.5 UUID

string

21 gatewayValidate (1.0)

21.1 Type Reference

21.1.1 BigIntRange

struct

`min`	**bigint**
`max`	**bigint**

21.1.2 FloatRange

Float Range

struct

`min`	**float** Minimum value of the range
`max`	**float** Maximum value of the range

21.1.3 IntRange

Integer Range

struct

`min`	**int** Minimum value of the range
`max`	**int** Maximum value of the range

21.1.4 Result

Validation result

struct

`result`	**bool** True if config is valid and false otherwise.
`msg`	**optional string** An error message, which may be given if the result is invalid.

22 genericDescriptor (1.2)

22.1 Overview

Changelog:

1.2

- **Changed**
  - The imported `component` version was bumped to 1.5.

1.1

- **Changed**
  - The imported `component` version was bumped to 1.4.

22.2 Type Reference

22.2.1 DescriptorAction

```
Dictate what action is to be performed on the provided descriptor.
```

enum

ADD
REMOVE
CHANGE

22.2.2 GenericPmtDescriptorEntry

```
Defines a generic descriptor entry with a given action, tag and optional data.

@param action       What action is to be performed with this descriptor entry.
@param tag          The Descriptor tag.
@param component    Optional component (inner loop) this descriptor sould apply to.
@param data         Optional data to add to/replace the descriptor.
```

struct

action	DescriptorAction
tag	int
component	optional component.ComponentSelector
payload	optional string

23 group (1.0)

23.1 Command Reference

23.1.1 GetST2110Group

Get all ST2110 groups from the database.

- **message** GetST2110Group.Request
- **message** GetST2110Group.Response

23.1.2 SetST2110Group

Create or update ST2110 groups in the database.

- **message** SetST2110Group.Request
- **message** SetST2110Group.Response

23.1.3 DeleteST2110Group

Delete the ST2110 groups corresponding to the list of ids given.

- **message** DeleteST2110Group.Request
- **message** DeleteST2110Group.Response

23.2 Type Reference

23.2.1 DeleteST2110Group.Request

DeleteST2110GroupRequest

23.2.2 DeleteST2110Group.Response

empty **struct**

23.2.3 DeleteST2110GroupRequest

DeleteST2110GroupRequest is a request used in DeleteST2110Group.

struct

`ids`	**list** of **UUID** List of UUIDs corresponding to groups that should be deleted.

23.2.4 GetST2110Group.Request

empty **struct**

23.2.5 GetST2110Group.Response

GetST2110GroupResponse

23.2.6 GetST2110GroupResponse

GetST2110GroupResponse is a response used in GetST2110Group.

struct

`data`	**map** from **UUID** to **ST2110Group** A map where the keys are UUIDs and the value are corresponding ST2110 groups.

23.2.7 ST2110Element

ST2110Element information about each service in the group

struct

`id`	**UUID** Contains a UUID to the Input or Output flow containing an essence.
`essenceType`	**ST2110Essence** Type of stream, video / audio / ancillary.
`channel`	**optional int** Audio channel. If the stream is of the type Audio, what channel does it belong to.

23.2.8 ST2110Essence

ST2110Essence defines the type of the streams contained in the ST2110 signal.

enum

`VIDEO`	Video
`AUDIO`	Audio
`ANCILLARY`	Teletext / Metadata

23.2.9 ST2110Group

ST2110Group contains references to all individual inputs or outputs in a SMPTE2110 input or output, as well as metadata about each essence.

struct

`label`	**string** A custom description or other textual identifier for the group.
`elements`	**list** of **ST2110Element** A list containing all the elements in a group

23.2.10 SetST2110Group.Request

SetST2110GroupRequest

23.2.11 SetST2110Group.Response

empty **struct**

23.2.12 SetST2110GroupRequest

SetST2110GroupRequest is a request used in SetST2110Group.

struct

`data`	**map** from **UUID** to **ST2110Group** A map where the keys are UUIDs and the values are corresponding ST2110 groups.

24 input (1.39)

24.1 Overview

Changelog:

1.39

```
- **Changed**
  - The imported `srt` version was bumped to 1.7.
```

1.38

```
- **Changed**
  - Renamed `InputAnalyzeTr07` to `InputAnalyzeSpts`.
  - Renamed `validateInputAnalyzeTr07` to `validateInputAnalyzeSpts`.
  - SPTS (previously called TR-07) is no longer work in progress.
```

1.37

```
- **Changed**
  - TR07 declared as work in progress in api doc
  - The option `tr07Mode` of type `InputAnalyzeTr07` was added to the InputAnalyzeMode variant
  - First parameter of functions `validateInputAnalyzeMpeg`, `validateInputAnalyzeDvb`, `validat
  - Change the `validateDejitterSettings` to not return false if CBR Dejitter Mode is used with
- **Added**
  - New function `validateInputAnalyzeTr07`
  - New `analyzeMode` parameter to functions `validateDejitter` and `validateDejitterSettings`
  - InputAnalyzeMode based validation in function `validateDejitterSettings`
```

1.36

```
- **Changed**
  - Use constants module 1.1 for product name to differenciate between dirrent traffic policing
```

1.35

```
- **Changed**
    - The imported `srt` version was bumped to 1.6.
```

1.34

```
- **Changed**
  - Removed is100G parameter from getDefaultInputBufferSize function.
  - Changed seamless buffer value from 100 to 20 for 100G.
  - Changed default dejitter buffer value from 100 to 40 for 100G.
```

1.33

- **Changed**
  - The imported `bissDescrambling` version was bumped to 1.6.

1.32

- **Changed**
  - The imported `srt` version was bumped to 1.5.

1.31

```
- **Changed**
  - Added new is100G parameter to getDefaultInputBufferSize function.
  - Added new getDefaultDejitterBufferSize function.
  - 100G min, max values for Traffic Policer
```

1.30

```
- **Changed**
  - The imported `srt` version was bumped to 1.4.
```

1.29

```
- **Changed**
  - The imported `srt` version was bumped to 1.3.
```

1.28

```
- **Changed**
  - Added optimizeLatency to InputUdpSeamless
```

1.27

```
- **Changed**
  - Make descrambling optional under AdditionalFeatures
  - Add triggers to AdditionalFeatures
```

1.26

```
- **Changed**
  - The field 'profile' in 'DejitterRtp' is changed from an int to new enum type 'RtpDejitterPro
```

1.25

```
- **Changed**
  - The imported bissDescrambling version was bumped to 1.5
  - Function getMaxBitrateRange is changed to getIpTrafficPolicingBitrateValidRange with new max
- **Added**
  - Function validateIpTrafficPolicingBitrate [15.11.22]
```

1.24

```
- **Added**
  - Validation for the field 'inputBufferSize' in 'InputUdpTransportSettings'.
  - Validation for the field 'bufferSize' in 'Dejitter'.
  - Validation for the field 'port.destination' in 'InputUdpSettings'.

- **Changed**
  - The imported srt version was bumped to 1.2
  - The field 'enabled' in 'IpTrafficPolicing' is now checked before the field 'maxBitrate' is v
```

1.23

- **Added**
  - The field ‘rptExcessiveLossThreshold’ was added to ‘InputUdpTransportSettings’.
  - The field ‘alarmWhenScrambled’ was added to ‘InputUdpTransportSettings’.

1.22

- **Added**

- The field 'rtp' was added to 'InputUdpSettings'.

1.21

- **Changed**
  - The imported `asiModule` version was bumped to 1.11

1.20

- **Changed**
  - The imported `asiModule` version was bumped to 1.10

1.19

- **Changed**
  - The imported `asiModule` version was bumped to 1.9.
  - The imported `srt` version was bumped to 1.1.
  - The imported `zixi` version was bumped to 1.1.

1.18

- **Changed**
  - The imported `bissDescrambling` version was bumped to 1.4.
  - The imported `asiModule` version was bumped to 1.8

1.17

- **Added**
  - The optional field `additionalFeatures` was added to `Input`.

    The `AdditionalInputFeatures` type includes the field `descrambling`, which is used to configure descrambling on the input.
- **Changed**
  - The imported `asiModule` version was bumped to 1.7.
- **Changed**
  - The imported `asiModule` version was bumped to 1.7.

1.16

- **Changed**
  - The imported `asiModule` version was bumped to 1.6.

1.15

- **Removed**
  - The field `cbrBitRate` was removed from `DejitterCbr`.

1.14

- **Added**
  - The field `identical` was added to `InputUdpSeamless`.

    It is used to signal if the sources are expected to be real 2022-7.

24.2 Command Reference

24.2.1 GetInputs

Get a list of all configured inputs.

- **message GetInputs.Request**
- **message GetInputs.Response**

24.2.2 SetInputs

Update configuration for inputs. If an object appear in the request which is not already in the configuration database, it will be added. Otherwise it will be updated.

- **message SetInputs.Request**
- **message SetInputs.Response**

24.2.3 DeleteInputs

```
Delete inputs.
```

- **message DeleteInputs.Request**
- **message DeleteInputs.Response**

24.2.4 CheckUsageInputs

```
Check if a set of inputs are in use by any outputs
```

- **message CheckUsageInputs.Request**
- **message CheckUsageInputs.Response**

24.3 Type Reference

24.3.1 AdditionalInputFeatures

Additional Input Features.

struct

descrambling	**optional bissDescrambling.DescramblerSettings** Settings for descrambling of TS data. Only supports BISS version 2 and Fixed key with AES scrambling. This feature is only available on BISS products.
perFlowTriggers	**optional triggers.TriggerConfig**

24.3.2 CheckUsageInputs.Request

struct

ids	**list** of **UUID**

24.3.3 CheckUsageInputs.Response

struct

data	**map** from **UUID** to **bool**

24.3.4 CheckUsageInputsRequest

struct

`ids`	**list** of **UUID**

24.3.5 CheckUsageInputsResponse

struct

`data`	**map** from **UUID** to **bool**

24.3.6 Dejitter

Settings for dejitter.

struct

`settings`	**DejitterSettings** Contains a variant describing which dejitter version to use and its settings.
`bufferSize`	**int** Buffer size given in milliseconds.

24.3.7 DejitterCbr

Settings for constant bit rate dejittering.

struct

`enableStatus`	**bool** Undocumented in version 1.1. Always set to false.
`useRtpTimestamps`	**bool** Use RTP Timestamps for better dejiterring.

24.3.8 DejitterPcr

Settings for PCR dejittering.

struct

`preferredPcrPid`	**optional int** Undocumented in version 1.1. Leave empty.

24.3.9 DejitterRtp

Settings for RTP dejittering.

struct

`enableStatus`	**bool** Undocumented in version 1.1. Always set to false.
	RtpDejitterProfile

profile	Selects which RTP dejitter profile to use for dejittering

24.3.10 DejitterSettings

Dejitter settings variant. Allows selection between RTP dejitter, CBR dejitter, and PCR dejitter mode.

variant

rtp	**DejitterRtp**
cbr	**DejitterCbr**
pcr	**DejitterPcr**

24.3.11 DeleteInputs.Request

struct

ids	**list** of **UUID**

24.3.12 DeleteInputs.Response

empty **struct**

24.3.13 GetInputs.Request

empty **struct**

24.3.14 GetInputs.Response

struct

data	**map** from **UUID** to **Input**
info	**firewallTypes.PaginationInfo**

24.3.15 Input

An Input configuration opens the firewall for a given data stream. Configuration is split into two parts: transport and analysis. Transport configuration consists of IP protocol, or ASI settings, while analysis configuration consists of settings describing how the data should be interpreted by the system.

struct

label	**string** Human readable label for the input.
enabled	**bool** Whether the input is enabled or not.
transportSettings	**InputTransportSettings** Transport properties for this input.
analyzeMode	**optional InputAnalyzeMode** Analyze properties for this input.
	optional AdditionalInputFeatures

`additionalFeatures`	Some optional additional features, only supported on some products.

24.3.16 InputAnalyzeAtsc

Settings applying to an ATSC input.

struct

`dejitter`	**optional Dejitter** Dejitter settings for the input.

24.3.17 InputAnalyzeDvb

Settings applying to a DVB input.

DVB Mode will analyze PID 0 (PAT) and PID 17 (SDT). The system will then analyze each PMT signaled in the PAT and announce it in the system for further processing, e.g, route to an output.

struct

`dejitter`	**optional Dejitter** Dejitter settings for the input.

24.3.18 InputAnalyzeMode

Selecting the analyze mode and settings for the input source.

variant

`mpegMode`	**InputAnalyzeMpeg** analyze the input as MPEG TS.
`dvbMode`	**InputAnalyzeDvb** analyze the input as DVB TS.
`rtpMode`	**InputRtp** do not analyze the input as TS.
`atscMode`	**InputAnalyzeAtsc** analyze the input as ATSC TS.
`sptsMode`	**InputAnalyzeSpts** analyze the input as SPTS. Only valid on the 100G product.

24.3.19 InputAnalyzeMpeg

Settings applying to an MPEG input.

struct

`dejitter`	**optional Dejitter** Dejitter settings for the input.

24.3.20 InputAnalyzeSpts

Settings applying to an SPTS input.

struct

`dejitter`	**optional Dejitter** Dejitter settings for the input.

24.3.21 InputRtp

Settings applying to an RTP input.

struct

`dejitter`	**optional Dejitter** Dejitter settings for the input.

24.3.22 InputTransportSettings

InputTransportSettings determines which type of transport is used for this input.

variant

`udp`	**InputUdpTransportSettings** If set, assumes this input is a UDP input and configures it using the provided settings.
`asi`	**asiModule.InputAsiTransportSettings** If set, assumes this input is an ASI input and configures it using the provided settings.
`srt`	**srt.SrtInputSettings** If set, assumes this input is an SRT input and configures it using the provided settings.
`zixi`	**zixi.ZixiInputSettings** If set, assumes this input is an Zixi input and configures it using the provided settings.

24.3.23 InputUdp

UDP Input. May either be a single source or multiple in seamless mode (SMTPE-2022-7).

variant

`seamless`	**InputUdpSeamless**
`single`	**InputUdpSettings**

24.3.24 InputUdpSeamless

Set the input in seamless transport mode (SMTPE-2022-7). The two sources, a and b, must be setup at different interfaces.

struct

`a`	**InputUdpSettings** UDP settings for the A pipe.
`b`	**InputUdpSettings** UDP settings for the B pipe.

preferred	**InputUdpSeamless.preferred** Preferred pipe.
identical	**bool** Means the expected sources are 2022-7 compliant, i.e identical; If they are not a proper alarm will be raised
optimizeLatency	**bool** Ensure the 2022-7 buffer is optimized for latency. Resets the input if optimization is required.

24.3.25 InputUdpSeamless.preferred

enum

A
B
FLOATING

24.3.26 InputUdpSettings

UDP input settings. If destination is a multicast address IGMP join will be performed. A destination must be unique for a given interface.

struct

interfaceId	**UUID** Interface Id. See ipinterface.IpInterface
destination	**firewallTypes.SocketAddress** Unicast or multicast address
fec	**bool** FEC settings
rtp	**bool** Expect RTP header present, used when InputAnalyzeMode is rtpMode.
sources	**list** of **string** List of IGMP source addresses. Maximum number of sources allowed is 20.

24.3.27 InputUdpTransportSettings

InputUdpTransportSettings Settings for all UDP transport inputs

struct

input	**InputUdp** UDP input. Variant either single or seamless.
inputBufferSize	**int** Input buffer size. Buffer for reconstructing missing packets on the input. Ranges 1-400ms (default 20ms) for seamless and 300-400ms(default 300ms) for FEC. 0ms is set if neither FEC or seamless is active.
trafficPolicing	**IpTrafficPolicing** Traffic Policing settings.

`rtpExcessiveLossThreshold`	**int** Configurable threshold for the excessive RTP loss alarm (default 100).
`alarmWhenScrambled`	**bool** If set as true then enable alarm when input is scrambled, otherwise alarm is disable.

24.3.28 IpTrafficPolicing

Type to hold the traffic policing information for IP Inputs.

struct

`enabled`	**bool** When true, the input stream will be taken down if the input bitrate exceeds maxBitrate.
`maxBitrate`	**int** Max allowed bitrate for traffic policing, in units of (kbps). Allowed range is 0 - 213 mbps.

24.3.29 RtpDejitterProfile

The supported profiles for RTP dejittering

enum

`SMPTE_2022_6`
`SMPTE_2022_2`

24.3.30 SetInputs.Request

struct

`data`	**map** from **UUID** to **Input**

24.3.31 SetInputs.Response

empty **struct**

25 inputStatus (1.1)

25.1 Type Reference

25.1.1 FlowId

int

25.1.2 PidStatus

```
Per PID status.

@param pid         The PID value
@param serviceId   The associated service ID (as signalled in PAT/PMT)
@param ccErrors    Accumulated number of CC errors
                   Natural number disguised as an integer: The receiver   must cast back to unsi
@param dataRate    Bitrate [bit/s]
                   Natural number disguised as an integer: The receiver   must cast back to unsi
@param scrambling  Scrambling status of PID
```

struct

pid	**int**
serviceId	**optional int**
ccErrors	**int**
dataRate	**bigint**
scrambling	**Scrambling**

25.1.3 Scrambling

```
Scrambling state of PID

@param NONE  No scrambling
@param EVEN  Scrambling with even control word
@param ODD   Scrambling with odd control word
```

enum

NONE
EVEN
ODD

26 ipCardSettings (1.2)

26.1 Overview

Changelog:

1.2

```
- **Added**
  - DnsConfig was added to `CardSettings`.
```

1.1

- **Added**
  - The field `nmos` was added to `CardSettings`.

26.2 Command Reference

26.2.1 GetCardSetting

- **message GetCardSetting.Request**
- **message GetCardSetting.Response**

26.2.2 SetCardSetting

- **message SetCardSetting.Request**
- **message SetCardSetting.Response**

26.2.3 DeleteCardSetting

- **message DeleteCardSetting.Request**
- **message DeleteCardSetting.Response**

26.3 Type Reference

26.3.1 CardSettings

Settings for the IP Gateway card

struct

`mode`	**TransportMode** The currently active transport mode for this card. Note: For X5, the value of this parameter is ignored, as it is not applicable.
	string

hostname	The hostname of this card. If several elements are given, the first element will become the Linux hostname, and the others will become the DNS suffix. Must follow the Linux hostname rules: "Each element of the hostname must be from 1 to 63 characters long and the entire hostname, including the dots, can be at most 253 characters long. Valid characters for hostnames are ASCII(7) letters from a to z, the digits from 0 to 9, and the hyphen (-). A hostname may not start with a hyphen."
nmos	**optional** **nmosConfig.NmosConfig** Parameters for accessing NMOS registry and labeling templates (No NMOS support for IPGW 3.38 yet)
dnsConfig	**optional** **dnsConfig.DnsConfig** Optional configuration for DNS

26.3.2 DeleteCardSetting.Request

struct

ids	**list** of **UUID**

26.3.3 DeleteCardSetting.Response

empty **struct**

26.3.4 GetCardSetting.Request

empty **struct**

26.3.5 GetCardSetting.Response

struct

data	**map** from **UUID** to **CardSettings**

26.3.6 SetCardSetting.Request

struct

data	**map** from **UUID** to **CardSettings**

26.3.7 SetCardSetting.Response

empty **struct**

26.3.8 TransportMode

Settings for the IP Gateway card

enum

`Normal`	IP Multicast / Unicast (default mode)
`Zixi`	Zixi
`SRT`	SRT (Secure, Reliable, Transport)

27 ipTriggers (1.1)

27.1 Overview

Changelog:

1.1

- **Added**
  - `Ipgw100gConfigurableTriggerAlarmIds` is added as available alarms for trigger management for 100G IPGW.
- **Changed**
  - Variable `IpgwConfigurableTriggerAlarmIds` was renamed to `Ipgw10gConfigurableTriggerAlarmIds` as available alarms for trigger management for 10G IPGW.

28 ipinterface (1.11)

28.1 Overview

Changelog:

1.11

- **Added**
  - Type `OutputRedundancyBehaviour` was created, which contains the old field `switchbackThreshold` from `IpInterface`, and a new field `gracePeriod`. It's used as a field called `redundancyBehaviour` in `IpInterface`.

1.10

- **Added**
  - The fields 'D1', 'D2', and 'D3' were added to `InterfaceName`.

1.9

- **Removed**
  - The field `enabled` was removed from `IpInterface`.

1.8

- **Added**
  - The fields `enabled` and `autoconfigure` were added to `IpInterfaceConfig`.
- **Changed**
  - The field `name` in `IpInterface` was changed from a string to an enum.
  - The fields `ipv4` and `ipv6` in `IpInterface` are not optional anymore.

1.7

- **Changed**
  - The field `IpInterface.lldp` was relocated to the `PhysicalPort` object in the `physicalports` API.

1.6

- **Added**
  - The field `lldp` was added to the `IpInterface` object.

1.5

- **Added**
  - The field `isManagementActive` was added to the `IpInterface` object.

1.4

- **Added**
  - The field `switchbackThreshold` was added to the `IpInterface` object.

1.3

- **Added**
  - A `label` field was added to the `IpInterface` object

1.2

- **Removed**
  - The fields `dejitterBufferSize`, `tx` and `rx` were moved from `IpInterface` object to `physicalports` API.

28.2 Command Reference

28.2.1 GetIpInterfaces

Get IP interfaces

Returns all IP interfaces that are configured for the card in question.

- **message GetIpInterfaces.Request**
- **message GetIpInterfaces.Response**

28.2.2 SetIpInterfaces

- **message SetIpInterfaces.Request**
- **message SetIpInterfaces.Response**

28.2.3 DeleteIpInterfaces

Delete IP interfaces

- **message DeleteIpInterfaces.Request**
- **message DeleteIpInterfaces.Response**

28.2.4 GetDhcpv4Leases

- **message GetDhcpv4Leases.Request**
- **message GetDhcpv4Leases.Response**

28.2.5 GetDhcpv6Leases

- **message GetDhcpv6Leases.Request**
- **message GetDhcpv6Leases.Response**

28.2.6 GetDuid

- **message GetDuid.Request**
- **message GetDuid.Response**

28.2.7 GetIaid

- **message GetIaid.Request**
- **message GetIaid.Response**

28.2.8 GetRouterAdvertisements

- **message GetRouterAdvertisements.Request**
- **message GetRouterAdvertisements.Response**

28.2.9 InvokePingCommand

- **message InvokePingCommand.Request**
- **message InvokePingCommand.Response**

28.3 Type Reference

28.3.1 DeleteIpInterfaces.Request

struct

`ids`	**list** of **UUID**

28.3.2 DeleteIpInterfaces.Response

empty **struct**

28.3.3 DhcpStatus

- DHCP status.
-
-

enum

`INVALID`	
`LINK_LOCAL`	
`DHCP_LEASED`	

28.3.4 Dhcpv4Lease

DHCPv4 Lease.

If DHCPv4 is enabled on an interface, this contains the relevant addressing information from the lease.

struct

`status`	**DhcpStatus** Indicates the current state/type of the provided autoconfigurated addressing information.
`address`	**optional string** The IPv4 address which the client running on the interface has received from the DHCPv4 server.
`gateway`	**optional string** The gateway address (default router) which was received from the DHCPv4 server.
`netmask`	**optional int** The network mask, i.e. how many of the most significant bits that are fixed in the subnet's addresses.

28.3.5 Dhcpv6Lease

DHCPv6 lease.

Contains a subset of information found in a valid DHCPv6 lease for the interface. Note that in contrast to DHCPv4, the default gateway is not contained in the lease, but is rather found in the ICMPv4 Router Advertisement message that preceeds it in the IPv6 stateful autoconfiguration flow.

struct

`status`	**DhcpStatus** Indicates the current state/type of the provided autoconfigurated addressing information.
`address`	**optional string** The IPv6 address that the client has received from the DHCPv6 server.

28.3.6 GetDhcpv4Leases.Request

GetDhcpv4LeasesRequest

28.3.7 GetDhcpv4Leases.Response

GetDhcpv4LeasesResponse

28.3.8 GetDhcpv4LeasesRequest

struct

`interfaces`	**optional list** of **UUID**

28.3.9 GetDhcpv4LeasesResponse

struct

`data`	**map** from **UUID** to **optional** **Dhcpv4Lease**

28.3.10 GetDhcpv6Leases.Request

GetDhcpv6LeasesRequest

28.3.11 GetDhcpv6Leases.Response

GetDhcpv6LeasesResponse

28.3.12 GetDhcpv6LeasesRequest

struct

`interfaces`	**optional list** of **UUID**

28.3.13 GetDhcpv6LeasesResponse

struct

`data`	**map** from **UUID** to **optional** **Dhcpv6Lease**

28.3.14 GetDuid.Request

GetDuidRequest

28.3.15 GetDuid.Response

GetDuidResponse

28.3.16 GetDuidRequest

empty **struct**

28.3.17 GetDuidResponse

struct

`duid`	**optional** **string**

28.3.18 GetIaid.Request

GetIaidRequest

28.3.19 GetIaid.Response

GetIaidResponse

28.3.20 GetIaidRequest

struct

`interface`	**InterfaceName**

28.3.21 GetIaidResponse

struct

`iaid`	**optional** **string**

28.3.22 GetIpInterfaces.Request

empty **struct**

28.3.23 GetIpInterfaces.Response

struct

data	**map** from **UUID** to **IpInterface**
info	**firewallTypes.PaginationInfo**

28.3.24 GetRouterAdvertisements.Request

GetRouterAdvertisementsRequest

28.3.25 GetRouterAdvertisements.Response

GetRouterAdvertisementsResponse

28.3.26 GetRouterAdvertisementsRequest

struct

interfaces	**optional list** of **UUID**

28.3.27 GetRouterAdvertisementsResponse

struct

data	**map** from **UUID** to **optional RouterAdvertisement**

28.3.28 InterfaceName

The name of the interface to use for an IpInterface.

enum

CTRL
D1
D2
D3
D4
UNDEFINED

28.3.29 InvokePingCommand.Request

PingRequest

28.3.30 InvokePingCommand.Response

PingResponse

28.3.31 IpInterface

IP interface.

Defining a pair of IPv4 and IPv6 addresses for a single physical or virtual interface.

struct

isManagementActive	**bool**
label	**string**
redundancyBehaviour	**OutputRedundancyBehaviour**
physicalPortId	**UUID**
name	**InterfaceName**
vlan	**int**
ipv4	**IpInterfaceConfig**
ipv6	**IpInterfaceConfig**

28.3.32 IpInterfaceConfig

IP interface configuration.

Contains a single set of either IPv4 or IPv6 addressing information.

struct

ipAddress	**firewallTypes.IpAddress**
gateway	**firewallTypes.IpAddress**
netmask	**int**
enabled	**bool**
autoconfigure	**bool**

28.3.33 OutputRedundancyBehaviour

struct

switchbackThreshold	**optional int**
gracePeriod	**optional int**

28.3.34 PingRequest

struct

address	**string**
interfaceId	**UUID**

28.3.35 PingResponse

struct

pingOutput	**string**

28.3.36 RouterAdvertisement

ICMPv6 Router Advertisement

Subset of the information given in a IPv6 Router Advertisement message. Advertisements are either sent on request upon receiving a Router Solicit (RS) from a node in the network, or periodically.

struct

`status`	**RouterAdvertisementStatus** Indicates the current state/type of the provided autoconfigurated addressing information.
`prefixLength`	**optional int** The network prefix for the network which the router sending the RA is responsible for.
`address`	**optional string** The IPv6 address of the router that sent the RA, for the given prefix. Usually a IPv6 link-local address.

28.3.37 RouterAdvertisementStatus

enum

`INVALID`
`VALID`

28.3.38 SetIpInterfaces.Request

struct

`data`	**map** from **UUID** to **IpInterface** The key in this map uniquely identifies an IpInterface object in the system. It is required to be deterministic based on given parameters in the IpInterface object in the following manner: `UUID5(BASE_UUID_NAMESPACE, string(IpInterface.name) + "." + string(IpInterface.vlan))` The base UUID namespace is a hardcoded value and all interfaces are required to use this uuid as the base for the uuid calculations. BASE_UUID_NAMESPACE = abc5aa1e-7750-ba12-edef-56789abcdef0. Example: `UUID5("abc5aa1e-7750-ba12-edef-56789abcdef0", "ni0.0")` Note that specifying isManagementActive on more than one interface is ill-defined. When specifying isManagementActive on interfaces, you should read out all the interfaces with GetIpInterfaces, modify the data to ensure that exactly one interface has isManagementActive enabled, and then submit all the data with a SetIpInterface call.

28.3.39 SetIpInterfaces.Response

empty **struct**

29 linkModes (1.5)

29.1 Overview

```
# Changelog
## 1.5
- **Added**
    - Added the following members to IpLinkSpeed: "IP_25G", "IP_100G"
  **Removed**
    - Removed the following members from IpLinkSpeed: "IP_25G_COPPER", "IP_25G_OPTICAL", "IP_100
    - Removed the following member from IpLinkMode: "IP_100G_COPPER"
## 1.4
- **Added**
    - Added the following members to:
      - IpLinkMode: "IP_1G", "IP_100G_COPPER", "IP_100G_OPTICAL"
      - IpLinkSpeed: "IP_10M", "IP_100M", "IP_1G", "IP_25G_COPPER", "IP_25G_OPTICAL", "IP_100G_C
  **Removed**
    - Removed the following members from IpLinkSpeed: "IP_25G"
```

29.2 Type Reference

29.2.1 IpFecMode

enum

NO_FEC
RS_FEC
FC_FEC

29.2.2 IpLinkMode

enum

IP_1G
IP_10G
IP_25G_COPPER
IP_25G_OPTICAL
IP_40G
IP_4x10G
IP_100G_OPTICAL

29.2.3 IpLinkSpeed

enum

IP_10M
IP_100M
IP_1G
IP_10G
IP_25G
IP_40G
IP_4x10G

```
IP_100G
NO_LINK
```

30 lldp (1.0)

30.1 Type Reference

30.1.1 LldpMedInventory

LLDP-MED Inventory. Appear specific implementation in parentheses.

struct

hwRev	**string** Hardware Revision
swRev	**string** Software Revision (Software Version)
fwRev	**string** Firmware Revision (FPGA Version)
serial	**string** Serial Number
manufacturer	**string** Manufacturer (“Appear”)
model	**string** Model (“X10” or “X20”)
assetId	**string** Asset ID (Hardware ID)

30.1.2 LldpMode

Choose either private or public configuration of LLDP frames.

enum

PUBLIC	Only mandatory TLVs and port detail TLVs.
PRIVATE	LLDP-MED inventory in addition to the PUBLIC TLVs.

30.1.3 LldpNeighbor

The optional variables are only present if GetLldpNeighborRequest.details is set to true in the rpc request. The following descriptions are the Appear specific implementations of each LLDP TLV. If neighbors from other manufacturers are detected, TLV values might differ. The LldpMode is specified in parentheses for variables that have different implementations based on the LldpMode set in the neighbor.

struct

chassisId	**string** MAC address of CTRL on MMI 1, fallback to CTRL on MMI 2 (PRIVATE) or “N/A” (PUBLIC).
mgmtIp	**list** of **string** IPv4 address of CTRL on MMI 1 and/or CTRL on MMI 2. Also IPv6 if enabled on the neighbor.
sysDescr	**optional string** Desktop Heading (PRIVATE) or “Unknown” (PUBLIC)
	optional string

`sysName`	Hostname (PRIVATE) or Interface.Port (PUBLIC)
`portId`	**string** MAC address of local port
`portDescr`	**optional string** Slot, Interface: Port-label
`ttl`	**optional string** Time-To-Live, how long an LLDP packet received from that neighbor should be considered valid.
`lldpMedInventory`	**optional LldpMedInventory** LLDP-MED Inventory

31 mappingTemplates (1.3)

31.1 Overview

Changelog:

1.3

- **Changed**
- Import `componentMapping` changed from version 1.3 to 1.4

1.2

- **Changed**
- Import `componentMapping` changed from version 1.2 to 1.3

1.1

- **Changed**
- Import `mappingTemplates` changed from version 1.1 to 1.2

31.2 Command Reference

31.2.1 GetTemplates

Gets templates stored in the unit

- **message GetTemplates.Request**
- **message GetTemplates.Response**

31.2.2 StoreTemplates

Stores templates on the unit

- **message StoreTemplates.Request**
- **message StoreTemplates.Response**

31.2.3 DeleteTemplates

Deletes templates on the unit

- **message DeleteTemplates.Request**
- **message DeleteTemplates.Response**

31.2.4 GetTemplateStatus

Get template status

- **message GetTemplateStatus.Request**
- **message GetTemplateStatus.Response**

31.3 Type Reference

31.3.1 AppliedTemplateRules

Keeps indices of applied template rules

list of **int**

31.3.2 DeleteTemplates.Request

struct

`ids`	**list** of **UUID**

31.3.3 DeleteTemplates.Response

empty **struct**

31.3.4 GetTemplateStatus.Request

empty **struct**

31.3.5 GetTemplateStatus.Response

TemplateStatusResponse

31.3.6 GetTemplates.Request

empty **struct**

31.3.7 GetTemplates.Response

struct

`data`	**map** from **UUID** to **OutputTemplates**

31.3.8 OutputTemplates

OutputTemplates aggregates all Component Mapping and Service Name templates.

struct

`rules`	**list** of **TemplateRule** List of handled templates.

31.3.9 ServiceMapping

ServiceMapping contains changes to service name in output mappings

struct

`serviceName`	**string** Name of mapped output service

31.3.10 StoreTemplates.Request

struct

data	**map** from **UUID** to **OutputTemplates**

31.3.11 StoreTemplates.Response

empty **struct**

31.3.12 TemplateRule

TemplateRule contains matchers and mapping rules for output

struct

matchedNetworkId	**optional gatewayValidate.IntRange** Original Network ID range selector. If empty any Network ID will be matched.
matchedTransportId	**optional gatewayValidate.IntRange** Transport ID range selector. If empty any Transport ID will be matched.
matchedServiceId	**optional gatewayValidate.IntRange** Service ID range selector. If empty any Service ID will be matched.
mappings	**TemplateRuleUpdates** Rules to apply. Can be any type of supported template mappings.

31.3.13 TemplateRuleUpdates

TemplateRuleUpdates contains variant of either component or service mapping

variant

component	**componentMapping.ComponentMapping** Variant for component mappings
service	**ServiceMapping** Variant for service name mappings

31.3.14 TemplateStatus

TemplateStatus keeps track of applied rules (by its index) for particular sources in output

map from **UUID** to **list** of **AppliedTemplateRules**

31.3.15 TemplateStatusResponse

TemplateStatuses represents rules that have been applied to all outputs

struct

data	**TemplateStatus**

32 nmosConfig (1.1)

32.1 Overview

```
# Changelog

## 1.1
- **Added**
    - Added new type `ConnectionType`
    - Added member 'connectionType' to NmosConfig
```

32.2 Type Reference

32.2.1 ConnectionType

Enum to control whether the NMOS traffic is in-band (through line card dataports) or out-of-band (through the MMI).

enum

IN_BAND	
OUT_OF_BAND	

32.2.2 NmosConfig

Configuration parameters for setting up NMOS.

struct

connectionType	**ConnectionType** Configuration to control whether traffic should be in-band or out-of-band
registry	**optional nmosRegistryConfig.NmosRegistryConfigStruct** Configuration parameters for connecting to the NMOS registry
labelTemplates	**optional nmosLabelConfig.NmosLabelConfig** Custom NMOS labels configuration. This feature is enabled when the optional is set

33 nmosLabelConfig (1.0)

33.1 Overview

Changelog

1.0

NMOS labelling

NMOS labelling API allows operator to define templates based on auto text options. This templates can be defined for for node, device, source, flow, sender and receiver.

Version 1.0 allows six auto text options.

What are the auto text options?

- ${SLOT} - The slot number of the card in use.
- ${DEV_ID} - The Device number. **Not supported for Node templates**.
- ${ESS_GR} - The label of the group where the essences belong to. **Not supported for Node templates**.
- ${ESS} - The label of the Essence in question. **Not supported for Node and Device templates**.
- ${ESS_T} - The Essence type: Video, Audio, Ancillary. This auto text option is substituted by what is defined in `EssenceTypeStrings` for each of the essences type. **Not supported for Node and Device templates**.
- ${ESS_T_INDEX} - The number of the essence belonging to one of the essence type - video, audio, ancillary. **Not supported for Node and Device templates**.

How to use the NMOS Labelling.

- NMOS templates are set in the fields part of `ResourceLabelTemplates`. Unused templates should be set with an empty string.
- Each field in `ResourceLabelTemplates` can have as many auto text options as the operator wants, as long as they are supported for the respective template.
- `EssenceTypeStrings` strings will substitute the auto text option ${ESS_T} with the defined string for the respective Essence type.

Example of strings to use for the fields in `ResourceLabelTemplates` **and** `EssenceTypeStrings`

- `nodeTemplate` - ECx210 Decoder ST2110 Sender Slot ${SLOT}
- `deviceTemplate` - ECx210 Decoder Sender Slot ${SLOT} ID ${DEV_ID}
- `sourceTemplate` - ECx210 Slot ${SLOT} ID ${DEV_ID}, ${ESS_T} Source: ${ESS}, Group: ${ESS_GR}, Essence Number: ${ESS_T_INDEX}
- `flowTemplate` - ECx210 Slot ${SLOT} ID ${DEV_ID}, ${ESS_T} Flow: ${ESS}, Group: ${ESS_GR}, Essence Number: ${ESS_T_INDEX}
- `senderTemplate` - ECx210 Slot ${SLOT} ID ${DEV_ID}, ${ESS_T} Sender: ${ESS}, Group: ${ESS_GR}, Essence Number: ${ESS_T_INDEX}
- `receiverTemplate` - ECx210 Slot ${SLOT} ID ${DEV_ID}, ${ESS_T} Receiver: ${ESS}, Group: ${ESS_GR}, Essence Number: ${ESS_T_INDEX}
- `video` - Video
- `audio` - Audio
- `ancillary` - Anc

33.2 Type Reference

33.2.1 EssenceTypeStrings

Labels for each essence type to use when generating custom NMOS labels.

struct

`video`	**string** Video essence label
`audio`	**string** Audio essence label
`ancillary`	**string** Ancillary essence label

33.2.2 NmosLabelConfig

Custom NMOS labels configuration.

struct

`resourceTemplates`	**ResourceLabelTemplates** Templates for each NMOS resource
`essenceStrings`	**EssenceTypeStrings** Labels to use for each essence type

33.2.3 ResourceLabelTemplates

NMOS label templates for each NMOS resource.

struct

`nodeTemplate`	**string** Template string for the node label
`deviceTemplate`	**string** Template string for the device label
`sourceTemplate`	**string** Template string for the source label
`flowTemplate`	**string** Template string for the flow label
`senderTemplate`	**string** Template string for the sender label
`receiverTemplate`	**string** Template string for the receiver label

34 nmosRegistryConfig (1.0)

34.1 Type Reference

34.1.1 DnsConfig

Method to use for discovery of registry server

variant

mDns	**DnsConfig.mDns** Use mDNS to reach the registry server
dnsSd	**DnsSdConfig** Configuration parameters to use DNS-SD for discovery of registry service

34.1.2 DnsConfig.mDns

empty **struct**

34.1.3 DnsSdConfig

Configuration parameters for DNS-SD

struct

dnsServer	**firewallTypes.IpAddress** IPv4 address of DNS-SD server
search	**string** Search domain name

34.1.4 NmosRegistryConfigStruct

Configuration parameters for reaching the NMOS registry

struct

registryConfig	**RegistryManualConfig** Configuration parameters to use in case of failure in mDNS or DNS-SD (secondary/fallback method)
dnsConfig	**DnsConfig** Configuration parameters for mDNS or DNS-SD (primary method)

34.1.5 RegistryManualConfig

Registry information to be used in case registry server can not be reached by mDNS or DNS-SD.

struct

fallbackEnable	**bool** Enable use of given information as a fallback
	firewallTypes.IpAddress

`registryIpAddr`	IPv4 address of NMOS registry
`registryPort`	**int** Port number of NMOS registry
`registryVersion`	**commonTypes.NmosVersionNumber** Version of registry API

35 nmosRegistryStatus (1.0)

35.1 Type Reference

35.1.1 NmosRegistryStatus

NMOS registry status

struct

`ipAddr`	**optional firewallTypes.SocketAddress** Currently used IP address and port number of the registry server. This value may have been obtained from DNS-SD, mDNS, or a manual IP entry.
`registered`	**bool** True if there is an active connection to the registry server.

36 nmosStatusTypes (1.0)

36.1 Type Reference

36.1.1 NmosStatus

NmosStatus Status of the NMOS service

struct

`nmosRegistryStatus`	**nmosRegistryStatus.NmosRegistryStatus** Status of the NMOS registry connection
`nodeApi`	**map** from **physicalIpPort.PortName** to **string** Map from port to the endpoint for NMOS Node API. Ex: PortName.D1 -> "10.20.10.34:8001", PortName.D2_2 -> "10.20.11.34:8001"

37 output (1.39)

37.1 Overview

Changelog:

1.39

```
- **Changed**
  - The imported `outputMapping` version was bumped to 1.6
```

1.38

```
- **Changed**
  - The imported `srt` version was bumped to 1.7
```

1.37

```
- **Changed**
  - The imported `outputMapping` version was bumped to 1.5
```

1.36

```
- **Changed**
  - The imported `srt` version was bumped to 1.6
```

1.35

```
  - **Changed**
    - The imported `bissScrambling` version was bumped to 1.6
```

1.34

```
- **Changed**
  - The imported `srt` version was bumped to 1.5
```

1.33

```
- **Changed**
  - The imported `srt` version was bumped to 1.4
```

1.32

```
- **Changed**
  - The imported `srt` version was bumped to 1.3
```

1.31

```
- **Changed**
  - The imported outputMapping version was bumped to 1.4
```

1.30

```
- **Changed**
  - The imported outputMapping version was bumped to 1.3
  - The imported bissScrambling version was bumped to 1.5
```

1.29

```
- **Changed**
  - The imported srt version was bumped to 1.2
```

1.28

```
- **Changed**
  - The imported outputMapping version was bumped to 1.2
```

1.27

```
- **Changed**
  - The imported asiModule version was bumped to 1.11
```

1.26

```
- **Changed**
  - The imported asiModule version was bumped to 1.10
```

1.25

```
- **Changed**
  - The imported asimodule version was bumped to 1.9
  - OutputMapping bumped to 1.1
  - The imported `srt` version was bumped to 1.1.
  - The imported `zixi` version was bumped to 1.1.
  - remember output source address and port
```

1.24

```
- **Changed**
  - The imported asimodule version was bumped to 1.8
  - OutputMapping was moved to its own idcl file
```

1.23

- **Changed**
  - The imported `bissScrambling` version was bumped to 1.4.
  - The imported asimodule version was bumped to 1.7

1.22

- **Added**
  - The field `scramblingIndex` was added to
    - `DvbMultiplexServiceSettings`
    - `MpegMultiplexProgramSettings`
    - `AtscMultiplexChannelSettings`
  - The optional field `additionalFeatures` was added to `Output`.

    The `AdditionalOutputFeatures` type includes the field `scrambling`, which is used to configure scrambling on the output.
  - Bumped asiModule from 1.6 to 1.7
- **Changed**
  - The imported `asiModule` version was bumped to 1.7.

- The validation of the fields `guardInterval` and `bufferDelay` in `InputRedundancy` changed, making the intervals dependent on if the stream is CBR or VBR (in practice; whitelist/blacklist).
- Minimum allowed value of `Max offset` is 4ms.
- Minimum allowed value of `Guard interval` is 2ms when CBR and 50ms when VBR.
- Sum of `Max offset` and `Guard interval` may not be greater than 150ms.

1.21

- **Changed**
  - The validation of the field `switchbackDelay` in `InputRedundancy` changed.

    Previous minimum value was 0, but has been increased to 10. Any values below 10 are set to 10.
  - The validation of the fields `guardInterval` and `bufferDelay` in `InputRedundancy` changed for whitelist mode.

    The minimum value for the guard interval was increased to 50, the maximum buffer delay is set to 100 and the maximum allowed sum of guard interval + buffer delay is set to 150. Any values outside the legal range are adjusted to be within the legal limits.

1.20

- **Removed**
  - The field `scrambledComponents` was removed from
    * `DvbMultiplexServiceSettings`
    * `MpegMultiplexProgramSettings`
    * `AtscMultiplexChannelSettings`

1.19

- **Added**
  - The field `latencyTuning` was added to `IdenicalSources`.

    This allows specifying if main is leading or lagging.
  - The field `guardInterval` was added to `Identicalsources`.

    This allows specifying a delay used to compensate for fluctuations in the time offset between the flows.

37.2 Command Reference

37.2.1 GetOutputs

- **message GetOutputs.Request**
- **message GetOutputs.Response**

37.2.2 SetOutputs

- **message SetOutputs.Request**
- **message SetOutputs.Response**

37.2.3 DeleteOutputs

- message DeleteOutputs.Request
- message DeleteOutputs.Response

37.3 Type Reference

37.3.1 AdditionalOutputFeatures

Additional Output Features.

struct

scrambling	bissScrambling.ScramblerSettings Settings for scrambling of TS data. Only supports BISS version 2 and Fixed key with AES scrambling. This feature is only available on BISS products.

37.3.2 DeleteOutputs.Request

struct

ids	list of UUID

37.3.3 DeleteOutputs.Response

empty struct

37.3.4 FECMode

enum

A
B

37.3.5 FECSettings

struct

mode	FECMode
columns	int
rows	int

37.3.6 GetOutputs.Request

variant

filter	GetOutputsFilter

37.3.7 GetOutputs.Response

struct

data	**map** from **UUID** to **Output** A map of outputs returned by the request.
info	**firewallTypes.PaginationInfo** Meta information about the data - i.e the total number of items on the server.

37.3.8 GetOutputsFilter

struct

pagination	**optional firewallTypes.PaginatedRequest** Settings for splitting the response data into pages

37.3.9 L2TPv3

L2TPv3 Configuration

struct

udp	**bool**
flowid	**int**
sessionid	**bigint**

37.3.10 Output

Output Configurations.

Output configuration is divided into two sections: transport and mapping. Transport settings consist of IP protocol or ASI settings, while mapping settings consist of settings related to stream content.

struct

label	**string** Name of the output
enabled	**bool** If the output is enabled or not
transportSettings	**OutputTransportSettings** Transport settings.
outputSettings	**outputMapping.OutputMapping** Mapping settings.
additionalFeatures	**optional AdditionalOutputFeatures** Some optional additional features, only supported on some products.

37.3.11 OutputTransportSettings

Variant for selecting the output transport settings

variant

udp	**OutputUdpTransportSettings** Used to configure UDP outputs
asi	**asiModule.OutputAsiTransportSettings** Used to configure ASI outputs
srt	**srt.SrtOutputSettings** Used to configure SRT outputs
zixi	**zixi.ZixiOutputSettings** Used to configure Zixi outputs

37.3.12 OutputUdp

UDP output variant - selects either cloned output or single output mode

variant

cloned	**OutputUdpCloned** Configuration for port A and port B
single	**OutputUdpSettings** Configuration for the only port used by this output

37.3.13 OutputUdpCloned

Cloned Configuration for port A and port B. Both interfaces must be configured to run at the same link speed.

struct

a	**OutputUdpSettings**
b	**OutputUdpSettings**

37.3.14 OutputUdpParameters

Parameters for UDP outputs

struct

dscp	**int** Type of service
ttl	**int** Time To Live
rtp	**bool** RTP
ts	**optional OutputUdpTsParameters** TS spesific parameters
l2tpv3	**optional L2TPv3** L2TPv3 settings
vlanPriority	**optional VlanPriority**

37.3.15 OutputUdpSettings

Output settings for UDP outputs.

struct

interfaceId	**UUID**
destination	**firewallTypes.SocketAddress**
sourceAddressEnabled	**bool**
sourceAddress	**string**
sourcePortEnabled	**bool**
sourcePort	**int**
fecEnabled	**bool**
fec	**optional FECSettings**

37.3.16 OutputUdpTransportSettings

Transport settings for an UDP output

struct

settings	**OutputUdpParameters**
output	**OutputUdp** UDP output settings - cloned or single port

37.3.17 OutputUdpTsParameters

UDP settings spesific to TS

struct

mpegPacketsPerFrame	**int** How many MPEG packets are sent per transport frame

37.3.18 SetOutputs.Request

struct

data	**map** from **UUID** to **Output**

37.3.19 SetOutputs.Response

empty **struct**

37.3.20 VlanPriority

VlanPriority - Set priority related 802.1Q header flags

Value: Traffic types 0 Best effort 1 Background (lowest priority) 2 Excellent effort 3 Critical applications 4 Video, < 100 ms latency and jitter 5 Voice, < 10 ms latency and jitter 6 Internetwork control

struct

pcp	**int**
dei	**bool**

38 outputMapping (1.6)

38.1 Overview

Changelog:

1.6

```
- **Changed**
  - `componentMapping` bumped from 1.3 to 1.4
```

1.5

```
- **Added**
  - Added field `sptsMode` to `OutputMappingTsWhitelist`
  - Added `SptsSource`, `SptsSettings` structs
```

1.4

```
- **Changed**
  - componentMapping bumped from 1.2 to 1.3
  - genericDescriptor bumped from 1.1 to 1.2
```

1.3

```
- **Added**
  - Added autoEmm option to TsSettings
```

1.2

```
- **Added**
  - Added floating switch mode to InputRedundancy
```

1.1

```
- **Changed**
  - genericDescriptor bumped to v 1.1
```

1.0

```
- **Added**
  - Added mapping part from output.1.24.idcl
  - The field `externalPids` was added to `OutputMappingTsBlacklist`
```

38.2 Type Reference

38.2.1 AnnexA

QAM Annex A input parameters, per the ITU-T J.83 specification.

By setting the Annex A parameters, one specifies how a TS should be QAMed, which ultimately results in a specific bitrate as described in the ITU-T spec.

struct

Modulation

`modulation`	Specifies the QAM modulation scheme, either QAM64 or QAM256.
`symbolrate`	**Symbolrate** Symbol rate to use for QAM.

38.2.2 AnnexB

Annex B input parameters, per the ITU-T J.83 specification.

In contrast to Annex A, Annex B only has two choices for how to QAM, and therefore only has two possible output bitrates.

struct

`modulation`	**Modulation** Specifies QAM modulation scheme, either QAM64 or QAM256. QAM64 results in a output bitrate of 26.970377 Mbps. QAM256 results in a output bitrate of 38.810700 Mbps.

38.2.3 AnnexC

QAM Annex C input parameters, per the ITU-T J.83 specification.

By setting the Annex C parameters, one specifies how a TS should be QAMed, which ultimately results in a specific bitrate as described in the ITU-T spec. Note that the resulting bitrate is the same as for Annex A.

struct

`modulation`	**Modulation** Specifies the QAM modulation scheme, either QAM64 or QAM256.
`symbolrate`	**Symbolrate** Symbol rate to use for QAM.

38.2.4 AtscChannelNumber

Major and minor channel overrides

struct

`majorChannelNumber`	**int** Override major channel number.
`minorChannelNumber`	**int** Override minor channel number.

38.2.5 AtscChannelSource

Source selection for a specific input service

struct

`source`	**UUID** UUID for a input service.
`inputRedundancy`	**InputRedundancy**

priority	**int** Value to identify the impact of losing specific source for redundancy logic
settings	**AtscMultiplexChannelSettings** PID, PMT, CVCT and TVCT filters and overrides.

38.2.6 AtscMultiplexChannelSettings

PID, PMT, and SDT filters and overrides.

struct

programNumber	**optional int**
channelNumber	**optional AtscChannelNumber** Override channel number.
shortName	**optional string** Override short name.
componentLayout	**componentMapping.ComponentLayoutMap** Remap, copy, and filter PIDs and components.
pmtPid	**optional int** Override PMT PID.
componentGeneration	**list** of **componentMapping.ComponentGeneration**
descriptors	**list** of **genericDescriptor.GenericPmtDescriptorEntry** Add, change or remove descriptors on this service.
scramblingIndex	**optional int** Scrambling index. May be given if scrambling is enabled for the TS, and this services should have different settings than the default settings.

38.2.7 AtscMultiplexSource

ATSC multiplex source type, which must be either be a channel, an EMM or an external PID. An empty AtscMultiplexSource is considered an invalid config.

variant

channel	**AtscChannelSource** An ATSC channel (service)
emm	**EmmSource** An EMM source
externalPid	**ExternalPidMapping** An External PID

38.2.8 AtscSource

variant

multiplex	**list** of **AtscMultiplexSource**

38.2.9 AtscTableSettings

ATSC PSI Table Settings

Settings to choose playout of PSI tables

struct

`cvct`	**bool** Set playout of CVCT enabled/disabled
`tvct`	**bool** Set playout of TVCT enabled/disabled

38.2.10 Bitrate

Bitrate describes the bitrate scheme to be used. Only one of the parameters can be specified.

variant

`manual`	**int** An integer that sets the desired bitrate in kbps (max 3000Mbps).
`annexA`	**AnnexA** A special struct for ITU-T J.83 Annex A, which contains modulation type and symbol rate.
`annexB`	**AnnexB** A special struct for ITU-T J.83 Annex B, which contains modulation type.
`annexC`	**AnnexC** A special struct for ITU-T J.83 Annex C, which contains modulation and symbol rate.
`vsb8`	**Vsb8** Used to indicate that the specific VSB-8 bitrate is to be used.

38.2.11 DvbMultiplexServiceSettings

PID, PMT, and SDT filters and overrides.

struct

`serviceId`	**optional int** Override service ID.
`serviceName`	**optional string** Override service name.
`serviceType`	**optional int** Override service type.
`serviceProvider`	**optional string** Override service provider.
`componentLayout`	**componentMapping.ComponentLayoutMap** Remap, copy, and filter PIDs and components.
`pmtPid`	**optional int** Override PMT PID.
`componentGeneration`	**list** of **componentMapping.ComponentGeneration**

`descriptors`	**list** of **genericDescriptor.GenericPmtDescriptorEntry** Add, change or remove descriptors on this service.
`scramblingIndex`	**optional int** Scrambling index. May be given if scrambling is enabled for the TS, and this service should have different settings than the default settings.

38.2.12 DvbMultiplexSource

DVB multiplex source type, which must be either be a service, an EMM or an external PID. An empty DvbMultiplexSource is considered an invalid config.

variant

`service`	**DvbServiceSource** An DVB service
`emm`	**EmmSource** An EMM source
`externalPid`	**ExternalPidMapping** An External PID

38.2.13 DvbServiceSource

Source selection for a specific input service

struct

`source`	**UUID** UUID for a input service. The service ID can be queried using services:1.0/GetInputServices or calculated: `UUID5(Input UUID, "dvbsource service SERVICEID")`.
`inputRedundancy`	**InputRedundancy** is Input Redundancy configuration for this output
`priority`	**int** Value to identify the impact of losing specific source for redundancy logic
`settings`	**DvbMultiplexServiceSettings** PID, PMT, and SDT filters and overrides.

38.2.14 DvbSource

variant

`multiplex`	**list** of **DvbMultiplexSource**

38.2.15 EmmSource

EMM source

struct

	UUID

source	UUID of an EMM
inputRedundancy	**InputRedundancy** is Input Redundancy configuration for this output
priority	**int** Value to identify the impact of losing specific source for redundancy logic

38.2.16 ExternalPidMapping

Map specific pid from a source with the specified uuid to specific destination pid on output

struct

source	**UUID** UUID to identify which input source to map pid from
priority	**int** Value to identify the impact of losing specific source for redundancy logic
srcPid	**int** PID to map from the specified source
dstPid	**int** PID to map the source PID to on the output

38.2.17 HotStandby

struct

identicalSources	**optional IdenticalSources** When set MAIN and first BACKUP sources will be switched internally in FPGA module. MAIN and first BACKUP sources should have identical contents.

38.2.18 IdenticalSources

struct

bufferDelay	**int** Delay in milliseconds for offset handling between streams.
guardInterval	**int** Delay in milliseconds used to compensate for fluctuations in the time offset between the flows
switchOnTei	**bool** When enabled FPGA module additionally uses TEI status to assess alternative health. Available only for HotStandbyWithIdenticalSources switch mode. Only one alternative allowed.
latencyTuning	**LatencyTuning** Selected latency tuning strategy

38.2.19 InputRedundancy

Data structure related to Input Redundancy feature

struct

`mode`	**InputRedundancySwitchMode** Selected switch mode
`backups`	**list** of **UUID** List of alternative sources
`preferredSource`	**optional PreferredSource** When set allows to override default source preferences
`switchbackDelay`	**int** Delay in seconds to wait before switching back to a more preferred alternative or MAIN source.
`hotStandby`	**optional HotStandby** When set all alternative's bitrates will be monitored. Flows are preemptively mapped to the output card providing improved switch times.

38.2.20 InputRedundancySwitchMode

Enum related to SwitchMode between alternative sources

enum

`Disabled`	InputRedundancy is inactive; only MAIN source is used.
`Reverting`	Always switch to the highest priority, healthy alternative. Usually after it has been healthy for a set SWITCHBACK DELAY. Main has the highest priority, then backup 1, backup 2, etc.
`Floating`	As opposed to reverting, it doesn't go back to a higher priority alternative even if it becomes healthy again after a SWITCHBACK DELAY.

38.2.21 LatencyTuning

Enum describing desired latency tuning strategy

enum

`Auto`	The first flow activated will be assumed to be leading
`MainLeading`	The Main source is assumed to be leading
`MainLagging`	The Main source is assumed to be lagging

38.2.22 Modulation

QAM Modulation Types.

The J.83 specification utilizes two QAM variants, QAM64 and QAM256. It is denoted as "64-QAM" and "256-QAM" in the ITU-T J.83 spec. The number denotes the amount of points in the signal constellation used when performing QAM.

enum

`QAM64`	
`QAM256`	

38.2.23 MpegMultiplexProgramSettings

PID and PMT filters and overrides.

struct

programNumber	**optional int** Override program number (Equivalent to serviceId in DVB).
componentLayout	**componentMapping.ComponentLayoutMap** Remap, copy and filter PIDs and components.
pmtPid	**optional int** Override the PMT PID.
componentGeneration	**list** of **componentMapping.ComponentGeneration**
descriptors	**list** of **genericDescriptor.GenericPmtDescriptorEntry** Add, change or remove descriptors on this service.
scramblingIndex	**optional int** Scrambling index. May be given if scrambling is enabled for the TS, and this services should have different settings than the default settings.

38.2.24 MpegMultiplexSource

MPEG multiplex source type, which must be either be a program, an EMM or an external PID. An empty MpegMultiplexSource is considered an invalid config.

variant

program	**MpegProgramSource** An MPEG program (service)
emm	**EmmSource** An EMM source
externalPid	**ExternalPidMapping** An External PID

38.2.25 MpegProgramSource

MPEG program source and settings bundle.

struct

source	**UUID** UUID of an input program (MPEG equivalent of DVB service)
inputRedundancy	**InputRedundancy** is Input Redundancy configuration for this output
priority	**int** Value to identify the impact of losing specific source for redundancy logic
settings	**MpegMultiplexProgramSettings** PID and PMT filters and overrides for this program.

38.2.26 MpegSource

An MPEG mode source. Can be SPTS or MPTS, where SPTS is considered a single program multiplex struct.

variant

`multiplex`	**list** of **MpegMultiplexSource** A list of MPEG sources this output consists of.

38.2.27 OutputMapping

Output settings variant, contains settings for the supported output types. It must be either tsWhitelist-Mode, tsBlacklistMode or rawMode. An empty OutputMapping is considered an invalid config.

variant

`tsWhitelistMode`	**OutputMappingTsWhitelist** Build TS by adding content
`tsBlacklistMode`	**OutputMappingTsBlacklist** Map entire TS, and possibly exclude part of the stream
`rawMode`	**OutputMappingRaw** Transparently map entire flow

38.2.28 OutputMappingAtsc

ATSC MPTS/SPTS Settings.

The system will generate PAT, PMT, MGT, and possibly CVCT and TVCT

struct

`settings`	**TsSettings** Generic stream settings.
`source`	**AtscSource** Content to multiplex.
`tables`	**AtscTableSettings** PSI table settings

38.2.29 OutputMappingDvb

DVB MPTS/SPTS Settings.

The system will generate PAT, PMT, and SDT

struct

`orgNetworkId`	**optional int** Orginal Network Id to set on stream.
`settings`	**TsSettings** Generic stream settings.
`source`	**DvbSource** Content to multiplex.

38.2.30 OutputMappingMpeg

struct

settings	**TsSettings**
source	**MpegSource**

38.2.31 OutputMappingRaw

Mapping of stream as is

struct

source	**UUID** Main source of stream
inputRedundancy	**InputRedundancy** Input Redundancy configuration for this output

38.2.32 OutputMappingSpts

SPTS Output Settings.

The system will generate PAT, PMT, and SDT

struct

orgNetworkId	**optional int** Orginal Network Id to set on stream.
settings	**TsSettings** Generic stream settings.
source	**SptsSource** Content to output

38.2.33 OutputMappingTsBlacklist

Transport stream blacklist mapping

If CBR is specified, the original NULL PIDs are discarded.

struct

source	**UUID** Main TS source
inputRedundancy	**InputRedundancy** Input Redundancy configuration for this output
bitrate	**Bitrate** Variant Bitrate - if specified, the transport is CBR, AnnexA or AnnexB with relevant attributes. Otherwise, it is VBR.
stopPids	**list** of **int** PIDs to be stopped on Playout
remapPids	**map** from **int** to **int** PIDs to be remapped on Playout
	list of **ExternalPidMapping**

externalPids	List of PID mappings from external sources

38.2.34 OutputMappingTsWhitelist

Transport stream whitelist mapping which must be either DVB, MPEG, ATSC or SPTS (only supported on 100G product). An empty OutputMappingTsWhitelist is considered an invalid config.

variant

mpegMode	**OutputMappingMpeg** MPEG mode will create an output with MPEG MPTS/SPTS settings
dvbMode	**OutputMappingDvb** DVB mode will create an output with DVB MPTS/SPTS settings
atscMode	**OutputMappingAtsc** ATSC mode will create an output with ATSC MPTS/SPTS settings
sptsMode	**OutputMappingSpts** SPTS mode will create an SPTS output. Currently only supported on the 100G product.

38.2.35 PreferredSource

Preferred source configuration

struct

index	**int** Selects a source that should be marked as preferred. Index = 0 is MAIN source, index = 1 means first backup, etc. Preferred source will be used as the default source when it is healthy. If it fails the system will switch using regular source priorities (for example: first MAIN, then first backup, then second backup and so on)
locked	**bool** When enabled disables switching to another source when preferred source is faulty

38.2.36 SptsSettings

PID, PMT, and SDT filters and overrides.

struct

serviceId	**optional int** Override service ID.
serviceName	**optional string** Override service name.
serviceType	**optional int** Override service type.
serviceProvider	**optional string** Override service provider.
	optional int

pmtPid	Override PMT PID.

38.2.37 SptsSource

SPTS source and settings bundle.

struct

source	**UUID** UUID of an input SPTS source
inputRedundancy	**InputRedundancy** is Input Redundancy configuration for this output
sptsSettings	**SptsSettings** PID, PMT, and SDT filters and overrides.

38.2.38 SymbolRate

Symbol rate.

A symbol rate is defined in DOCSIS 2.0 as (m/n) * 10.24MHz, where M and N are 16-bit unsigned, nonzero integers, and 10.24MHz is the CMTS clock frequency.

struct

m	**int** The numerator of the desired fraction of CMTS clock frequency.
n	**int** The denominator of the desired fraction of CMTS clock frequency.

38.2.39 TsSettings

Settings shared between all TS configurations

struct

tsId	**int** Transport stream ID for the output TS
bitrate	**Bitrate** Variant Bitrate - if specified, the transport is CBR, AnnexA or AnnexB with relevant attributes. Otherwise, it is VBR.
autoEmm	**bool** Automatically include CAT and EMM from the input source. Automatic CAT/EMM is supported for single service outputs (SPTS) only. Also, only the CAS ID referenced in the ECM for the service will be mapped. The CAT will be regenerated to signal the required CAS ID only.

38.2.40 Vsb8

8-VSB struct. It is used by the Bitrate type to specify that the 8-VSB modulation scheme is desired. Per the specification, 8-VSB has no configurable inputs, and returns a MPEG II stream of 19.39...Mbits/s ready for 8-VSB modulation.

empty **struct**

39 pagination (1.0)

39.1 Type Reference

39.1.1 DataPageInfo

struct

startIndex	**int**
hasReachedEnd	**bool**

39.1.2 DataSetInfo

struct

token	**optional UUID** Used to inform about changes in dataset. If backend can detect changes in dataset then this should have a value. If this value changes it means that dataset has changed.
size	**optional int** Total size of full dataset. Should have a value if backend can know the size of full dataset.

39.1.3 DataSetSlice

struct

startIndex	**int**
elementCount	**int**

39.1.4 PaginatedDataInfo

struct

datasetInfo	**DataSetInfo**
pageInfo	**DataPageInfo**

39.1.5 PaginatedQueryResult

variant

ok	**PaginatedDataInfo** Requested data can be addressed and you get it now.
processing	**PaginatedQueryResult.processing** Requested data can be addressed but you can't get it now, system started to retrieve the data so you will be able to get it soon.
noContent	**PaginatedQueryResult.noContent** Requested data can be addressed but you can't get it now, system is in a state in which it's currently impossible to retrieve data.

39.1.6 PaginatedQueryResult.noContent

empty **struct**

39.1.7 PaginatedQueryResult.processing

empty **struct**

40 performRecording (1.1)

40.1 Overview

Changelog:

1.1

- **Added**
  - PerformNetworkTrafficRecording
  - GetRecordingStatus
  - GetRecording

1.0

- **Added**
  - PerformRecording

40.2 Command Reference

40.2.1 PerformRecording

Performs a pcap recording on the input, which is then saved to a file. The recording is max 10 seconds, or 50mb in size. Whichever comes first. A seamless recording will capture both multicasts / inputs into one pcap file.

- **message PerformRecording.Request**
- **message PerformRecording.Response**
- **message PerformRecording.Failure**

40.2.2 PerformNetworkTrafficRecording

Performs a pcap recording of the network traffic on all interfaces, which is then saved to a file. The recording will stop after the specified duration has elapsed, or the filesize reaches 50mb. Whichever comes first.

- **message PerformNetworkTrafficRecording.Request**
- **message PerformNetworkTrafficRecording.Response**
- **message PerformNetworkTrafficRecording.Failure**

40.2.3 GetRecordingStatus

```
Checks if a recording is complete, in progress or failed.
```

- **message GetRecordingStatus.Request**
- **message GetRecordingStatus.Response**
- **message GetRecordingStatus.Failure**

40.2.4 GetRecording

```
Gets the url to the recorded pcap file
```

- **message GetRecording.Request**
- **message GetRecording.Response**
- **message GetRecording.Failure**

40.3 Type Reference

40.3.1 GetRecording.Failure

GetRecordingFailed

40.3.2 GetRecording.Request

GetRecordingRequest

40.3.3 GetRecording.Response

GetRecordingResponse

40.3.4 GetRecordingFailed

Parameters for GetRecording failure

struct

code	**Status**
msg	**string**

40.3.5 GetRecordingRequest

No parameters for GetRecording Request

empty **struct**

40.3.6 GetRecordingResponse

Parameters for GetRecording Response

struct

urlToPcap	**string**

40.3.7 GetRecordingStatus.Failure

GetRecordingStatusFailed

40.3.8 GetRecordingStatus.Request

GetRecordingStatusRequest

40.3.9 GetRecordingStatus.Response

GetRecordingStatusResponse

40.3.10 GetRecordingStatusFailed

Parameters for GetRecordingStatus failure

struct

code	**Status**
msg	**string**

40.3.11 GetRecordingStatusRequest

No parameters for GetRecordingStatus Request

empty **struct**

40.3.12 GetRecordingStatusResponse

Parameters for GetRecordingStatus Response

struct

status	**Status**

40.3.13 PerformNetworkTrafficRecording.Failure

PerformNetworkTrafficRecordingFailed

40.3.14 PerformNetworkTrafficRecording.Request

PerformNetworkTrafficRecordingRequest

40.3.15 PerformNetworkTrafficRecording.Response

PerformNetworkTrafficRecordingResponse

40.3.16 PerformNetworkTrafficRecordingFailed

Parameters for PCaptureNetworkTraffic failure

struct

code	**Status**
msg	**string**

40.3.17 PerformNetworkTrafficRecordingRequest

Parameters for PCaptureNetworkTraffic Request

struct

duration	**int**

40.3.18 PerformNetworkTrafficRecordingResponse

Parameters for PCaptureNetworkTraffic Response

struct

status	**Status**

40.3.19 PerformRecording.Failure

PerformRecordingFailed

40.3.20 PerformRecording.Request

PerformRecordingRequest

40.3.21 PerformRecording.Response

PerformRecordingResponse

40.3.22 PerformRecordingFailed

Parameters for PCapture failure

struct

code	**Status**
msg	**string**

40.3.23 PerformRecordingRequest

Parameters for PCapture Request

struct

UUID	**UUID**

40.3.24 PerformRecordingResponse

Parameters for PCapture Response

struct

status	**Status**
urlToPcap	**string**

40.3.25 Status

```
RecorderStatus

@param SUCCESS:            Indicates that the recording has been successfully performed.
@param UUID_NOT_FOUND:     The specified input UUID was not present in the database. Ensure that
```

```
@param BUSY:               Only one recording can be performed at a time. Please wait until the
@param UNEXPECTED_FAILURE: Some internal error occured which caused the recording to fail. Pleas
```

enum

SUCCESS
UUID_NOT_FOUND
BUSY
UNEXPECTED

41 physicalIpPort (1.0)

41.1 Type Reference

41.1.1 PortName

enum

CTRL
D1
D2
D3
D4
D1_1
D1_2
D1_3
D1_4
D2_1
D2_2
D2_3
D2_4
D3_1
D3_2
D3_3
D3_4
D4_1
D4_2
D4_3
D4_4

42 physicalports (1.16)

42.1 Overview

Changelog:

1.16

- **Changed**
- Added an autoNegotiation module to control auto-negotiation settings on physical ports D1 and D2 on X5 product.

1.15

- **Changed**
  - Changed fields 'rx' and 'tx' in struct 'CurrentLinkRates' from int to bigint

1.14

- **Added**
  - L2 channel bonding
    - added field 'channelBonding' to `VirtualPort`, new enum `ChannelBondingMode`
    - added RPC method `GetVirtualPortsStatus` and `VirtualPortStatus` type
- **Changed**
  - linkModes bumped to 1.5 which removes 100G COPPER as option for ipLinkMode

1.13

- **Added**
  - The field `ipLinkMode` was added to `PhysicalPort` to support configuration of data port link speed for the 100G product
  - The field `portFecMode` was added to `PhysicalPort` to support FEC reconfiguration on the data ports for the 100G product
  - The enum `PortMode` was expanded with member QSFP to be used for configure the data ports on the 100G product

1.12

- **Changed**
  - interfaceToPortName now takes in "CTRL", "D1", "D2", "D3" and "D4" instead of ni6, ni0, ni1, ni4 and ni5.
  - portNameToInterface now returns "CTRL", "D1", "D2", "D3" and "D4" instead of ni6, ni0, ni1, ni4 and ni5
- **Removed**
  - The ifToDisplayName function has been removed.
- **Added**
  - portNameToLinuxInterface mapping has been added, which takes in a physicalport port-name, and returns the relevant linux interface name as a string. Not applicable for 1G DSNG.
  - portNameToLinuxInterface1Gcard mapping has been added, which takes in a physical port name, and returns the relevant linux interface name as a string. Only applicable on the 1G DSNG.

1.11

- **Added**
  - The field `replicateLinkState` has been added to `VirtualPort` to support link state replication on mated physical port

1.10

1.9

- **Added**
  - The RPC GetCurrentLinkRates has been added, which provides the RX/TX rates on the data ports.
  - The RPC GetPortStatus has been added, which provides RX/TX rates on data ports, and the amount of bits sent/received.
- **Changed**
  - The RPC GetLinkSpeed has been renamed to GetMaxLinkRates to clarify what it represents.
  - The corresponding enum LinkSpeed has been renamed to LinkRate. Same for the string converters.
  - The mapping linkSpeedToString has been renamed to linkRateToString.

1.8

- **Added**
  - The field `bridge` was added to `PhysicalPort` to support enabling a VLAN bridge.
  - The field `helloInterval` was added to OSPF.
- **Changed**
  - The field `md5Auth` in OSPF was changed to contain two optional `MD5AuthKeyPairs`. This makes it possible to specify only one at a time.
- **Removed**
  - The fields `ifname` and `switchDelays` were removed from `PhysicalPort`.
  - The fields `allowCloned`, `allowSeamless`, and 'label' were removed from `VirtualPort` after being marked as 'not used' for multiple versions.

1.7

- **Added**
  - The RPC `GetSfpStatus` was added.
  - The `PhysicalPort` object got new fields `enabled` and `mtuSize`.
- **Changed**
  - The `lldp` related concepts were relocated to lldp.1.0.
  - The `PhysicalPort.lldp` field changed type from `optional(LldpMode)` to `lldp.1.0.LldpMode`.

1.6

- **Added**
  - The RPC `GetLldpNeighbors` was added.
  - The field `lldp` was added to the `PhysicalPort` object.

1.5

- **Added**
  - The `GetMacAddresses` RPC was added.

1.4

- **Added**
  - The `VirtualPort` object got a new field, `exclusiveOutput`.

1.3

- **Added**
  - The fields `name` and `monitorInOut` were added to the `PhysicalPort` object.

1.2

- **Added**
  - The fields `rx`, `tx`, `ospf`, `pim`, `muteOnError` and `switchDelays` were added to the `PhysicalPort` object.
- **Changed**
  - The `LinkSpeed` enum was expanded with members `MBPS_10`, `MBPS_100`, `GBPS_25_40` and `NO_LINK`
- **Removed**
  - The `LinkSpeed` enum got its `UNKNOWN` member removed

42.2 Command Reference

42.2.1 GetPhysicalPorts

- **message GetPhysicalPorts.Request**
- **message GetPhysicalPorts.Response**

42.2.2 SetPhysicalPorts

- **message SetPhysicalPorts.Request**
- **message SetPhysicalPorts.Response**

42.2.3 DeletePhysicalPorts

- **message DeletePhysicalPorts.Request**
- **message DeletePhysicalPorts.Response**

42.2.4 GetVirtualPorts

- **message GetVirtualPorts.Request**
- **message GetVirtualPorts.Response**

42.2.5 GetVirtualPortsStatus

- **message GetVirtualPortsStatus.Request**
- **message GetVirtualPortsStatus.Response**

42.2.6 SetVirtualPorts

- **message SetVirtualPorts.Request**
- **message SetVirtualPorts.Response**

42.2.7 DeleteVirtualPorts

Delete virtual ports

- **message DeleteVirtualPorts.Request**
- **message DeleteVirtualPorts.Response**

42.2.8 GetMaxLinkRates

Get max link rates. Returns the (theoretical) maximum rate supported per port.

- **message GetMaxLinkRates.Request**
- **message GetMaxLinkRates.Response**

42.2.9 GetMacAddresses

Get MAC Addresses.

Returns the MAC addresses that are set on each port.

- **message GetMacAddresses.Request**
- **message GetMacAddresses.Response**

42.2.10 GetLldpNeighbors

Get LLDP Neighbors.

Returns data about the LLDP neighbor seen per port. To see data for a specific port, LLDP must be enabled for that PhysicalPort.

- **message GetLldpNeighbors.Request**
- **message GetLldpNeighbors.Response**

42.2.11 GetSfpStatus

Get SFP Status.

Get status and diagnostics information reported from the SFP plug for applicable ports (only applicable for optical ports).

- **message GetSfpStatus.Request**
- **message GetSfpStatus.Response**

42.2.12 GetCurrentLinkRates

- **message GetCurrentLinkRates.Request**
- **message GetCurrentLinkRates.Response**

42.3 Type Reference

42.3.1 BondingActivePort

Which port in the bond is currently active. Not applicable if Bonding is disabled or both interfaces are down.

enum

`PORT_A`	
`PORT_B`	

42.3.2 Bridge

Bridge from the specified 10G port. Traffic with the specified VLAN tag will be directly forwarded to this physical port.

struct

`port`	**PortName** Which of the 10G ports (D1 or D2) the bridge should be to.
`vlanTag`	**int** Only traffic with this vlanTag is forwarded over the bridge.

42.3.3 ChannelBondingMode

Represents different modes of operation for L2 channel bonding, where ports A and B are logically bonded into a single port.

- When L2 channel bonding is enabled, ports A and B are treated as a single bonded port.
- In most applications, the bonded port is identified as port A, and port B can't be used, e.g. input or outputs can only use port A (the bond) but not port B.
- Most settings applied to port A will be applied to the whole bond, i.e. also to port B.
- However, certain settings on port B can be changed even when it's bonded with port A (e.g. whether to use RJ45/SFP/SFPP/QSFP).
- The bonding logic dynamically selects whether port A or B is the active port.

enum

`UNKNOWN`	Reserved to indicate modes from future versions of API that has no corresponding values in this version.
`DISABLED`	Bonding is disabled; ports A and B of the port pair can be used separately.
`ACTIVE_BACKUP`	Bonding is enabled in active-backup mode; port A in the port pair is the *primary* port, and port B is the *backup*. The primary reselection policy is set to *failure* (aka "floating").
`FORCE_PORT_A`	Bonding is enabled, but port A is always the active port. Useful for testing connectivity.
`FORCE_PORT_B`	Bonding is enabled, but port B is always the active port. Useful for testing connectivity.

42.3.4 ChannelBondingStatus

struct

activePort	**BondingActivePort** Indicates the currently active port in the bond.
switchCount	**int** Number of switch occurrences between ports A and B.

42.3.5 CurrentLinkRates

struct

rx	**bigint**
tx	**bigint**

42.3.6 CurrentLinkRatesMap

map from **UUID** to **CurrentLinkRates**

42.3.7 DeletePhysicalPorts.Request

struct

ids	**list** of **UUID**

42.3.8 DeletePhysicalPorts.Response

empty **struct**

42.3.9 DeleteVirtualPorts.Request

struct

ids	**list** of **UUID**

42.3.10 DeleteVirtualPorts.Response

empty **struct**

42.3.11 GetCurrentLinkRates.Request

GetCurrentLinkRatesRequest

42.3.12 GetCurrentLinkRates.Response

GetCurrentLinkRatesResponse

42.3.13 GetCurrentLinkRatesRequest

empty **struct**

42.3.14 GetCurrentLinkRatesResponse

struct

data	**CurrentLinkRatesMap**

42.3.15 GetLldpNeighbors.Request

GetLldpNeighborsRequest

42.3.16 GetLldpNeighbors.Response

GetLldpNeighborsResponse

42.3.17 GetLldpNeighborsRequest

Get LLDP Neighbors Request

struct

physicalports	**optional list** of **UUID** List of UUIDs uniquely identifying PhysicalPorts. Only LLDP Neighbors from the specified PhysicalPorts are returned. If no value is set, the request will return neighbors from all ports that have LLDP enabled.
details	**bool** If false only chassisID, mgmtIP, and portID will be returned for each LLDP Neighbor. If true all available information is returned.

42.3.18 GetLldpNeighborsResponse

struct

data	**map** from **UUID** to **optional lldp.LldpNeighbor**

42.3.19 GetMacAddresses.Request

GetMacAddressesRequest

42.3.20 GetMacAddresses.Response

GetMacAddressesResponse

42.3.21 GetMacAddressesRequest

empty **struct**

42.3.22 GetMacAddressesResponse

struct

data	**map** from **UUID** to **string**

42.3.23 GetMaxLinkRates.Request

GetMaxLinkRatesRequest

42.3.24 GetMaxLinkRates.Response

GetMaxLinkRatesResponse

42.3.25 GetMaxLinkRatesRequest

empty **struct**

42.3.26 GetMaxLinkRatesResponse

struct

data	**map** from **UUID** to **linkModes.IpLinkSpeed**

42.3.27 GetPhysicalPorts.Request

empty **struct**

42.3.28 GetPhysicalPorts.Response

struct

data	**map** from **UUID** to **PhysicalPort**
info	**firewallTypes.PaginationInfo**

42.3.29 GetPortStatusRequest

empty **struct**

42.3.30 GetPortStatusResponse

struct

data	**PortStatusMap**

42.3.31 GetSfpStatus.Request

GetSfpStatusRequest

42.3.32 GetSfpStatus.Response

GetSfpStatusResponse

42.3.33 GetSfpStatusRequest

Get SFP Status Request

struct

physicalports	**optional list** of **UUID** List of UUIDs uniquely identifying PhysicalPorts. Only SFP Status from the specified PhysicalPorts are returned. If no value is set, the request will return SFP Status from all ports.

42.3.34 GetSfpStatusResponse

struct

data	**map** from **UUID** to **optional** **sfpStatus.SfpStatus**

42.3.35 GetVirtualPorts.Request

empty **struct**

42.3.36 GetVirtualPorts.Response

struct

data	**map** from **UUID** to **VirtualPort**
info	**firewallTypes.PaginationInfo**

42.3.37 GetVirtualPortsStatus.Request

empty **struct**

42.3.38 GetVirtualPortsStatus.Response

GetVirtualPortsStatusResponse

42.3.39 GetVirtualPortsStatusResponse

struct

data	**map** from **UUID** to **VirtualPortStatus**

42.3.40 MD5Auth

struct

keyPair1	**optional** **MD5AuthKeyPair**
keyPair2	**optional** **MD5AuthKeyPair**

42.3.41 MD5AuthKeyPair

struct

keyId	**int**
key	**string**

42.3.42 MonitorInOut

struct

monitoredPortId	**UUID**

42.3.43 OSPF

struct

ospfArea	**string**
routerId	**optional string**
md5Auth	**optional MD5Auth**
stubbyArea	**bool**
ospfMetric	**int**
helloInterval	**int**

42.3.44 PIM

struct

rpPointA	**string**
rpPointB	**string**

42.3.45 PhysicalPort

```
Settings for physical ports on a card.
PIM, muteOnError and switchDelays are not supported in version <= 1.2

@param name            Name of the PhysicalPort (on-chassis printed)
@param label           Custom label to describe the portMode
@param enabled         Enable or disable the interface port. Setting this value to false will ta
@param mtuSize         Optionally set maximum transmission unit (MTU) size. It is the largest am
@param portMode        Which mode or direction the port should be in
@param ipLinkMode      Optionally set the link speed for the data ports. Only applicable for 100
@param portFecMode     Optionally set the FEC mode for the data ports. Only applicable for 100G
@param autoNegotiation Optionally set Auto-Negotiation settings for physical ports. Only applica
@param lldp            The LldpMode to set for the port. Leaving this empty indicates LLDP off.
@param rx              Enable or disable RX.
@param tx              Enable or disable TX.
@param bridge          Optional bridge to a VLAN on one of the 10G ports (D1 or D2), this is onl
@param ospf            Optionally enable OSPF (Output Redundancy). Not supported on X5.
@param pim             Optionally enable PIM (Output Redundancy). Not supported on X5.
@param muteOnError     Optionally enable muting on error (Output Redundancy). Not supported on X
@param monitorInOut    Optionally enable monitor input/output (Output Redundancy). It monitors o
```

struct

name	PortName
label	string
enabled	bool
mtuSize	optional int
portMode	PortMode
ipLinkMode	optional linkModes.IpLinkMode
portFecMode	optional linkModes.IpFecMode
autoNegotiation	optional autoNegotiation.AutoNegotiation
lldp	optional lldp.LldpMode
rx	bool
tx	bool
bridge	optional Bridge
ospf	optional OSPF
pim	optional PIM
muteOnError	bool
monitorInOut	optional MonitorInOut

42.3.46 PortMode

enum

RJ45
SFPP
SFP
QSFP

42.3.47 PortName

enum

CTRL
D1
D2
D3
D4

42.3.48 PortStatus

struct

currentRxRate	bigint
currentTxRate	bigint
totalRxBits	bigint
totalTxBits	bigint

42.3.49 PortStatusMap

map from UUID to PortStatus

42.3.50 SetPhysicalPorts.Request

struct

`data`	**map** from **UUID** to **PhysicalPort**
	The key in this map uniquely identifies a PhysicalPort object in the system. It is required to be deterministic based on given parameters in the PhysicalPort object in the following manner:
	`UUID5(constants::0.0/BASE_UUID_NAMESPACE, string(PhysicalPort.name))`
	Example:
	`UUID5(constants::0.0/BASE_UUID_NAMESPACE, "D1")`

42.3.51 SetPhysicalPorts.Response

empty **struct**

42.3.52 SetVirtualPorts.Request

struct

`data`	**map** from **UUID** to **VirtualPort**
	The key in this map uniquely identifies a VirtualPort object in the system. It is required to be deterministic based on given parameters in the VirtualPort object in the following manner:
	`UUID5(constants::0.0/BASE_UUID_NAMESPACE, string(uuid_of_port_A) + "." + string(uuid_of_port_b))`
	Example:
	`UUID5(constants::0.0/BASE_UUID_NAMESPACE, "212c0d7e-9128-54e2-aab5-40ac002052ac.20a021e7-0e36-580f-952f-b06`

42.3.53 SetVirtualPorts.Response

empty **struct**

42.3.54 VirtualPort

VirtualPort is a pair of PhysicalPorts, with their common properties defined.

struct

`physicalPortA`	**UUID** uuid for physicalPortA
`physicalPortB`	**UUID** uuid for physicalPortB
`exclusiveOutput`	**bool** enable/disable exclusive output for pair of physicalports. Not supported on X5.
`replicateLinkState`	**bool** enable/disable link state replication for pair of physical ports. Not supported on X5.
	ChannelBondingMode

`channelBonding`	enable/disable L2 channel bonding on this pair of physical ports, supported by SRT modules

42.3.55 VirtualPortStatus

struct

`bonding`	**optional ChannelBondingStatus** Channel bonding status. Will have a value if channel bonding is enabled, otherwise empty optional.

43 services (1.3)

43.1 Command Reference

43.1.1 GetInputServices

Returns service information for the queried list of service node IDs. By default returns only services which are currently present on inputs. Can be provided a query object which can change default behavior.

- **message GetInputServices.Request**
- **message GetInputServices.Response**

43.1.2 GetInputServicesPage

- **message GetInputServicesPage.Request**
- **message GetInputServicesPage.Response**

43.1.3 GetOutputServices

Returns service information for output services

- **message GetOutputServices.Request**
- **message GetOutputServices.Response**

43.1.4 GetServiceDependencies

Returns service dependency information

- **message GetServiceDependencies.Request**
- **message GetServiceDependencies.Response**

43.2 Type Reference

43.2.1 ChildSourceInfo

struct

name	**string**
nodeType	**string**
body	**string**
owner	**string**
isPresent	**bool**

43.2.2 FlowSinkInfo

struct

name	**string**
label	**string**
nodeType	**string**
body	**string**

sources	map from UUID to FlowSourceServiceInfo
owner	string

43.2.3 FlowSourceInfo

struct

name	string
label	string
nodeType	string
body	string
owner	string
isPresent	bool
sources	map from UUID to ChildSourceInfo

43.2.4 FlowSourcePage

struct

result	pagination.PaginatedQueryResult
data	map from UUID to FlowSourceInfo

43.2.5 FlowSourceServiceInfo

ChildSourceInfo

43.2.6 GetInputServices.Request

GetInputServicesRequest

43.2.7 GetInputServices.Response

struct

data	map from UUID to FlowSourceInfo

43.2.8 GetInputServicesPage.Request

GetInputServicesPageRequest

43.2.9 GetInputServicesPage.Response

FlowSourcePage

43.2.10 GetInputServicesPageRequest

struct

slice	pagination.DataSetSlice
query	optional InputServiceQuery

43.2.11 GetInputServicesRequest

struct

query	optional InputServiceQuery

43.2.12 GetOutputServices.Request

empty struct

43.2.13 GetOutputServices.Response

struct

data	map from UUID to FlowSinkInfo

43.2.14 GetServiceDependencies.Request

empty struct

43.2.15 GetServiceDependencies.Response

struct

data	list of ServiceDependencyInfo

43.2.16 InputServicePresenceQuery

enum

ALL
ONLY_PRESENT
ONLY_MISSING

43.2.17 InputServiceQuery

struct

servicePresence	InputServicePresenceQuery

43.2.18 ServiceDependencyInfo

struct

source	UUID
dependency	UUID

44 sfpConstants (1.1)

44.1 Type Reference

44.1.1 Identifier

Physical Device Identifier Values. (SFF 8472 Rev12.3 spec Table 5-1)

enum

UNKNOWN	Unknown or unspecified
GBIC	GBIC
SOLDERED	Module soldered to motherboard (ex. SFF)
SFP_SFPP	SFP or SFP+
NOT_USED	Not used by SFF 8472 Rev12.2.1. These values are maintained in the Transceiver Management section of SFF-8024.
VENDOR	Vendor specific

44.1.2 Transceiver

Transceiver Compliance Codes. (SFF 8472 Rev12.3 spec Table 5-3)

enum

ER_10G	10B Base-ER
LRM_10G	10B Base-LRM
LR_10G	10B Base-LR
SR_10G	10B Base-SR
PX	BASE-PX
BX10	BASE-BX10
FX_100	100BASE-FX
LX_100	100BASE-LX/LX10
T_1000	1000BASE-T
CX_1000	1000BASE-CX
LX_1000	1000BASE-LX
SX_1000	1000BASE-SX
ACTIVE_CABLE	Active Cable
PASSIVE_CABLE	Passive Cable

45 sfpStatus (1.2)

45.1 Type Reference

45.1.1 SfpDiagnostics

Real Time Diagnostic for SFP/SFP+, with A2h memory space address in square brackets. (SFF 8472 Rev12.3 spec Table 9-11)

struct

`temp`	**float** Internally measured module temperature. [96-97]
`vcc`	**float** Internally measured supply voltage in transceiver. [98-99]
`txPwr`	**float** Measured TX output power. [102-103]
`rxPwr`	**float** Measured RX input power. [104-105]

45.1.2 SfpStatus

SFP Status Data Fields, with A0h memory space address in square brackets. (SFF 8472 Rev12.3 spec Table 4-1) We currently only support SFP/SFP+ status, but support for QSFP might be added at a later point.

struct

`identifier`	**sfpConstants.Identifier** Type of transceiver (SFP/SFP+ or unknown). [0]
`transceiver`	**list** of **sfpConstants.Transceiver** Code for electronic or optical compatibility [3-10]
`brNominal`	**float** Nominal signalling rate, units of 100 MBd. [12]
`vendorName`	**string** SFP vendor name. [20-35]
`vendorOUI`	**string** SFP vendor IEEE company ID. [37-39]
`vendorPN`	**string** Part number provided by SFP vendor. [40-55]
`vendorRev`	**string** Revision level for part number provided by vendor. [56-59]
`wavelength`	**string** Laser wavelength. [60-61]
`vendorSN`	**string** Serial number provided by SFP vendor. [68-83]
`dateCode`	**string** Vendor's manufacturing date code. [84-91]
`diagnostics`	**optional SfpDiagnostics** Optional Diagnostics Monitor Data. Only for SFP/SFP+.

46 srt (1.7)

46.1 Overview

Changelog:

1.7

- **Added**
  - Made new transport mode SrtBroadcastTransport
  - Added srtBroadcastBonding to SrtInputTransport and SrtOutputTransport
  - Removed seamlessBufferSize from SrtSeamlessTransport

1.6

- **Added**
  - Added 'maxNumberOfConnections' to SrtOutputSettings
  - Added 'rejectedPeers' and 'acceptedPeers' to SrtListener
  - Added 'streamIdFilter' to SrtListener
- **Changed**
  - Changed max overhead bandwidth from 100
  - Changed min latency from 60ms to 1ms
  - Moved streamId from SrtOutputSettings and SrtInputSettings to SrtCaller

1.5

- **Added**
  - Added 'trafficPolicing' to both SrtOutputSettings and SrtInputSettings

1.4

- **Added**
  - Added 'highBitrate' to SrtOutputSettings and SrtInputSettings

1.3

- **Added**
  - Added 'streamId' to SrtOutputSettings and SrtInputSettings
- **Changed**
  - Option 'AES_192' has been removed from SrtAlgorithm
  - Port parameters in SrtCaller have changed names. It now has the same meaning independent of input / output.
  - Port parameter in SrtListener also have changed name for consistency with caller.

1.2

- **Added**
  - Added field 'rtp' to type SrtOutputSettings

1.1

- **Added**

- Added the field 'transport' to types SrtOutputSettings and SrtInputSettings
- Added the types SrtInputTransport, SrOutputTransport, SrtSeamlessTransport, SrtCloned-Transport and SrtTransportSettings

- **Moved**
  - Moved the fields 'interfaceId' and 'mode' from SrtOutputSettings and SrtInputSettings to SrtTransportSettings

46.2 Type Reference

46.2.1 SrtAlgorithm

SRT Encryption Algorithm

enum

AES_128	AES with 128-bit key
AES_256	AES with 256-bit key

46.2.2 SrtBroadcastTransport

SRT Broadcast Transport

struct

pathA	**SrtTransportSettings** Broadcast Path A
pathB	**SrtTransportSettings** Broadcast Path B

46.2.3 SrtCaller

SRT Caller Mode

struct

address	**string** IP address of remote / peer
localPort	**int** Local UDP port
remotePort	**int** Remote / peer UDP port
streamId	**optional string** SRT Stream ID

46.2.4 SrtClonedTransport

SRT Cloned Transport

struct

SrtTransportSettings

pathA	Cloned Path A
pathB	**SrtTransportSettings** Cloned Path B

46.2.5 SrtEncryption

SRT Encryption

struct

algorithm	**SrtAlgorithm** Encryption algorithm
key	**string** Encryption key as a hex string

46.2.6 SrtInputSettings

SRT Input Settings

struct

transport	**SrtInputTransport** SRT input transport settings.
decryption	**optional SrtEncryption** Decryption settings. Not applicable for listener mode.
receiveLatency	**int** Receive latency in milliseconds.
highBitrate	**bool** Enabling high bitrate (HBR) mode will allow the stream to use higher maximum bitrate.
trafficPolicing	**optional int** Traffic policing of incoming data at given data rate (in kbps). If the input data rate is above the configured value, then the input is muted.

46.2.7 SrtInputTransport

SRT Input Transport

Seamless or single input

variant

single	**SrtTransportSettings** Single input
seamless	**SrtSeamlessTransport** Seamless input
srtBroadcast	**SrtBroadcastTransport** Broadcast input

46.2.8 SrtListener

SRT Listener Mode

struct

`localPort`	**int** Local UDP port
`rejectedPeers`	**optional list** of **string** List of rejected peers, using CIDR notation
`acceptedPeers`	**optional list** of **string** List of accepted peers, using CIDR notation
`streamIdFilter`	**optional string** Incoming connections will be rejected if their stream ID does not matches this value

46.2.9 SrtMode

SRT Mode

variant

`caller`	**SrtCaller** Caller mode
`listener`	**SrtListener** Listener mode
`rendezvous`	**SrtRendezvous** Rendezvous mode

46.2.10 SrtOutputSettings

SRT Output Settings

struct

`transport`	**SrtOutputTransport** SRT output transport settings
`tsPacketsPerFrame`	**int** Number of TS packets per IP frame [1-7]
`encryption`	**optional SrtEncryption** Encryption settings. Not applicable for listener mode.
`receiveLatency`	**int** Receive latency in milliseconds
`overheadBandwidth`	**int** Overhead bandwidth in per cent
`rtp`	**bool** Send TS with RTP headers (needed if received as 2022-7)
`highBitrate`	**bool** Enabling high bitrate (HBR) mode will allow the stream to use higher maximum bitrate.
	optional int

`trafficPolicing`	Traffic policing of outgoing data at given data rate (in kbps). If the source data rate is above the configured value, then the output is muted.
`maxNumberOfConnections`	**optional int** Maximum number of connections that can be established, only applicable for listener mode.

46.2.11 SrtOutputTransport

SRT Output Transport

Cloned or single output

variant

`single`	**SrtTransportSettings** Single output
`cloned`	**SrtClonedTransport** Cloned output
`srtBroadcast`	**SrtBroadcastTransport** Broadcast output

46.2.12 SrtRendezvous

SRT Rendezvous Mode

struct

`address`	**string** IP address of remote / peer
`port`	**int** UDP port (same for both local and remote)

46.2.13 SrtSeamlessTransport

SRT Seamless Transport

struct

`pathA`	**SrtTransportSettings** Seamless Path A
`pathB`	**SrtTransportSettings** Seamless Path B
`identical`	**bool** Means the expected sources are 2022-7 compliant
`preferred`	**SrtSeamlessTransport.preferred** Preferred pipe

46.2.14 SrtSeamlessTransport.preferred

enum

A	
B	
FLOATING	

46.2.15 SrtTransportSettings

SRT Transport Settings

struct

interfaceId	**UUID** UUID of the network interface
mode	**SrtMode** SRT mode

47 srtCardSettings (1.0)

47.1 Command Reference

47.1.1 GetCardSettings

- **message GetCardSettings.Request**
- **message GetCardSettings.Response**

47.1.2 SetCardSettings

- **message SetCardSettings.Request**
- **message SetCardSettings.Response**

47.2 Type Reference

47.2.1 CardSettings

Gloabl settings for the SRT module

struct

`maxSrtLatency`	**MaxSrtLatency** Global maximum SRT latency. Higher maximum SRT latency will reduce the maximum total bitrate, and also the maximum bitrate per stream.

47.2.2 GetCardSettings.Request

empty **struct**

47.2.3 GetCardSettings.Response

struct

`config`	**CardSettings**

47.2.4 MaxSrtLatency

Maximum SRT latency

enum

`SECONDS_1`	Maximum 1 second SRT latency
`SECONDS_2`	Maximum 2 second SRT latency
`SECONDS_4`	Maximum 4 second SRT latency
`SECONDS_6`	Maximum 6 second SRT latency
`SECONDS_8`	Maximum 8 second SRT latency

47.2.5 SetCardSettings.Request

struct

config	**CardSettings**

47.2.6 SetCardSettings.Response

empty **struct**

48 srtValidate (1.1)

48.1 Type Reference

48.1.1 FloatRange

Float Range

struct

`min`	**float** Minimum value of the range
`max`	**float** Maximum value of the range

48.1.2 IntRange

Integer Range

struct

`min`	**int** Minimum value of the range
`max`	**int** Maximum value of the range

48.1.3 Minimum

Minimum

struct

`min`	**int** Minimum value of the range

48.1.4 Result

Validation result

struct

`result`	**bool** True if config is valid and false otherwise.
`msg`	**optional string** An error message, which may be given if the result is invalid.

49 status (1.25)

49.1 Overview

Changelog:

1.25

- **Added**
  - Added 'localGroupId' to 'SrtConnectionStatus'
  - Added 'SRT' into SeamlessProtocol enum

1.24

- **Added**
  - added srtSocketId to 'SrtConnectionStatus'
  - Renamed previous 'SrtOutputStatus' to 'SrtOutputConnectionStatusPair' and created a new SrtOutputStatus with list of 'SrtOutputStatusPair', 'activeConnections' and 'connectionCount-Total'

1.23

- **Added**
  - The field 'rtpErrors' was added to 'OutputStatus'

1.22

- **Added**
  - Additional SRT connection status fields; 'peerEndpoint', 'peerStreamId', 'srtLatency', and 'up-time'.
- **Changed**
  - 'SrtSocketState' has reduced number of enum values, removing unused values. Values are also renamed.
  - Formats of 'SrtInputConnectionStatus' and 'SrtOutputConnectionStatus' are slightly changed, moving connection related values into common type 'SrtConnectionStatus'

1.21

- **Added**
  - The field 'bufferDelay' was added to the 'SeamlessPortStatus' type.

1.20

- **Removed**
  - Rpc 'GetOutputPidStatus' has been moved to tsstatus, and thus has been removed from this file.

1.19

- **Added**
  - The type 'IpSourceAddressStatus' where added.
- **Changed**
  - 'ipSourceAddressStatus' optional was added on type 'IpInputStatus'.

1.18

- **Added**
  - The field 'totalFlowBitrateWithoutFec' was added to 'IpInputStatus'
- **Changed**
  - 'totalFlowBitrate' was changed from float to bigint in 'IpInputStatus', and the unit changed from Kbps to bps.
  - 'effectiveFlowBitrate' was changed from float to bigint in 'IpInputStatus', and the unit changed from Kbps to bps.
  - 'bitrate' was changed from float to bigint in 'SeamlessPortStatus', and the unit changed from Kbps to bps.

1.17

- **Changed**
  - 'sequenceErrors' was changed from int to bigint in 'SeamlessPortStatus'
  - 'rtpErrors' was changed from int to bigint in 'IpInputStatus'

1.16

- **Added**
  - Add `GetOutputPidStatus` RPC
  - The types 'ZixiInputPathStatus' and 'SrtInputPathStatus' were added.
  - The fields 'inbandLinkStatus' and 'zixiStatus'/'srtStatus' were moved into the field 'path1' on types 'ZixiInputStatus' and 'SrtInputStatus'.
  - The fields 'rtpErrors', 'syncByteError', 'teiBitError', 'dejitterStatus' and 'seamlessStatus' were added to types 'ZixiInputStatus' and 'SrtInputStatus'.
  - The field 'srtStatus' was renamed to 'path1' on type 'SrtOutputStatus'.
  - The field 'zixiStatus' was renamed to 'path1' on type 'ZixiOutputStatus'.
  - The field 'path2' was added to types 'ZixiOutputStatus', 'SrtOutputStatus', 'ZixiInputStatus' and 'SrtInputStatus'.
- **Changed**
  - The imported version of `tsstatus` was bumped to 1.3

1.15

- **Added**
  - The field 'IpOutputBitrates' was added to OutputStatus.

1.14

- **Added**
  - The field 'totalFlowBitrate' and 'totalFlowBitsReceived' was added to IpInputStatus
- **Changed**
  - The field `InbandLinkStatus.statusCcErrors` has been renamed to `InbandLinkStatus.seqErrorCount`

1.13

- **Added**
  - The field `alternativeStatuses` was added to `OutputSourceStatus`.
- **Changed**

- The imported version of `tsstatus` was bumped to 1.1.

  The fields `tsId` and `originalNetworkId` were added to `TransportStreamStatus`.

1.12

- **Added**
  - The fields `mainBufferDelay`, `backupBufferDelay`, and `currentAlternativeIdx` were added to `OutputSourceStatus`.

49.2 Command Reference

49.2.1 GetIpInputStatus

- **message GetIpInputStatus.Request**
- **message GetIpInputStatus.Response**

49.2.2 ClearAllCounters

- **message ClearAllCounters.Request**
- **message ClearAllCounters.Response**

49.2.3 GetPidStatus

Get Pid status for a set of inputs.

- **message GetPidStatus.Request**
- **message GetPidStatus.Response**

49.2.4 GetZixiInputStatus

Get zixi status for a set of inputs.

- **message GetZixiInputStatus.Request**
- **message GetZixiInputStatus.Response**

49.2.5 GetZixiOutputStatus

Get zixi status for a set of outputs.

- **message GetZixiOutputStatus.Request**
- **message GetZixiOutputStatus.Response**

49.2.6 GetSrtInputStatus

- **message GetSrtInputStatus.Request**
- **message GetSrtInputStatus.Response**

49.2.7 GetSrtOutputStatus

- **message GetSrtOutputStatus.Request**
- **message GetSrtOutputStatus.Response**

49.2.8 GetOutputStatus

- **message GetOutputStatus.Request**
- **message GetOutputStatus.Response**

49.3 Type Reference

49.3.1 AlternativeStatus

Input Redundancy alternative status

struct

`isBroken`	**bool** indicates status of an alternative

49.3.2 CbrRegulationStage

Enum to hold possible stages of CBR dejittering algorithm

enum

`INIT_FILTER_STAGE`
`FIRST_FILTER_STAGE`
`SECOND_FILTER_STAGE`
`REGULATION_STAGE`

49.3.3 ClearAllCounters.Request

empty **struct**

49.3.4 ClearAllCounters.Response

empty **struct**

49.3.5 DejitterStatus

Type to hold status information for input dejittering

struct

`mode`	**DejitteringMode**
`stage`	**optional CbrRegulationStage**
`maxJitter`	**float**
`minDelay`	**float**
`resetCounter`	**int**

49.3.6 DejitteringMode

Enum to hold possible dejittering modes, status for PCR dejittering is not supported.

enum

CBR	
RTP	
PCR	
OFF	

49.3.7 FECStatus

struct

recived	**bool**
enabled	**bool**
fecType	**FECType**
columns	**int**
rows	**int**
invalidParams	**bool**
decTimeout	**bool**
delayToLow	**bool**
unrecvBurstLoss	**bool**
columnRate	**int**
rowRate	**int**
unrecoverable	**int**
recovered	**int**
sequenceErrors	**int**
recommendedBufferSize	**int**
totalBitrateWithFEC	**float**

49.3.8 FECType

Fec type A Column only, Fec type B Row and Column

enum

A	
B	

49.3.9 GetIpInputStatus.Request

IpStatusReq

49.3.10 GetIpInputStatus.Response

IpStatusRes

49.3.11 GetOutputStatus.Request

empty **struct**

49.3.12 GetOutputStatus.Response

GetOutputStatusResponse

49.3.13 GetOutputStatusResponse

struct

data	map from UUID to OutputStatus

49.3.14 GetPidStatus.Request

PidStatusRequest

49.3.15 GetPidStatus.Response

PidStatusResponse

49.3.16 GetSrtInputStatus.Request

SrtStatusReq

49.3.17 GetSrtInputStatus.Response

SrtStatusInputRes

49.3.18 GetSrtOutputStatus.Request

SrtStatusReq

49.3.19 GetSrtOutputStatus.Response

SrtStatusOutputRes

49.3.20 GetZixiInputStatus.Request

ZixiInputStatusReq

49.3.21 GetZixiInputStatus.Response

ZixiInputStatusRes

49.3.22 GetZixiOutputStatus.Request

ZixiOutputStatusReq

49.3.23 GetZixiOutputStatus.Response

ZixiOutputStatusRes

49.3.24 InbandLinkStatus

In-band Link Status

struct

sourceHostname	**string** Hostname of source module
sourceLabel	**string** User label of source flow
delay	**optional float** End-to-end packet delay (ms), 0.032ms resolution up to 2.115 s
jitter	**float** Packet jitter (us), 12.8 us resolution up to 0.84 s
seqErrorCount	**bigint** Number of sequential errors for the status packets
timeSourceSender	**TimeSource** Time source used on the sender
timeSourceReceiver	**TimeSource** Time source used on the receiver

49.3.25 IpInbandLinkStatus

In-band Link Status for IP Inputs

struct

path1	**optional InbandLinkStatus** Status for single input, or status for input path 1 if input is configured as seamless.
path2	**optional InbandLinkStatus** Status for input path 2 if input is configured as seamless.

49.3.26 IpInputBitrates

struct

totalFlowBitrate	**bigint**
totalFlowBitsReceived	**bigint**
effectiveFlowBitrate	**bigint**
effectiveFlowBitsReceived	**bigint**
totalFlowWithoutFecBitrate	**bigint**
totalFlowWithoutFecBitsReceive	**bigint**

49.3.27 IpInputStatus

struct

hasBitrate	**bool**
bitrates	**optional IpInputBitrates**
rtpErrors	**optional bigint**

ccErrors	optional int
syncByteError	optional int
teiBitError	optional int
services	list of ServiceSourceInfo
fecStatus	optional FECStatus
seamlessStatus	optional SeamlessStatus
dejitterStatus	optional DejitterStatus
inbandLinkStatus	IpInbandLinkStatus
ipSourceAddressStatus	IpSourceAddressStatus

49.3.28 IpOutputBitrates

struct

total	IpOutputTotalBitrates
effective	IpOutputEffectiveBitrates

49.3.29 IpOutputEffectiveBitrates

struct

effectiveFlowBitrate	float
effectiveFlowBitsSent	bigint

49.3.30 IpOutputTotalBitrates

struct

totalFlowBitrate	float
totalFlowBitsSent	bigint

49.3.31 IpSeamlessSourceStatus

Seamless ip source status

struct

a	optional IpSourceStatus
b	optional IpSourceStatus

49.3.32 IpSourceAddressStatus

Seamless or Single ip source status

variant

seamless	IpSeamlessSourceStatus
single	optional IpSourceStatus

49.3.33 IpSourceStatus

Input ip source addess status

struct

multipleSources	**bool**
sourceIpAddress	**string**

49.3.34 IpStatusReq

variant

ids	**list** of **UUID**

49.3.35 IpStatusRes

struct

data	**map** from **UUID** to **IpInputStatus**

49.3.36 OutputSourceStatus

GetOutputStatus : Units are all in microseconds (us), but in the GUI it is converted to milliseconds (ms) with two decimals.

struct

effectiveIdenticalSourceOffset	**optional int** us of delay between Hot MAIN and BACKUP sources; it will be filled if OutputSource is configured with HotStandbyWithIdenticalSources
mainBufferDelay	**optional int** Current buffer delay in microseconds (us) for MAIN source. Undefined when MAIN is faulty
backupBufferDelay	**optional int** Current buffer delay in us for BACKUP source. Undefined when BACKUP is faulty
currentAlternativeIdx	**int** Index of currently selected alternative. 0 is MAIN, 1 is first backup and so on
alternativeStatuses	**list** of **AlternativeStatus** Status of each alternative defined for given source

49.3.37 OutputStatus

struct

bitrates	**IpOutputBitrates**
outputSources	**list** of **OutputSourceStatus** list of output's outputSources
	optional bigint

rtpErrors	The amount of RTP errors seen on this output

49.3.38 PidStatusRequest

struct

inputs	**list** of **UUID**

49.3.39 PidStatusResponse

struct

data	**map** from **UUID** to **tsstatus.TransportStreamStatus**

49.3.40 SeamlessPortStatus

Seamless Port Status

struct

activeSource	**firewallTypes.SocketAddress** Active source address
bitrate	**bigint** Current bitrate in Kbps
seamlessProtocol	**SeamlessProtocol** Seamless protocol used
role	**SeamlessRole** Role of the port (Master/Slave)
sequenceErrors	**bigint** Number of sequence errors
bufferDelay	**optional int** Buffer delay in microseconds (us)

49.3.41 SeamlessProtocol

enum

RTP
SequencedUDP
UDP
SRT
Unknown

49.3.42 SeamlessRole

enum

Master
Slave

49.3.43 SeamlessStatus

Seamless Status

struct

portA	**optional SeamlessPortStatus** Status for port A
portB	**optional SeamlessPortStatus** Status for port B
relativeDelay	**optional float** Relative delay between the two ports in seconds
synchronized	**bool** True if the two ports are synchronized

49.3.44 ServiceSourceInfo

struct

serviceId	**int**
name	**string**
owner	**string**

49.3.45 SrtConnectionStatus

SRT Connection Status

Status for an active SRT connection.

struct

srtSocketId	**int** Local SRT socket ID
peerEndpoint	**string** UDP endpoint of peer, given as a string of the form 'ip:port'.
peerStreamId	**optional string** SRT Stream ID reported by peer. Status is only applicable for listeners, as only callers can report Stream ID.
srtLatency	**int** Actual SRT latency of the connection, as a result of handshake, given in milliseconds.
rtt	**float** Round trip time, given in milliseconds (ms).
uptime	**int** Connection uptime, given in seconds.
localGroupId	**optional int** Local group ID, if connection is using SRT bonding

49.3.46 SrtInputConnectionStatus

SRT Input Connection Status

struct

socketStatus	**SrtSocketState** Current SRT socket status
connectionStatus	**optional SrtConnectionStatus** SRT connection status, if connected
totalPackets	**bigint** Total number of received packets
totalReceivedRetransmittedPackets	**bigint** Total number of received retransmitted packets
totalDroppedPackets	**bigint** Total number of packets dropped

49.3.47 SrtInputPathStatus

SRT Input Path Status Status for input if the input is single, or one path if input is seamless

struct

srtStatus	**optional SrtInputConnectionStatus** SRT status for path
inbandLinkStatus	**optional InbandLinkStatus** Inband Link Status for path

49.3.48 SrtInputStatus

SRT Input Status

struct

hasBitrate	**bool** True if input TS has bitrate
rtpErrors	**optional int** RTP Errors on input
syncByteError	**optional int** Sync Byte Errors on input
teiBitError	**optional int** TEI bit Errors on input
ccErrors	**optional int** Number of CC errors on the TS
services	**list** of **ServiceSourceInfo** Info on the services in TS
dejitterStatus	**optional DejitterStatus** Dejitter Status on Input
path1	**optional SrtInputPathStatus** Status for single input, or status for input path 1 if input is configured as seamless.
path2	**optional SrtInputPathStatus** Status for input path 2 if input is configured as seamless.
seamlessStatus	**optional SeamlessStatus** Seamless status for input if configured.

49.3.49 SrtOutputConnectionStatus

SRT Output Connection Status

struct

socketStatus	**SrtSocketState** Current SRT socket status
connectionStatus	**optional SrtConnectionStatus** SRT connection status, if connected
totalPackets	**bigint** Total number of packets, including retransmissions
totalRetransmittedPackets	**bigint** Total number of retransmitted packets

49.3.50 SrtOutputConnectionStatusPair

SRT Output Status pair

struct

path1	**optional SrtOutputConnectionStatus** Status for single output, or status for output path 1 if output is configured as cloned.
path2	**optional SrtOutputConnectionStatus** Status for output path 2 if output is configured as cloned.

49.3.51 SrtOutputStatus

SRT Output Status

struct

status	**list** of **SrtOutputConnectionStatusPair** List of status per active stream
activeConnections	**int** Number of active SRT connections
connectionCountTotal	**int** Accumulated number of successful SRT connections

49.3.52 SrtSocketState

SRT Socket State

enum

CLOSED	SRT socket is not connected
CONNECTED	SRT socket is connected

49.3.53 SrtStatusInputRes

struct

data	**map** from **UUID** to **SrtInputStatus**

49.3.54 SrtStatusOutputRes

struct

data	**map** from **UUID** to **SrtOutputStatus**

49.3.55 SrtStatusReq

variant

ids	**list** of **UUID**

49.3.56 TimeSource

Time Source

enum

PTP	PTP (Precision Time Protocol) time source
NTP	NTP (Network Time Protocol) time source
NoSource	No time source

49.3.57 ZixiConnectionStatus

Zixi Connection State

enum

ZIXI_DISCONNECTED	Connection is closed
ZIXI_CONNECTING	Connection is being established
ZIXI_CONNECTED	Connection is established
ZIXI_DISCONNECTING	Connection is being closed
ZIXI_RECONNECTING	Connection is being established after connection loss

49.3.58 ZixiInputPathStatus

Zixi Input Path Status Status for input if the input is single, or one path if input is seamless

struct

zixiStatus	**optional ZixiStatus** Zixi specific status for path
inbandLinkStatus	**optional InbandLinkStatus** Inband Link Status for path

49.3.59 ZixiInputStatus

Input Status for Zixi Inputs

struct

hasBitrate	**bool** True if input TS has bitrate
rtpErrors	**optional int** RTP Errors on input
syncByteError	**optional int** Sync Byte Errors on input
teiBitError	**optional int** TEI bit Errors on input
ccErrors	**optional int** Number of CC errors on the TS
services	**list** of **ServiceSourceInfo** Info on the services in TS
dejitterStatus	**optional DejitterStatus** Dejitter Status on Input
path1	**optional ZixiInputPathStatus** Status for single input, or status for input path 1 if input is configured as seamless.
path2	**optional ZixiInputPathStatus** Status for input path 2 if input is configured as seamless.
seamlessStatus	**optional SeamlessStatus** Seamless status for input if configured.

49.3.60 ZixiInputStatusReq

Request and response to deliver ZixiInputStatus

variant

ids	**list** of **UUID**

49.3.61 ZixiInputStatusRes

struct

data	**map** from **UUID** to **ZixiInputStatus**

49.3.62 ZixiOutputStatus

Zixi Output Status

struct

path1	**optional ZixiStatus** Status for single output, or status for output path 1 if output is configured as cloned.
path2	**optional ZixiStatus** Status for output path 2 if output is configured as cloned.

49.3.63 ZixiOutputStatusReq

Request and response to deliver ZixiOutputStatus

variant

ids	**list** of **UUID**

49.3.64 ZixiOutputStatusRes

struct

data	**map** from **UUID** to **ZixiOutputStatus**

49.3.65 ZixiStatus

Zixi Specific Input Status

struct

connection	**ZixiConnectionStatus** Status for connection to Zixi broadcaster
upTime	**bigint** Lifetime of current connection, given in seconds
rtt	**int** Round trip time given in milliseconds
jitter	**int** jitter time in milliseconds
totalPackets	**bigint** Total number of packets delivered
droppedPackets	**bigint** Number of packets dropped
notRecoveredPackets	**bigint** Number of nonrecovered packets
fecRecoveredPackets	**bigint** Number of recovered packets by forward error correction
arqRecoveredPackets	**bigint** Number of recovered packets by automatic repeat request

50 streamtype (1.1)

50.1 Overview

Changelog:

1.1

- **Changed**
  - Add MPEG2_AAC to StreamType type

50.2 Type Reference

50.2.1 StreamType

StreamType as defined in ISO 13818-1 Table 2-29.

enum

MPEG_VIDEO
MPEG_AUDIO
TTX
DVBSUB
PRIVATE
AC_3
H264
VBI
MPEG2_AAC
AAC
VC_1
EAC3
AIT
HBBTV_CAROUSEL
DATA_CAROUSEL
H265
S302M
ATSC_MPEG2_VIDEO
ATSC_AC3_AUDIO
ATSC_SCTE_35
ATSC_ENHANCED_AC3_AUDIO

51 triggers (1.0)

51.1 Command Reference

51.1.1 GetTriggers

Retrieve current Triggers object from database.

- **message GetTriggers.Request**
- **message GetTriggers.Response**

51.1.2 SetTriggers

Update Triggers object in database.

- **message SetTriggers.Request**
- **message SetTriggers.Response**

51.1.3 GetConfigurableTriggers

Returns a map of all configurable triggers. The key is the alarm ID and the value is the human-readable name of the trigger override.

- **message GetConfigurableTriggers.Request**
- **message GetConfigurableTriggers.Response**

51.2 Type Reference

51.2.1 AlarmId

A string uniquely identifying an alarm in the system. See Alarm Documentation for details. Should contain both namespace and ID (e.g., "stream_analysis/pmt_missing").

string

51.2.2 ConfigurableTriggerAlarmIds

Set of alarm IDs for alarms that supports trigger overriding.

set of **AlarmId**

51.2.3 GetConfigurableTriggers.Request

empty **struct**

51.2.4 GetConfigurableTriggers.Response

struct

`configurableTriggers`	**map** from **AlarmId** to **TriggerName**

51.2.5 GetTriggers.Request

empty **struct**

51.2.6 GetTriggers.Response

Triggers

51.2.7 SetTriggers.Request

Triggers

51.2.8 SetTriggers.Response

empty **struct**

51.2.9 TriggerActive

Boolean specifying the state of a trigger override.

bool

51.2.10 TriggerConfig

Each trigger corresponds to an alarm, but trigger overrides do not impact the alarms themselves. If a trigger is disabled, the alarm will appear but the flow will not be considered broken.

struct

`triggers`	**map** from **AlarmId** to **TriggerActive**

51.2.11 TriggerName

Human-readable name used to diplay a trigger override in the UI.

string

51.2.12 Triggers

- Object stored in the database containing the Trigger Configurations
- The config contains a map of all trigger overrides that should apply globally for all flows on this card.
- The map should contain all the triggers returned by GetConfigurableTriggers.
- For all configurable triggers not present, a default TriggerActive will be used instead.
- Default TriggerActive is True for Critial alarms and False for alarms with other severities.
-

struct

`config`	**TriggerConfig**

51.2.13 ValidateTriggerConfigResult

struct

validTriggers	**set** of **AlarmId**
invalidTriggers	**set** of **AlarmId**
missingTriggers	**set** of **AlarmId**

52 tsstatus (1.5)

52.1 Overview

Changelog:

1.5

- **Added**
  - RPC `GetOutputPidStatus` was added, which takes in a list of uuid of outputs and returns the status of the respective output transport streams.

1.4

- **Changed**
  - `component` from 1.4 to 1.5

1.3

- **Changed**
  - Dependency module component bumped from version 1.1 to 1.4

52.2 Command Reference

52.2.1 GetOutputPidStatus

Get output PID status

- **message GetOutputPidStatus.Request**
- **message GetOutputPidStatus.Response**

52.3 Type Reference

52.3.1 GetOutputPidStatus.Request

OutputPidStatusRequest

52.3.2 GetOutputPidStatus.Response

OutputPidStatusResponse

52.3.3 OutputPidStatusRequest

Output PID status request

struct

`outputs`	**list** of **UUID**

52.3.4 OutputPidStatusResponse

Response of an output PID status request

struct

data	**map** from **UUID** to **OutputTransportStreamStatus**

52.3.5 OutputTransportStreamStatus

Status for the output TS

struct

services	**map** from **int** to **ServiceSourceStatus** Status for each service in the TS. Map from service ID to status struct.
pids	**list** of **PidStatus** Info on all PIDs, both signalled and unsignalled. Map from service PID value to status struct.

52.3.6 PidStatus

PidStatus state of PID

struct

serviceId	**list** of **int** List of serviceIds a PID belong to.
componentType	**optional component.ComponentType** ComponentType of PID
ccErrors	**int** CC errors on PID
dataRate	**bigint** Date rate of PID
scrambling	**Scrambling** Scrambling of PID (none, even or odd)
PCR	**bool**
present	**bool** True if signalled pid has rate
language	**optional string** Language of PID, if applicable (e.g. audio pid)
casId	**optional int** Id of CAS system, if applicable (e.g. ECM pid)
pidIn	**int** PID number on input
pidOut	**optional int** PID number on output

52.3.7 Scrambling

Scrambling state of PID

enum

NONE	No scrambling
EVEN	Scrambling with even control word

ODD	Scrambling with odd control word

52.3.8 ServiceSourceStatus

Status for an input service

struct

serviceName	**optional string** Service name
pmtPid	**int** PMT pid for service
pcrPid	**optional int** PCR pid for service if present

52.3.9 TransportStreamStatus

Status for the TS

struct

ccErrors	**int** Total CC error count for this TS.
services	**map** from **int** to **ServiceSourceStatus** Status for each service in the TS. Map from service ID to status struct.
pids	**map** from **int** to **PidStatus** Info on all PIDs, both signalled and unsignalled. Map from service PID value to status struct.
tsId	**optional int** Transport Stream ID; set if PAT pid present in stream
originalNetworkId	**optional int** Original Network ID; set if SDT pid present in stream

53 zixi (1.1)

53.1 Type Reference

53.1.1 ZixiAlgorithm

Zixi Encryption Algorithm

enum

AES_128	AES with 128-bit key
AES_192	AES with 192-bit key
AES_256	AES with 256-bit key
CHACHA20	Chacha20 with 256-bit key

53.1.2 ZixiClonedTransport

Zixi Cloned Transport

struct

pathA	**ZixiTransportSettings** Cloned Path A
pathB	**ZixiTransportSettings** Cloned Path B

53.1.3 ZixiEncryption

Zixi Encryption

struct

algorithm	**ZixiAlgorithm** Encryption algorithm
key	**string** Encryption key as a hex string

53.1.4 ZixiInputSettings

Zixi Input Settings

struct

transport	**ZixiInputTransport**
decryption	**optional ZixiEncryption**
maxLatency	**int**

53.1.5 ZixiInputTransport

Zixi Input Transport

Seamless or single input

variant

`seamless`	**ZixiSeamlessTransport** Seamless input
`single`	**ZixiTransportSettings** Single input

53.1.6 ZixiOutputSettings

Zixi Output Settings

struct

`transport`	**ZixiOutputTransport**
`maxLatency`	**int**
`encryption`	**optional ZixiEncryption**
`tsPacketsPerFrame`	**int**
`maxBitrate`	**int**

53.1.7 ZixiOutputTransport

Zixi Output Transport

Cloned or single output

variant

`cloned`	**ZixiClonedTransport** Cloned output
`single`	**ZixiTransportSettings** Single output

53.1.8 ZixiSeamlessTransport

Zixi Seamless Transport

struct

`pathA`	**ZixiTransportSettings** Seamless Path A
`pathB`	**ZixiTransportSettings** Seamless Path B
`preferred`	**ZixiSeamlessTransport.preferred** Preferred pipe
`seamlessBufferSize`	**int** Input buffer size. Buffer for reconstructing missing packets on the input. Ranges 1-400ms

53.1.9 ZixiSeamlessTransport.preferred

enum

A	
B	
FLOATING	

53.1.10 ZixiTransportSettings

Zixi Transport Settings

struct

interfaceId	**UUID** UUID of the network interface
streamId	**string** Stream id, as indicated by the broadcaster
streamPassword	**optional string** Stream password, optional field used for authentication
remoteHost	**string** The IP-address of the broadcaster
remotePort	**int** Port of the remote broadcaster in the range [1024-65535]
localPort	**int** Local UDP port for Zixi traffic

54 zixiValidate (1.0)

54.1 Type Reference

54.1.1 IntRange

Integer Range

struct

`min`	**int** Minimum value of the range.
`max`	**int** Maximum value of the range.

54.1.2 Minimum

Minimum

struct

`min`	**int** Minimum value of the range

54.1.3 Result

Validation result

struct

`result`	**bool** True if config is valid and false otherwise.
`msg`	**optional string** An error message, which may be given if the result is invalid.