Open API For FX Series DSP Amplifiers

for Installers

January 2023

Table of Contents

1 Introduction

1.1 Revision History

Revision	Date	Changed By	Description	Firmware
3	05/09-2022	MAM	API Version 1.3	1.3+
2	09/03-2022	MAM	API Version 1.2	1.2+
1	16/12-2021	MAM	API Version 1.1	1.1+
0	11/11-2021	MAM	Initial Version	1.0+

1.1.1 API Version 1.3 Changes

- Added Output Gain
- Added Clip Limiter Mode
- Added Security Registers for WebPage Security
- Added: Input Gain Min + Input Gain Max
- Added: Analog Volume Control Value register as Value
- Remove: Analog Volume Control Volume register
- Update: Input Gain - Range increase from [-10, 10] to [-15, 15] dB
- Update: Zone Gain - when using Analog Volume Control
- Update: SETUP.LAN and SETUP.WIFI registers as read only

1.1.1.1 Registers Added

Register Name
ZONE-{ZID}.GAIN_MIN
ZONE-{ZID}.GAIN_MAX
OUT-{OID}.GAIN
OUT-{OID}.CLIP_LIMITER.MODE
VC-{VID}.VALUE
SYSTEM.SECURITY.PASSWORD_ENABLE
SYSTEM.SECURITY.PASSWORD_HASH

1.1.1.2 Registers Updated

Register Name	Change
IN-{IID}.GAIN	Limits
ZONE-{ZID}.GAIN	Limits, more
SETUP.LAN.NETWORK_MODE	Read Only
SETUP.LAN.IP	Read Only
SETUP.LAN.MASK	Read Only
SETUP.LAN.GATEWAY	Read Only
SETUP.LAN.DNS1	Read Only
SETUP.LAN.DNS2	Read Only
SETUP.WIFI.ENABLE	Read Only
SETUP.WIFI.DISABLE_LAN_CONNECTED	Read Only
SETUP.WIFI.DISABLE_AFTER	Read Only
SETUP.WIFI.MODE	Read Only
SETUP.WIFI.AP_SSID	Read Only
SETUP.WIFI.AP_PASS	Read Only
SETUP.WIFI.STA_SSID	Read Only
SETUP.WIFI.STA_PASS	Read Only

1.1.1.3 Registers Removed

Register Name
VC-{VID}.VOLUME

1.1.2 API Version 1.2 Changes

- INC Command support for Input Gain
- Added Frequency parameter for SUBSCRIBE command

1.1.2.1 Registers Added

Register Name
SETUP.SYSTEM.CUSTOM1
SETUP.SYSTEM.CUSTOM2
SETUP.SYSTEM.CUSTOM3
OUT-{OID}.PRESET.PROTECTED

1.1.2.2 Registers Updated

Register Name
OUT-{OID}.PRESET.LOCKED

1.1.3 API Version 1.1 Changes

- Added TCP Socket API
- SUBSCRIBE Command now handles subscriptions to Registers or Dynamics or all.
- INC Command for Zone Gain

1.1.3.1 Registers Renamed

Old Name	New Name
SYSTEM.STATE	SYSTEM.STATUS.STATE
SYSTEM.STATUS.SIGNAL_IN	SYSTEM.STATUS.SIGNAL_IN
SYSTEM.STATUS.SIGNAL_OUT	SYSTEM.STATUS.SIGNAL_OUT
SYSTEM.STATUS.LAN	SYSTEM.STATUS.LAN
SYSTEM.STATUS.WIFI	SYSTEM.STATUS.WIFI
SETUP.DEVICE.SWID	SYSTEM.DEVICE.SWID
SETUP.DEVICE.HWID	SYSTEM.DEVICE.HWID
SETUP.DEVICE.VENDOR_NAME	SYSTEM.DEVICE.VENDOR_NAME
SETUP.DEVICE.MODEL_NAME	SYSTEM.DEVICE.MODEL_NAME
SETUP.DEVICE.SERIAL	SYSTEM.DEVICE.SERIAL
SETUP.DEVICE.FIRMWARE	SYSTEM.DEVICE.FIRMWARE
SETUP.DEVICE.FIRMWARE	SYSTEM.DEVICE.FIRMWARE
SETUP.DEVICE.MAC	SYSTEM.DEVICE.MAC
SETUP.DEVICE.WIFI_MAC	SYSTEM.DEVICE.WIFI_MAC

Old Name	New Name
OUT-{OID}.LIMITER.AUTO	OUT-{OID}.PEAK_LIMITER.AUTO
OUT-{OID}.LIMITER.THRESHOLD	OUT-{OID}.PEAK_LIMITER.THRESHOLD
OUT-{OID}.LIMITER.ATTACK	OUT-{OID}.PEAK_LIMITER.ATTACK
OUT-{OID}.LIMITER.RELEASE	OUT-{OID}.PEAK_LIMITER.RELEASE
OUT-{OID}.LIMITER.HOLD	OUT-{OID}.PEAK_LIMITER.HOLD

1.1.3.2 Registers Added

Register Name
ZONE-{ZID}.COMPRESSOR.HOLD
OUT-{OID}.PRESET.NAME
OUT-{OID}.PRESET.ID
OUT-{OID}.PRESET.LOCKED
OUT-{OID}.POLARITY.PROTECTED
OUT-{OID}.OUTPUT_MODE.PROTECTED
OUT-{OID}.SPEAKER_DELAY.PROTECTED
OUT-{OID}.LIMITER.PROTECTED
OUT-{OID}.SPEAKER_EQ.PROTECTED
OUT-{OID}.XR.PROTECTED
OUT-{OID}.FIR.PROTECTED
OUT-{OID}.PEAK_LIMITER.BYPASS
OUT-{OID}.PEAK_LIMITER.KNEE
OUT-{OID}.RMS_LIMITER.BYPASS
OUT-{OID}.RMS_LIMITER.THRESHOLD
OUT-{OID}.RMS_LIMITER.ATTACK
OUT-{OID}.RMS_LIMITER.RELEASE
OUT-{OID}.RMS_LIMITER.HOLD
OUT-{OID}.RMS_LIMITER.KNEE
OUT-{OID}.CLIP_LIMITER.BYPASS
OUT-{OID}.FIR.BYPASS
OUT-{OID}.FIR.TAPS

1.2 Connecting to the Amplifier

Out of the Box the amplifier is hard-coded with the Ethernet Address 192.168.64.100. It is also possible to connect to the amplifier using Wifi. Connect to the Wifi AP (SSID) and connect using the default IP address of 192.168.4.1.

1.3 Discovery (mDNS)

If the application requires the amplifier to have a dynamic IP address, it is possible to use mDNS to locate the amplifier.

The service type is: `_pasconnect._tcp`

The following properties is defined:

- **api_version** - the api version of the device
- **device_type** - the device type. For amplifiers this will always be `AshAmpControl`
- **model** - the model name of the device
- **software_id** - software id of the amplifier (Manufacturer and Model Specific)
- **hardware_id** - hardware id of the amplifier (Model ID)

Example (Avahi for Linux):

```
$> avahi-browse -t -r _ashconnect._tcp
+ enp0s8 IPv4 ASHLY-IP1252-2122-00031.local
_ashconnect._tcp     local
= enp0s8 IPv4 ASHLY-IP1252-2122-00031.local
_pashconnect._tcp     local
   hostname = [ASHLY-IP1252-2122-00031.local.local]
   address = [192.168.64.100]
   port = [80]
   txt = ["api_version=1.3" "device_type=AshAmpControl" "manufacturer=Ashly
Audio" "model=IP 125.2" "software_id=2" "model_id=2"]
```

1.4 Definitions

1.4.1 Input Channels

The following input channels is defined for the amplifier.

- **0** - Unused Input (Silent)
- **100** - Analog Input 1
- **101** - Analog Input 2
- **102** - Analog Input 3
- **103** - Analog Input 4
- **200** - SPDIF 1 (Left)
- **201** - SPDIF 1 (Right)
- **400** - Noise Generator

1.4.2 Zones

The following zones is defined for the amplifier.

- **A** - Zone A
- **B** - Zone B
- **C** - Zone C *(4 channel version only)*
- **D** - Zone D *(4 channel version only)*

1.4.3 Output Channels sec:output_channels

- **1** - Output 1
- **2** - Output 2
- **3** - Output 3 *(4 channel version only)*
- **4** - Output 4 *(4 channel version only)*

1.4.4 Variable Types

- **Float** - Float format, delimited with '.'
- **Integer** - Normal integer
- **Enum** - Basically a string with a predefined set of options
- **String** - String - might have limitations on number of characters. String values containing spaces must be enclosed in double-quotes.

2 API Endpoints

2.1 Raw Socket API

The Primary API in the amplifier is based on TCP Socket based (**Port 7621**) and is **Line based**. That means every line is delimited by newline \n. Every line contains a single message. The API consists of 2 parts - a Command/Response interface and a Publish/Subscribe Interface.

2.1.1 Ncat

Examples in documentation is be based on Ncat https://nmap.org/download.html. The specific syntax is powershell - but can easily be converted to bash for Linux.

Powershell style:

```
$> "POWER_ON" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*POWER_ON
```

bash style:

```
$> echo "POWER_ON" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*POWER_ON
```

2.2 WebSocket API

It is also possible to connect to the Websocket based API in the amplifier. The syntax of commands and replies is exactly the same between the Socket based API and the WebSocket based API - though a single websocket message might contain/return multiple lines of text - with each line containing a single message.

3 Command/Response

The Command/Response interface allows for Querying/Updating the registers in the amplifier and to execute commands.

To execute a command - send a websocket message with the command followed by newline.

- If the command executes successfully the response will be an asterisk followed by the command text.

```
$> "<COMMAND>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*<COMMAND>
```

- If the command fails the response will be an hash followed by an error description.

```
$> "<COMMAND>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
# <Error Message>
```

- If the command returns data in form of registers the response will be:

```
$> "<COMMAND>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+<RESPONSE>
*<COMMAND>
```

3.1 Command Types

3.1.1 GET

Get value of amplifier register. The command supports wildcards.

Format:

```
"GET <REGISTER>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+<RESPONSE(s)>
*<COMMAND>
```

Example:

```
$> "GET IN-100.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.NAME "Analog 1"
*GET IN-100.NAME

"GET IN-*.NAME" | websocat -t -0 ws://192.168.64.100/ws
+IN-100.NAME "Analog 1"
+IN-101.NAME "Analog 2"
```

```
+IN-102.NAME "Analog 3"
+IN-103.NAME "Analog 4"
+IN-200.NAME "S/PDIF 1"
+IN-201.NAME "S/PDIF 1R"
+IN-400.NAME "Noise Generator"
*GET IN-*.NAME
```

3.1.2 SET

Set value in amplifier register. The command does not support wildcards!

Format:

```
"SET <REGISTER> <VALUE>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*<COMMAND>
```

Example

```
$> "SET IN-100.NAME ""Streamer""" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+IN-100.NAME "Analog 1"
*SET IN-100.NAME "Streamer"
```

3.1.3 INC

Modifies the value in amplifier register by the amount specified in the command. The valus can be positive or negative. The command does not support wildcards!

Format:

```
"INC <REGISTER> <VALUE>" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+<REGISTER> <MODIFIED VALUE>
*<COMMAND>
```

Example

```
$> "INC ZONE-A.GAIN -5 | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN -5.00
*INC ZONE-A.GAIN -5
```

3.1.4 SUBSCRIBE

Subscribe to changes in all registers and dynamics. The subscribe command does not support subscriptions to individual registers. *This might change in a later release*.

The register changes will stream to the websocket after subscription…

```
$> "SUBSCRIBE" | ncat 192.168.64.100 7621 --no-shutdown
...
+IN-100.DYN.SIGNAL -49.9777
+IN-100.DYN.CLIP 0
+IN-101.DYN.SIGNAL -49.3077
+IN-101.DYN.CLIP 0
```

```
+IN-102.DYN.SIGNAL -99.7209
+IN-102.DYN.CLIP 0
...
*SUBSCRIBE
```

3.1.5 SUBSCRIBE <BLANK|*|REG|DYN> <FREQ>

- **REG** - Register Updates Only
- **DYN** - Dynamic updates Only
- **"*"** - All register updates - Equal to BLANK
- **"BLANK"** - IF EMPTY - Both Dynamic and register updates
- **"FREQ"** - Frequency of updates: 1=1 update per second, 0.5 equals 1 update every 5 seconds.

Subscribe to changes in all registers or dynamic updates. The subscribe command does not support subscriptions to individual registers. *This might change in a later release*.

The register changes will stream to the socket/websocket after subscription…

Example: Subscribe to All updates

```
$> "SUBSCRIBE" | ncat 192.168.64.100 7621 --no-shutdown
...
+IN-100.DYN.SIGNAL -49.9777
+IN-100.DYN.CLIP 0
+IN-101.DYN.SIGNAL -49.3077
+IN-101.DYN.CLIP 0
+IN-102.DYN.SIGNAL -99.7209
+IN-102.DYN.CLIP 0
...
*SUBSCRIBE
```

Example: Subscribe to Register only updates

```
$> "SUBSCRIBE REG" | ncat 192.168.64.100 7621 --no-shutdown
...
+ZONE-A.GAIN -5.00
...
*SUBSCRIBE REG
```

Example: Subscribe to Dynamic updates - but limit frequency to 1 Hz

```
$> "SUBSCRIBE DYN 1" | ncat 192.168.64.100 7621 --no-shutdown
...
+IN-100.DYN.SIGNAL -49.9777
+IN-100.DYN.CLIP 0
+IN-101.DYN.SIGNAL -49.3077
+IN-101.DYN.CLIP 0
+IN-102.DYN.SIGNAL -99.7209
```

```
+IN-102.DYN.CLIP 0
...
*SUBSCRIBE DYN 1
```

3.1.6 UNSUBSCRIBE <BLANK|*|REG|DYN>

- **REG** - Register Updates Only
- **DYN** - Dynamic updates Only
- **"BLANK"** - IF EMPTY - Both dynamic and register updates

Unsubscribe to the previous subscription. The parameter must match a previous subscription. An Unsubscribe All (Blank value) will not unsubscribe a subscription to register only updates.

3.1.7 POWER_ON

TYPE: Command

Methods: POWER_ON

Example:

```
$> "POWER_ON" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*POWER_ON
```

3.1.8 POWER_OFF

TYPE: Command

Methods: POWER_OFF

Example:

```
$> "POWER_OFF" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*POWER_OFF
```

3.2 Registers

Supported Registers for General Use

3.2.1 Base Registers

Register Name	Type	Access	Note
API_VERSION	String	Get	
SYSTEM.STATUS.STATE	Enum	Get	{INIT, STANDBY, ON, FAULT}
SYSTEM.STATUS.SIGNAL_IN	Enum	Get	{OFF, NO_SIGNAL, SIGNAL, CLIP}
SYSTEM.STATUS.SIGNAL_OUT	Enum	Get	{OFF, NO_SIGNAL, SIGNAL, CLIP, FAULT}
SYSTEM.STATUS.LAN	String	Get	IP Address or Empty
SYSTEM.STATUS.WIFI	String	Get	

3.2.2 Device Information Registers

Register Name	Type	Access	Note
SYSTEM.DEVICE.SWID	Integer	Get	
SYSTEM.DEVICE.HWID	Integer	Get	
SYSTEM.DEVICE.VENDOR_NAME	String	Get	
SYSTEM.DEVICE.MODEL_NAME	String	Get	
SYSTEM.DEVICE.SERIAL	String	Get	
SYSTEM.DEVICE.FIRMWARE	String	Get	
SYSTEM.DEVICE.FIRMWARE_DATE	String	Get	
SYSTEM.DEVICE.MAC	String	Get	
SYSTEM.DEVICE.WIFI_MAC	String	Get	

3.2.3 System Information Registers

Register Name	Type	Access	Note
SETUP.SYSTEM.DEVICE_NAME	String[32]	Get, Set	
SETUP.SYSTEM.VENUE_NAME	String[32]	Get, Set	
SETUP.SYSTEM.CUSTOMER_NAME	String[32]	Get, Set	
SETUP.SYSTEM.ASSET_TAG	String[32]	Get, Set	
SETUP.SYSTEM.INSTALLER_NAME	String[32]	Get, Set	
SETUP.SYSTEM.CONTACT_INFO	String[32]	Get, Set	
SETUP.SYSTEM.INSTALL_DATE	String[32]	Get, Set	
SETUP.SYSTEM.INSTALL_NOTES	String[512]	Get, Set	
SETUP.SYSTEM.LOCATING	Boolean	Get, Set	
SETUP.SYSTEM.CUSTOM1	String[8192]	Get, Set	
SETUP.SYSTEM.CUSTOM2	String[8192]	Get, Set	
SETUP.SYSTEM.CUSTOM3	String[8192]	Get, Set	

3.2.4 Generator Registers

Register Name	Type	Access	Note
GENERATOR.ENABLE	Boolean	Get, Set	

3.2.5 Input Registers

Register Name	Type	Access	Unit	Range
IN.COUNT	Integer	Get		*IID* in [1, .COUNT]
IN-{IID}.NAME	String[32]	Get, Set		
IN-{IID}.SENS	Enum	Get, Set		{14DBU, 4DBU, -10DBV, MIC}
IN-{IID}.GAIN	Float	Get, Set	dB	[-15.0, 15.0]

Register Name	Type	Access	Unit	Range
				[-48, 0] for Generator
IN-{IID}.STEREO	Boolean	Get, Set		
IN-{IID}.DYN.SIGNAL	Float	Get, Set	dB	[-144, 20.0]
IN-{IID}.DYN.CLIP	Boolean	Get, Set		

3.2.6 Zone Registers

Register Name	Type	Access	Unit	Range
ZONE.COUNT	Integer	Get, Set		*ZID* in [1, .COUNT]
ZONE-{ZID}.NAME	String[32]	Get, Set		
ZONE-{ZID}.PRIMARY_SRC	Integer	Get, Set		Valid *IID*
ZONE-{ZID}.GAIN	Float	Get, Set	dB	[GAIN_MIN, GAIN_MAX]
ZONE-{ZID}.GAIN_MIN	Float	Get, Set	dB	[-80, GAIN_MAX]
ZONE-{ZID}.GAIN_MAX	Float	Get, Set	dB	[GAIN_MIN, 0]
ZONE-{ZID}.STEREO	Boolean	Get, Set		
ZONE-{ZID}.GPIO_VC	Integer	Get, Set		Valid *VID, 0 for OFF*
ZONE-{ZID}.MUTE	Boolean	Get, Set		
ZONE-{ZID}.DYN.SIGNAL	Float	Subscribe	dB	[-144, 20.0]

3.2.7 Zone Compressor Registers

Register Name	Type	Access	Unit	Range
ZONE-{ZID}.COMPRESSOR.AUTO	Boolean	Get, Set		
ZONE-{ZID}.COMPRESSOR.THRESHOLD	Float	Get, Set	dB	[-40, 20]
ZONE-{ZID}.COMPRESSOR.ATTACK	Float	Get, Set	Sec	[0.0003, 0.050]
ZONE-{ZID}.COMPRESSOR.RELEASE	Float	Get, Set	Sec	[0.001, 1.0]
ZONE-{ZID}.COMPRESSOR.HOLD	Float	Get, Set	Sec	[0, 1]
ZONE-{ZID}.COMPRESSOR.RATIO	Float	Get, Set		[1, 50]
ZONE-{ZID}.COMPRESSOR.KNEE	Float	Get, Set	dB	[0, 12]
ZONE-{ZID}.COMPRESSOR.BYPASS	Boolean	Get, Set		

3.2.8 Output Registers

Register Name	Type	Access	Unit	Range
OUT.COUNT	Integer	Get		*OID* in [1, .COUNT]
OUT-{OID}.NAME	String[32]	Get, Set		

Register Name	Type	Access	Unit	Range
OUT-{OID}.GAIN	Float	Get, Set	dB	[-30.0, 15.0]
OUT-{OID}.MUTE	Boolean	Get, Set		
OUT-{OID}.SRC	String[1]	Get, Set		*ZID*
OUT-{OID}.SRC_CHANNEL	Enum	Get, Set		{L, R, S}
OUT-{OID}.POLARITY	Integer	Get, Set		{-1, 1}
OUT-{OID}.OUTPUT_MODE	Enum	Get, Set		{OFF, 8R, 70V, 100V, BTL}
OUT-{OID}.OUTPUT_HIGHPASS	Float	Get, Set	Hz	{0, [20-1000]}
OUT-{OID}.DYN.SIGNAL	Float	Subscribe	dB	[-144, 20.0]
OUT-{OID}.DYN.CLIP	Boolean	Subscribe		

3.2.9 Output Delay Registers

Register Name	Type	Access	Unit	Range
OUT-{OID}.DELAY.TIME	Float	Get, Set	Sec	[0.0, 0.1]
OUT-{OID}.DELAY.BYPASS	Boolean	Get, Set		

3.3 Advanced Registers

Please contact your manufacturer - for help integrating the advanced API's described below.

3.3.1 Output Speaker Preset

Register Name	Type	Access	Unit	Range
OUT-{OID}.PRESET.NAME	String	Get		
OUT-{OID}.PRESET.ID	String	Get		
OUT-{OID}.PRESET.LOCKED	Boolean	Get		
OUT-{OID}.PRESET.CUSTOMIZED	Boolean	Get		
OUT-{OID}.POLARITY.PROTECTED	Boolean	Get		
OUT-{OID}.OUTPUT_MODE.PROTECTED	Boolean	Get		
OUT-{OID}.SPEAKER_DELAY.PROTECTED	Boolean	Get		
OUT-{OID}.LIMITER.PROTECTED	Boolean	Get		
OUT-{OID}.SPEAKER_EQ.PROTECTED	Boolean	Get		
OUT-{OID}.XR.PROTECTED	Boolean	Get		
OUT-{OID}.FIR.PROTECTED	Boolean	Get		

3.3.2 Output Speaker Delay Registers

Register Name	Type	Access	Unit	Range
`OUT-{OID}.SPEAKER_DELAY.TIME`	`Float`	Get, Set	Sec	[0.0, 0.01]
`OUT-{OID}.SPEAKER_DELAY.BYPASS`	`Boolean`	Get, Set		

3.3.3 Output Peak Limiter Registrers

Register Name	Type	Access	Unit	Range
`OUT-{OID}.PEAK_LIMITER.BYPASS`	`Boolean`	Get, Set		
`OUT-{OID}.PEAK_LIMITER.AUTO`	`Boolean`	Get, Set		
`OUT-{OID}.PEAK_LIMITER.THRESHOLD`	`Float`	Get, Set	Vpeak	[1, 200]
`OUT-{OID}.PEAK_LIMITER.ATTACK`	`Float`	Get, Set	Sec	[0.0003, 0.100]
`OUT-{OID}.PEAK_LIMITER.RELEASE`	`Float`	Get, Set	Sec	[0.004, 2.0]
`OUT-{OID}.PEAK_LIMITER.HOLD`	`Float`	Get, Set	Sec	[0, 1.0]
`OUT-{OID}.PEAK_LIMITER.KNEE`	`Float`	Get, Set	dB	[0, 6.0]

3.3.4 Output RMS Limiter Registrers

Register Name	Type	Access	Unit	Range
`OUT-{OID}.RMS_LIMITER.BYPASS`	`Boolean`	Get, Set		
`OUT-{OID}.RMS_LIMITER.THRESHOLD`	`Float`	Get, Set	Vpeak	[1, 150]
`OUT-{OID}.RMS_LIMITER.ATTACK`	`Float`	Get, Set	Sec	[0.010, 30]
`OUT-{OID}.RMS_LIMITER.RELEASE`	`Float`	Get, Set	Sec	[0.010, 30]
`OUT-{OID}.RMS_LIMITER.HOLD`	`Float`	Get, Set	Sec	[0, 1.0]
`OUT-{OID}.RMS_LIMITER.KNEE`	`Float`	Get, Set	dB	[0, 6.0]

3.3.5 Output Clip Limiter Registrers

Register Name	Type	Access	Unit	Range
`OUT-{OID}.CLIP_LIMITER.BYPASS`	`Boolean`	Get, Set		
`OUT-{OID}.CLIP_LIMITER.MODE`	`Enum`	Get, Set		{NORMAL, FAST}

3.3.6 Output Eq Registrers

Register Name	Type	Access	Unit	Range
`OUT.EQ.COUNT`	`Integer`	Get		*EID* in [1, .COUNT]
`OUT-{OID}.EQ.BYPASS`	`Boolean`	Get, Set		
`OUT-{OID}.EQ-{EID}.TYPE`	`Enum`	Get, Set		{PARAMETRIC, LOWPASS_6, HIGHPASS_6, LOWPASS_12, HIGHPASS_12, LOW_SHELF, LOW_SHELF_Q, LOW_SHELF_6, LOW_SHELF_12,

Register Name	Type	Access	Unit	Range
				HIGH_SHELF, HIGH_SHELF_Q, HIGH_SHELF_6, HIGH_SHELF_12, BANDPASS, NOTCH, ALLPASS_1, ALLPASS_2}
`OUT-{OID}.EQ-{EID}.GAIN`	`Float`	Get, Set	dB	[-15, 15]
`OUT-{OID}.EQ-{EID}.FREQ`	`Float`	Get, Set	Hz	[20, 20000]
`OUT-{OID}.EQ-{EID}.Q`	`Float`	Get, Set		[0.4, 30]
`OUT-{OID}.EQ-{EID}.BYPASS`	`Boolean`	Get, Set		

3.3.7 Output SpeakerEq Registrers

Register Name	Type	Access	Unit	Range
`OUT.SPEAKER_EQ.COUNT`	`Integer`	Get		*SID* in [1, .COUNT]
`OUT-{OID}.SPEAKER_EQ.BYPASS`	`Boolean`	Get, Set		
`OUT-{OID}.SPEAKER_EQ-{SID}.TYPE`	`Enum`	Get, Set		{PARAMETRIC,
				LOWPASS_6, HIGHPASS_6, LOWPASS_12, HIGHPASS_12, LOW_SHELF, LOW_SHELF_Q, LOW_SHELF_6, LOW_SHELF_12, HIGH_SHELF, HIGH_SHELF_Q, HIGH_SHELF_6, HIGH_SHELF_12, BANDPASS, NOTCH, ALLPASS_1, ALLPASS_2}
`OUT-{OID}.SPEAKER_EQ-{SID}.GAIN`	`Float`	Get, Set	dB	[-15, 15]
`OUT-{OID}.SPEAKER_EQ-{SID}.FREQ`	`Float`	Get, Set	Hz	[20, 20000]
`OUT-{OID}.SPEAKER_EQ-{SID}.Q`	`Float`	Get, Set		[0.4, 30]
`OUT-{OID}.SPEAKER_EQ-{SID}.BYPASS`	`Boolean`	Get, Set		

3.3.8 Output Crossover Registrers

Register Name	Type	Access	Unit	Range
OUT-{OID}.XR.BYPASS	Boolean	Get, Set		
OUT-{OID}.XR.GAIN	Float	Get, Set	dB	[-15, 15]
OUT-{OID}.XR.LOWPASS_TYPE	Enum	Get, Set		{OFF, BUT6, BUT12, BUT18, BUT24, BUT48, BES12, BES24, BES48, LR12, LR24, LR36, LR48}
OUT-{OID}.XR.LOWPASS_FREQUENCY	Float	Get, Set	Hz	[20, 20000]
OUT-{OID}.XR.HIGHPASS_TYPE	Enum	Get, Set		{OFF, BUT6, BUT12, BUT18, BUT24, BUT48, BES12, BES24, BES48, LR12, LR24, LR36, LR48}
OUT-{OID}.XR.HIGHPASS_FREQUENCY	Float	Get, Set	Hz	[20, 20000]

3.3.9 Output FIR

Register Name	Type	Access	Unit	Range
OUT-{OID}.FIR.BYPASS	Boolean	Get, Set		
OUT-{OID}.FIR.TAPS	Integer	Get	dB	[0, 512]

3.3.10 Analog Volume Control Registers

Register Name	Type	Access	Unit	Range
VC.COUNT	Integer	Get		*VID* in range [1, VC.Count]
VC-{VID}.NAME	String	Get		
VC-{VID}.VALUE	Float	Get	Percent	[0, 100]

3.3.11 Power Management Registers

Register Name	Type	Access	Unit	Range
SETUP.POWER.POWER_ON	Enum	Get, Set		{AUDIO, AUDIO_ECO, TRIGGER, TRIGGER_ECO, NETWORK}
SETUP.POWER.MUTE_TIME	Integer	Get, Set	Sec	[0, 3600]
SETUP.POWER.STANDBY_TIME	Integer	Get, Set	Sec	[0, 3600]

3.3.12 GPIO Registers

Register Name	Type	Access	Unit	Range
SETUP.GPIO.PIN2	Enum	Get, Set		{OFF, STANDBY_NO, STANDBY_NC, MUTE_NO, MUTE_NC}
SETUP.GPIO.PIN4	Enum	Get, Set		{OFF, VOLUME_CONTROL}
SETUP.GPIO.PIN5	Enum	Get, Set		{OFF, VOLUME_CONTROL}
SETUP.GPIO.PIN6	Enum	Get, Set		{OFF, VOLUME_CONTROL, TRIGGER_12V_IN}
SETUP.GPIO.PIN7	Enum	Get, Set		{OFF, VOLUME_CONTROL, TRIGGER_12V_OUT}
SETUP.GPIO.PIN8	Enum	Get, Set		{VCC_3V3}

3.3.13 LAN Registers

Register Name	Type	Access	Unit	Range
SETUP.LAN.NETWORK_MODE	Enum	Get		{STATIC, DHCP}
SETUP.LAN.IP	String	Get		
SETUP.LAN.MASK	String	Get		
SETUP.LAN.GATEWAY	String	Get		
SETUP.LAN.DNS1	String	Get		
SETUP.LAN.DNS2	String	Get		

3.3.14 WiFi Registers

Register Name	Type	Access	Unit	Range
SETUP.WIFI.ENABLE	Boolean	Get		
SETUP.WIFI.DISABLE_LAN_CONNECTED	Boolean	Get		
SETUP.WIFI.DISABLE_AFTER	Float	Get	Sec	[0, 3600]
SETUP.WIFI.MODE	Enum	Get		{AP, STA}
SETUP.WIFI.AP_SSID	String	Get		
SETUP.WIFI.AP_PASS	String	Get		
SETUP.WIFI.STA_SSID	String	Get		
SETUP.WIFI.STA_PASS	String	Get		

3.3.15 Security Registers

Register Name	Type	Access	Unit	Range
SYSTEM.SECURITY.PASSWORD_ENABLE	Boolean	Get, Set		
SYSTEM.SECURITY.PASSWORD_HASH	String	Get, Set		

4 Register Reference

4.1 API_VERSION

TYPE: Register

METHODS: Get

VALUES: Enum:

- **INIT** - Amplifier is initializing
- **STANDBY** - Amplifier is in standby
- **ON** - Amplifier is on
- **FAULT** - Amplifier has Non recoverable Error

Example:

```
$> "GET API_VERSION" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+API_VERSION "1.1"
*GET API_VERSION
```

4.2 SYSTEM.STATUS.STATE

TYPE: Register

METHODS: Get

VALUES: Enum:

- **INIT** - Amplifier is initializing
- **STANDBY** - Amplifier is in standby
- **ON** - Amplifier is on
- **FAULT** - Amplifier has Non recoverable Error

Example:

```
$> "GET SYSTEM.STATUS.STATE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.STATUS.STATE "ON"
*GET SYSTEM.STATUS.STATE
```

4.3 SYSTEM.STATUS.SIGNAL_IN

TYPE: Register

METHODS: Get

VALUES: Enum:

- **OFF** - Input(s) is Off
- **NO_SIGNAL** - Input(s) has no signal (Below threshold)
- **SIGNAL** - Input(s) has signal (Above threshold)

- **CLIP** - Input(s) is clipping ADC - please decrease sensitivity

Example:

```
$> "GET SYSTEM.STATUS.SIGNAL_IN" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SYSTEM.STATUS.SIGNAL_IN "SIGNAL"
*GET SYSTEM.STATUS.SIGNAL_IN
```

4.4 SYSTEM.STATUS.SIGNAL_OUT

TYPE: Register

METHODS: Get

VALUES: Enum:

- **OFF** - Output(s) is Off
- **NO_SIGNAL** - Output(s) has no signal (Below threshold)
- **SIGNAL** - Output(s) has signal (Above threshold)
- **CLIP** - Output(s) is clipping in amplifier - please decrease volume.
- **FAULT** - Output(s) has unspecified fault

Example:

```
$> "GET SYSTEM.STATUS.SIGNAL_OUT" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SYSTEM.STATUS.SIGNAL_OUT "SIGNAL"
*GET SYSTEM.STATUS.SIGNAL_OUT
```

4.5 SYSTEM.STATUS.LAN

TYPE: Register

METHODS: Get

VALUES: STRING:

- **IP Address** - LAN is connected and has received IP Address
- **EMPTY** - LAN is not connected or no IP Address received/configured

Example:

```
$> "GET SYSTEM.STATUS.LAN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.STATUS.LAN "192.168.64.100"
*GET SYSTEM.STATUS.LAN
```

4.6 SYSTEM.STATUS.WIFI

TYPE: Register

METHODS: Get

VALUES: STRING:

- **IP Address** - WIFI is connected and has received IP Address
- **EMPTY** - WIFI is not connected or no IP Address received/configured

Example:

```
$> "GET SYSTEM.STATUS.WIFI" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.STATUS.WIFI "192.168.4.1"
*GET SYSTEM.STATUS.WIFI
```

4.7 SETUP.SYSTEM.DEVICE_NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.DEVICE_NAME" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SETUP.SYSTEM.DEVICE_NAME Ashly-2122023201X00031
*GET SETUP.SYSTEM.DEVICE_NAME

$> "SET SETUP.SYSTEM.DEVICE_NAME "MyBlaze"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.DEVICE_NAME MyAmp

$> "GET SETUP.SYSTEM.DEVICE_NAME" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SETUP.SYSTEM.DEVICE_NAME MyAmp
*GET SETUP.SYSTEM.DEVICE_NAME
```

4.8 SETUP.SYSTEM.VENUE_NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.VENUE_NAME" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SETUP.SYSTEM.VENUE_NAME ""
*GET SETUP.SYSTEM.VENUE_NAME

$> "SET SETUP.SYSTEM.VENUE_NAME "THouse"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
```

```
*SET SETUP.SYSTEM.VENUE_NAME "THouse"

$> "GET SETUP.SYSTEM.VENUE_NAME" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SETUP.SYSTEM.VENUE_NAME "THouse"
*GET SETUP.SYSTEM.VENUE_NAME
```

4.9 SETUP.SYSTEM.CUSTOMER_NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.CUSTOMER_NAME" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.CUSTOMER_NAME ""
*GET SETUP.SYSTEM.CUSTOMER_NAME

$> "SET SETUP.SYSTEM.CUSTOMER_NAME "R. Rock"" | ncat 192.168.64.100 7621 --
no-shutdown -i 1
*SET SETUP.SYSTEM.CUSTOMER_NAME "R. Rock"

$> "GET SETUP.SYSTEM.CUSTOMER_NAME" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.CUSTOMER_NAME "R. Rock"
*GET SETUP.SYSTEM.CUSTOMER_NAME
```

4.10 SETUP.SYSTEM.ASSET_TAG

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.ASSET_TAG" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.ASSET_TAG ""
*GET SETUP.SYSTEM.ASSET_TAG

$> "SET SETUP.SYSTEM.ASSET_TAG "XZ233WV"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.ASSET_TAG "XZ233WV"

$> "GET SETUP.SYSTEM.ASSET_TAG" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.ASSET_TAG "XZ233WV"
*GET SETUP.SYSTEM.ASSET_TAG
```

4.11 SETUP.SYSTEM.INSTALLER_NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.INSTALLER_NAME" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.INSTALLER_NAME ""
*GET SETUP.SYSTEM.INSTALLER_NAME

$> "SET SETUP.SYSTEM.INSTALLER_NAME "AV.X"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.INSTALLER_NAME "AV.X"

$> "GET SETUP.SYSTEM.INSTALLER_NAME" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.INSTALLER_NAME "AV.X"
*GET SETUP.SYSTEM.INSTALLER_NAME
```

4.12 SETUP.SYSTEM.CONTACT_INFO

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SETUP.SYSTEM.CONTACT_INFO" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+SETUP.SYSTEM.CONTACT_INFO ""
*GET SETUP.SYSTEM.CONTACT_INFO

$> "SET SETUP.SYSTEM.CONTACT_INFO "555-9753"" | ncat 192.168.64.100 7621 --
no-shutdown -i 1
*SET SETUP.SYSTEM.CONTACT_INFO "555-9753"

$> "GET SETUP.SYSTEM.CONTACT_INFO" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+SETUP.SYSTEM.CONTACT_INFO "555-9753"
*GET SETUP.SYSTEM.CONTACT_INFO
```

4.13 SETUP.SYSTEM.INSTALL_DATE

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 64 chars)

Example:

```
$> "GET SETUP.SYSTEM.INSTALL_DATE" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+SETUP.SYSTEM.INSTALL_DATE ""
*GET SETUP.SYSTEM.INSTALL_DATE

$> "SET SETUP.SYSTEM.INSTALL_DATE "01-01-2021"" | ncat 192.168.64.100 7621 --
no-shutdown -i 1
*SET SETUP.SYSTEM.INSTALL_DATE "01-01-2021"

$> "GET SETUP.SYSTEM.INSTALL_DATE" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+SETUP.SYSTEM.INSTALL_DATE "01-01-2021"
*GET SETUP.SYSTEM.INSTALL_DATE
```

4.14 SETUP.SYSTEM.INSTALL_NOTES

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 512 chars)

Example:

```
$> "GET SETUP.SYSTEM.INSTALL_NOTES" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.INSTALL_NOTES ""
*GET SETUP.SYSTEM.INSTALL_NOTES

$> "SET SETUP.SYSTEM.INSTALL_NOTES "Nice"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.INSTALL_NOTES "Nice"

$> "GET SETUP.SYSTEM.INSTALL_NOTES" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SETUP.SYSTEM.INSTALL_NOTES "Nice"
*GET SETUP.SYSTEM.INSTALL_NOTES
```

4.15 SETUP.SYSTEM.LOCATING

TYPE: Register

METHODS: Get, Set

VALUES: Boolean

Example:

```
$> "GET SETUP.SYSTEM.LOCATING" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.LOCATING 0
*GET SETUP.SYSTEM.LOCATING

$> "SET SETUP.SYSTEM.LOCATING 1" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET SETUP.SYSTEM.LOCATING 1

$> "GET SETUP.SYSTEM.LOCATING" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.LOCATING 1
*GET SETUP.SYSTEM.LOCATING
```

4.16 SETUP.SYSTEM.CUSTOM1

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 8192 chars)

Example:

```
$> "GET SETUP.SYSTEM.CUSTOM1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM1 ""
*GET SETUP.SYSTEM.LOCATING

$> "SET SETUP.SYSTEM.CUSTOM1 "Custom"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.CUSTOM1 "Custom"

$> "GET SETUP.SYSTEM.CUSTOM1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM1 "Custom"
*GET SETUP.SYSTEM.CUSTOM1
```

4.17 SETUP.SYSTEM.CUSTOM2

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 8192 chars)

Example:

```
$> "GET SETUP.SYSTEM.CUSTOM2" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM2 ""
*GET SETUP.SYSTEM.LOCATING

$> "SET SETUP.SYSTEM.CUSTOM2 "Custom"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
```

```
*SET SETUP.SYSTEM.CUSTOM2 "Custom"

$> "GET SETUP.SYSTEM.CUSTOM2" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM2 "Custom"
*GET SETUP.SYSTEM.CUSTOM2
```

4.18 SETUP.SYSTEM.CUSTOM3

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 8192 chars)

Example:

```
$> "GET SETUP.SYSTEM.CUSTOM3" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM3 ""
*GET SETUP.SYSTEM.LOCATING

$> "SET SETUP.SYSTEM.CUSTOM3 "Custom"" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET SETUP.SYSTEM.CUSTOM3 "Custom"

$> "GET SETUP.SYSTEM.CUSTOM3" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SETUP.SYSTEM.CUSTOM3 "Custom"
*GET SETUP.SYSTEM.CUSTOM3
```

4.19 SYSTEM.DEVICE.SWID

TYPE: Register

METHODS: Get

VALUES: Integer

Example:

```
$> "GET SYSTEM.DEVICE.SWID" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.SWID 2
*GET SYSTEM.DEVICE.SWID
```

4.20 SYSTEM.DEVICE.HWID

TYPE: Register

METHODS: Get

VALUES: Integer

Example:

```
$> "GET SYSTEM.DEVICE.HWID" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.HWID 4
*GET SYSTEM.DEVICE.HWID
```

4.21 SYSTEM.DEVICE.VENDOR_NAME

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.VENDOR_NAME" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+SYSTEM.DEVICE.VENDOR_NAME Ashly Audio
*GET SYSTEM.DEVICE.VENDOR_NAME
```

4.22 SYSTEM.DEVICE.MODEL_NAME

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.MODEL_NAME" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+SYSTEM.DEVICE.MODEL_NAME IP 125.2
*GET SYSTEM.DEVICE.MODEL_NAME
```

4.23 SYSTEM.DEVICE.SERIAL

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.SERIAL" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.SERIAL "2122023201X00031"
*GET SYSTEM.DEVICE.SERIAL
```

4.24 SYSTEM.DEVICE.FIRMWARE

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.FIRMWARE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.FIRMWARE "1.0.0"
*GET SYSTEM.DEVICE.FIRMWARE
```

4.25 SYSTEM.DEVICE.FIRMWARE_DATE

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.FIRMWARE_DATE" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+SYSTEM.DEVICE.FIRMWARE_DATE "Nov  5 2021 07:51:56"
*GET SYSTEM.DEVICE.FIRMWARE_DATE
```

4.26 SYSTEM.DEVICE.MAC

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.MAC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.MAC "C4:5B:BE:31:42:F3"
*GET SYSTEM.DEVICE.MAC
```

4.27 SYSTEM.DEVICE.WIFI_MAC

TYPE: Register

METHODS: Get

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET SYSTEM.DEVICE.WIFI_MAC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+SYSTEM.DEVICE.WIFI_MAC "C4:5B:BE:31:42:F0"
*GET SYSTEM.DEVICE.WIFI_MAC
```

4.28 INPUT.COUNT

TYPE: Register

METHODS: Get

VALUES: [Integer]

Example:

```
$> "GET IN.COUNT" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN.COUNT 7
*GET IN.COUNT
```

4.29 IN-{IID}.NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET IN-100.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.NAME "ANALOG 1"
*GET IN-100.NAME

$> "SET IN-100.NAME "CD Player"" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET IN-100.NAME "CD Player"

$> "GET IN-100.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.NAME "CD Player"
*GET IN-100.NAME
```

4.30 IN-{IID}.SENS

TYPE: Register

METHODS: Get, Set

VALUES: Enumeration

- **14DBU** 14 DBU Sensitivity - Max input (ADC Clip) +24 DBU
- **4DBU** 4 DBU Sensitivity - Max input (ADC Clip) +14 DBU
- **-10DBV** -10 dBV Sensitivity - Max input (ADC Clip) +4 DBU
- **MIC** Max sensitivity for Microphone

Example:

```
$> "GET IN-100.SENS" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.SENS "4DBU"
*GET IN-100.SENS

$> "SET IN-100.SENS "-10DBV"" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET IN-100.SENS "-10DBV"

$> "GET IN-100.SENS" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.SENS "-10DBV"
*GET IN-100.SENS
```

4.31 IN-{IID}.GAIN

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Gain in dB. Range [-15.0 - 15.0], [-48, 0] for Generator

Example:

```
$> "GET IN-100.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.GAIN 0.000
*GET IN-100.GAIN

$> "SET IN-100.GAIN -4.0" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET IN-100.GAIN -4.0

$> "GET IN-100.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.GAIN -4.000
*GET IN-100.GAIN
```

4.32 IN-{IID}.STEREO

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

NOTES: Only valid for *PRIMARY* channels: 100, 102, 200. Error if other channel or generator

Example:

```
$> "GET IN-100.STEREO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.STEREO 0
*GET IN-100.STEREO

$> "SET IN-100.STEREO 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET IN-100.STEREO 1
```

```
$> "GET IN-100.STEREO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+IN-100.STEREO 1
*GET IN-100.STEREO
```

4.33 IN-{IID}.DYN.SIGNAL

TYPE: Subscription Only

VALUES: [Float] Signal level in dB. Range [-144 - 20]. -144 if no signal

NOTES: Updated every 50 ms.

Example:

```
$> "SUBSCRIBE" | websocat -t -n -0 -q ws://192.168.64.100/ws
*SUBSCRIBE
...
+IN-100.DYN.SIGNAL -73.4993
+IN-101.DYN.SIGNAL -72.8205
+IN-102.DYN.SIGNAL -101.728
+IN-103.DYN.SIGNAL -98.6826
+IN-200.DYN.SIGNAL -144
+IN-201.DYN.SIGNAL -144
```

4.34 IN-{IID}.DYN.CLIP

TYPE: Subscription Only

VALUES: Signal Clip. True when ADC is clipping [0 or 1]

NOTES: Updated every 50 ms.

Example:

```
$> "SUBSCRIBE" | websocat -t -n -0 -q ws://192.168.64.100/ws
*SUBSCRIBE
...
+IN-100.DYN.CLIP 0
+IN-101.DYN.CLIP 0
+IN-102.DYN.CLIP 0
+IN-103.DYN.CLIP 0
+IN-200.DYN.CLIP 0
```

4.35 ZONE.COUNT

TYPE: Register

METHODS: Get

VALUES: Integer

Example:

```
$> "GET ZONE.COUNT" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE.COUNT 2
*GET ZONE.COUNT
```

4.36 ZONE-{ZID}.NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET ZONE-A.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.NAME "ZONE A"
*GET ZONE-A.NAME

$> "SET ZONE-A.NAME "Bar"" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.NAME "Bar"

$> "GET ZONE-A.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.NAME "Bar"
*GET ZONE-A.NAME
```

4.37 ZONE-{ZID}.GAIN

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Gain in dB. Range [`ZONE-{ZID}.GAIN_MIN` - `ZONE-{ZID}.GAIN_MAX`]. Default [-80, 0]

NOTES: Read-Only if `ZONE-{ZID}.GPIO_VC` is set on zone

Example:

```
$> "GET ZONE-A.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN -40.00
*GET ZONE-A.GAIN

$> "SET ZONE-A.GAIN -20.0" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.GAIN -20.0

$> "GET ZONE-A.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN -20.000
*GET ZONE-A.GAIN
```

4.38 ZONE-{ZID}.GAIN_MIN

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Minimum Gain in dB. Range [-80.0 - `ZONE-{ZID}.GAIN_MAX`]

Example:

```
$> "GET ZONE-A.GAIN_MIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN_MIN -40.00
*GET ZONE-A.GAIN_MIN

$> "SET ZONE-A.GAIN_MIN -20.0" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.GAIN_MIN -20.0

$> "GET ZONE-A.GAIN_MIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN_MIN -20.000
*GET ZONE-A.GAIN_MIN
```

4.39 ZONE-{ZID}.GAIN_MAX

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Gain in dB. Range [`ZONE-{ZID}.GAIN_MIN` - 0.0]

Example:

```
$> "GET ZONE-A.GAIN_MAX" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN_MAX -40.00
*GET ZONE-A.GAIN_MAX

$> "SET ZONE-A.GAIN_MAX -20.0" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.GAIN_MAX -20.0

$> "GET ZONE-A.GAIN_MAX" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GAIN_MAX -20.000
*GET ZONE-A.GAIN_MAX
```

4.40 ZONE-{ZID}.MUTE

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

Example:

```
$> "GET ZONE-A.MUTE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.MUTE 0
*GET ZONE-A.MUTE
```

```
$> "SET ZONE-A.MUTE 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.MUTE 1

$> "GET ZONE-A.MUTE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.MUTE 1
*GET ZONE-A.MUTE
```

4.41 ZONE-{ZID}.PRIMARY_SRC

TYPE: Register

METHODS: Get, Set

VALUES: Input ID. See paragraph Input Channels

Example:

```
$> "GET ZONE-A.PRIMARY_SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.PRIMARY_SRC 100
*GET ZONE-A.PRIMARY_SRC

$> "SET ZONE-A.PRIMARY_SRC 100" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.PRIMARY_SRC 100

$> "GET ZONE-A.STEREO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.PRIMARY_SRC 100
*GET ZONE-A.PRIMARY_SRC 100
```

4.42 ZONE-{ZID}.STEREO

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

NOTES: Only valid for *PRIMARY* zones: ‘A’ and ‘C’. Error if *secondary* zone

Example:

```
$> "GET ZONE-A.STEREO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.STEREO 0
*GET ZONE-A.STEREO

$> "SET ZONE-A.STEREO 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.STEREO 1

$> "GET ZONE-A.STEREO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.STEREO 1
*GET ZONE-A.STEREO
```

4.43 ZONE-{ZID}.DYN.SIGNAL

TYPE: Subscription Only

VALUES: [Float] Signal level in dB. Range [-144 - 20]. -144 if no signal

NOTES: Updated every 50 ms.

Example:

```
$> "SUBSCRIBE" | websocat -t -n -0 -q ws://192.168.64.100/ws
*SUBSCRIBE
...
+ZONE-A.DYN.SIGNAL -100.358
+ZONE-B.DYN.SIGNAL -99.9367
```

4.44 ZONE-{ZID}.GPIO_VC

TYPE: Register

VALUES: VID or 0 of OFF

- **0** for OFF
- **1** for GPIO4
- **2** for GPIO5
- **3** for GPIO6
- **4** for GPIO7

NOTES: The register will not check if the GPIO pin is configured for Volume Control - which is required for the External Volume control to work.

Example:

```
$> "GET ZONE-A.GPIO_VC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GPIO_VC 0
*GET ZONE-A.GPIO_VC

$> "SET ZONE-A.GPIO_VC 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET ZONE-A.GPIO_VC 1

$> "GET ZONE-A.GPIO_VC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.GPIO_VC 1
*GET ZONE-A.GPIO_VC
```

4.45 ZONE-{ZID}.COMPRESSOR.AUTO

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

NOTES: Use automatic parameters for Attack, Release and Ratio based on crossover frequency

Example:

```
$> "GET ZONE-A.COMPRESSOR.AUTO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.COMPRESSOR.AUTO 1
*GET ZONE-A.COMPRESSOR.AUTO

$> "SET ZONE-A.COMPRESSOR.AUTO 0" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET ZONE-A.COMPRESSOR.AUTO 0

$> "GET ZONE-A.COMPRESSOR.AUTO" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.COMPRESSOR.AUTO 0
*GET ZONE-A.COMPRESSOR.AUTO
```

4.46 ZONE-{ZID}.COMPRESSOR.THRESHOLD

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Threshold for compressor in dBFS. Range [-40, 20]

Example:

```
$> "GET ZONE-A.COMPRESSOR.THRESHOLD" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+ZONE-A.COMPRESSOR.THRESHOLD 0.000
*GET ZONE-A.COMPRESSOR.THRESHOLD

$> "SET ZONE-A.COMPRESSOR.THRESHOLD -10" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET ZONE-A.COMPRESSOR.THRESHOLD -10

$> "GET ZONE-A.COMPRESSOR.THRESHOLD" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
+ZONE-A.COMPRESSOR.THRESHOLD -10.000
*GET ZONE-A.COMPRESSOR.THRESHOLD
```

4.47 ZONE-{ZID}.COMPRESSOR.ATTACK

TYPE: Register

METHODS: Get, Set

PATH:

VALUES: [Float] Attack Time for compressor in Seconds. Range [0.0003, 0.050]

Example:

```
$> "GET ZONE-A.COMPRESSOR.ATTACK" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+ZONE-A.COMPRESSOR.ATTACK 0.045
*GET ZONE-A.COMPRESSOR.ATTACK

$> "SET ZONE-A.COMPRESSOR.ATTACK 0.1" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET ZONE-A.COMPRESSOR.ATTACK 0.1

$> "GET ZONE-A.COMPRESSOR.ATTACK" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+ZONE-A.COMPRESSOR.ATTACK 0.100
*GET ZONE-A.COMPRESSOR.ATTACK
```

4.48 ZONE-{ZID}.COMPRESSOR.RELEASE

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Release Time for compressor in Seconds. Range [0.001, 1.0]

Example:

```
$> "GET ZONE-A.COMPRESSOR.RELEASE" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+ZONE-A.COMPRESSOR.RELEASE 0.750
*GET ZONE-A.COMPRESSOR.RELEASE

$> "SET ZONE-A.COMPRESSOR.RELEASE 0.8" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
*SET ZONE-A.COMPRESSOR.RELEASE 0.8

$> "GET ZONE-A.COMPRESSOR.RELEASE" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
+ZONE-A.COMPRESSOR.RELEASE 0.800
*GET ZONE-A.COMPRESSOR.RELEASE
```

4.49 ZONE-{ZID}.COMPRESSOR.RATIO

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Ratio for compressor. Range [1, 50]

Example:

```
$> "GET ZONE-A.COMPRESSOR.RATIO" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+ZONE-A.COMPRESSOR.RATIO 10.000
*GET ZONE-A.COMPRESSOR.RATIO
```

```
$> "SET ZONE-A.COMPRESSOR.RATIO 12" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
*SET ZONE-A.COMPRESSOR.RATIO 12

$> "GET ZONE-A.COMPRESSOR.RATIO" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+ZONE-A.COMPRESSOR.RATIO 12.000
*GET ZONE-A.COMPRESSOR.RATIO
```

4.50 ZONE-{ZID}.COMPRESSOR.HOLD

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Hold for compressor. Range [0, 1] Seconds

Example:

```
$> "GET ZONE-A.COMPRESSOR.HOLD" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.COMPRESSOR.HOLD 0.000
*GET ZONE-A.COMPRESSOR.HOLD

$> "SET ZONE-A.COMPRESSOR.HOLD 0.1" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
*SET ZONE-A.COMPRESSOR.HOLD 0.1

$> "GET ZONE-A.COMPRESSOR.HOLD" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+ZONE-A.COMPRESSOR.HOLD 0.100
*GET ZONE-A.COMPRESSOR.HOLD
```

4.51 ZONE-{ZID}.COMPRESSOR.KNEE

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Knee for compressor. Range [1, 12]

Example:

```
$>  printf "GET ZONE-A.COMPRESSOR.KNEE" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
+ZONE-A.COMPRESSOR.KNEE 4.000
*GET ZONE-A.COMPRESSOR.KNEE

$> "SET ZONE-A.COMPRESSOR.KNEE 5" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET ZONE-A.COMPRESSOR.KNEE 5

$>  printf "GET ZONE-A.COMPRESSOR.KNEE" | ncat 192.168.64.100 7621 --no-
```

```
shutdown -i 1
+ZONE-A.COMPRESSOR.KNEE 5.000
*GET ZONE-A.COMPRESSOR.KNEE
```

4.52 ZONE-{ZID}.COMPRESSOR.BYPASS

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean] Bypass compressor. Set to 0 to enable compressor, 1 to disable.

Example:

```
$>  printf 'GET ZONE-A.COMPRESSOR.BYPASS" | ncat 192.168.64.100 7621 --no-
shutdown -i 1
+ZONE-A.COMPRESSOR.BYPASS 1
*GET ZONE-A.COMPRESSOR.BYPASS

$> "SET ZONE-A.COMPRESSOR.BYPASS 0" | ncat 192.168.64.100 7621 --no-shutdown
-i 1
*SET ZONE-A.COMPRESSOR.THRESHOLD 0

$> "GET ZONE-A.COMPRESSOR.BYPASS" | ncat 192.168.64.100 7621 --no-shutdown -i
1
+ZONE-A.COMPRESSOR.BYPASS 0
*GET ZONE-A.COMPRESSOR.BYPASS
```

4.53 OUTPUT.COUNT

TYPE: Register

METHODS: Get

VALUES: [Integer]

Example:

```
$> "GET OUT.COUNT" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT.COUNT 2
*GET OUT.COUNT
```

4.54 OUT-{OID}.NAME

TYPE: Register

METHODS: Get, Set

VALUES: String (Max Length 32 chars)

Example:

```
$> "GET OUT-1.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.NAME "Output CH 1"
*GET OUT-1.NAME

$> "SET OUT-1.NAME "Left Speaker"" | ncat 192.168.64.100 7621 --no-shutdown -
i 1
*SET OUT-1.NAME "Left Speaker"

$> "GET OUT-1.NAME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.NAME "Left Speaker"
*GET OUT-1.NAME
```

4.55 OUT-{OID}.SRC

TYPE: Register

METHODS: Get, Set

VALUES: [STRING] - 'A' for *Zone A*, 'B' for *Zone B*...

NOTES: If source zone is stereo it is still possible to select Zone-B but as the value is 'invalid' as Zone-B is undefined when Zone-A is stereo (And links Zone-B) no sound will be playing. If source zone is stereo is is nessesary to set subchannel Source to play Left Channel, Right Channel or Sum of both channels.

Example:

```
$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.SRC "A"
*GET OUT-1.SRC

$> "SET OUT-1.SRC B" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUT-1.SRC B

$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.SRC "B"
*GET OUT-1.SRC
```

4.56 OUT-{OID}.SRC_CHANNEL

TYPE: Register

METHODS: Get, Set

VALUES: [ENUM]

- **L** - Left Channel Only
- **R** - Right Channel Only
- **S** - For Sum of Left and Right Channels

NOTES: If source zone is stereo is is nessesary to set subchannel Source to play Left Channel, Right Channel or Sum of both channels.

Example:

```
$> "GET OUT-1.SRC_CHANNEL" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.SRC_CHANNEL "S"
*GET OUT-1.SRC_CHANNEL

$> "SET OUT-1.SRC_CHANNEL L" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUT-1.SRC_CHANNEL L

$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.SRC_CHANNEL "L"
*GET OUT-1.SRC_CHANNEL
```

4.57 OUT-{OID}.POLARITY

TYPE: Register

METHODS: Get, Set

VALUES: Integer

1 - Normal Polarity **-1** - Reversed Polarity

Example:

```
$> "GET OUT-1.POLARITY" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.POLARITY 1
*GET OUT-1.POLARITY

$> "SET OUT-1.POLARITY -1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUT-1.POLARITY -1

$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.POLARITY -1
*GET OUT-1.POLARITY
```

4.58 OUT-{OID}.OUTPUT_MODE

TYPE: Register

METHODS: Get, Set

VALUES: ENUM

OFF - Output is Off **8R** - Output is LowZ **70V** - Output is HiZ 70 Volt **100V** - Output is HiZ 100 Volt **BTL** - Output is Bridged - *(Not supported for all models)*

Example:

```
$> "GET OUT-1.OUTPUT_MODE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.OUTPUT_MODE "8R"
*GET OUT-1.OUTPUT_MODE

$> "SET OUT-1.OUTPUT_MODE "100V"" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET OUT-1.OUTPUT_MODE "100V"

$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.OUTPUT_MODE "100V"
*GET OUT-1.OUTPUT_MODE
```

4.59 OUT-{OID}.OUTPUT_HIGHPASS

TYPE: Register

METHODS: Get, Set

VALUES: Float [20, 1000] Hz

Example:

```
$> "GET OUT-1.OUTPUT_HIGHPASS" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.OUTPUT_HIGHPASS 100.000
*GET OUT-1.OUTPUT_HIGHPASS

$> "SET OUT-1.OUTPUT_HIGHPASS 80" | ncat 192.168.64.100 7621 --no-shutdown -i
1
*SET OUT-1.OUTPUT_HIGHPASS 80

$> "GET OUT-1.SRC" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.OUTPUT_HIGHPASS 80.000
*GET OUT-1.OUTPUT_HIGHPASS
```

4.60 OUT-{OID}.GAIN

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Gain in dB. Range [-30.0 - 15.0]

Example:

```
$> "GET OUT-1.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUTPUT-A.GAIN 0
*GET OUTPUT-A.GAIN

$> "SET OUT-1.GAIN 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUTPUT-A.GAIN 1.0

$> "GET OUT-1.GAIN" | ncat 192.168.64.100 7621 --no-shutdown -i 1
```

```
+OUTPUT-A.GAIN 1.0
*GET OUTPUT-A.GAIN
```

4.61 OUT-{OID}.MUTE

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

Example:

```
$> "GET OUT-1.MUTE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUTPUT-A.MUTE 0
*GET OUTPUT-A.MUTE

$> "SET OUT-1.MUTE 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUTPUT-A.MUTE 1

$> "GET OUT-1.MUTE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUTPUT-A.MUTE 1
*GET OUTPUT-A.MUTE
```

4.62 OUT-{OID}.DYN.SIGNAL

TYPE: Subscription Only

VALUES: [Float] Signal level in dB. Range [-144 - 20]. -144 if no signal

NOTES: Updated every 50 ms.

Example:

```
$> "SUBSCRIBE" | websocat -t -n -0 -q ws://192.168.64.100/ws
*SUBSCRIBE
...
+OUT-1.DYN.SIGNAL -73.4993
+OUT-2.DYN.SIGNAL -72.8205
```

4.63 OUT-{OID}.DYN.CLIP

TYPE: Subscription Only

VALUES: Signal Clip. True when DAC is clipping [0 or 1]

NOTES: Updated every 50 ms.

Example:

```
$> "SUBSCRIBE" | websocat -t -n -0 -q ws://192.168.64.100/ws
*SUBSCRIBE
...
```

```
+OUT-1.DYN.CLIP 0
+OUT-2.DYN.CLIP 0
```

4.64 OUT-{OID}.DELAY.TIME

TYPE: Register

METHODS: Get, Set

VALUES: [Float] Time in seconds. Range [0.0, 0.1]

Example:

```
$> "GET OUT-1.DELAY.TIME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.DELAY.TIME 0.00000
*GET OUT-1.DELAY.TIME

$> "SET OUT-1.DELAY.TIME 0.01" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUT-1.DELAY.TIME 0.01

$> "GET OUT-1.DELAY.TIME" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.DELAY.TIME 0.01000
*GET OUT-1.DELAY.TIME
```

4.65 OUT-{OID}.DELAY.BYPASS

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

Example:

```
$> "GET OUT-1.DELAY.BYPASS" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.DELAY.BYPASS 1
*GET OUT-1.DELAY.BYPASS

$> "SET OUT-1.DELAY.BYPASS 0" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET OUT-1.DELAY.BYPASS 0

$> "GET OUT-1.DELAY.BYPASS" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+OUT-1.DELAY.BYPASS 0
*GET OUT-1.DELAY.BYPASS
```

4.66 GENERATOR.ENABLE

TYPE: Register

METHODS: Get, Set

VALUES: [Boolean]

Example:

```
$> "GET GENERATOR.ENABLE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+GENERATOR.ENABLE 0
*GET GENERATOR.ENABLE

$> "SET GENERATOR.ENABLE 1" | ncat 192.168.64.100 7621 --no-shutdown -i 1
*SET GENERATOR.ENABLE 1

$> "GET GENERATOR.ENABLE" | ncat 192.168.64.100 7621 --no-shutdown -i 1
+GENERATOR.ENABLE 1
*GET GENERATOR.ENABLE
```