M28 REMOTE CONTROL

Reference Manual

v2.2

Index

Change History

Ethernet Connection

The M28 integrates a 100Mbit/s Ethernet port allowing a full real-time remote control and monitor of the mixer processing functions.

A remote controller (e.g. a laptop) can connect the M28 in two different modes:

1) Point-To-Point mode

In Point-to-Point mode the remote controller directly connects the mixer through a standard ethernet cable, without any external router as shown below.

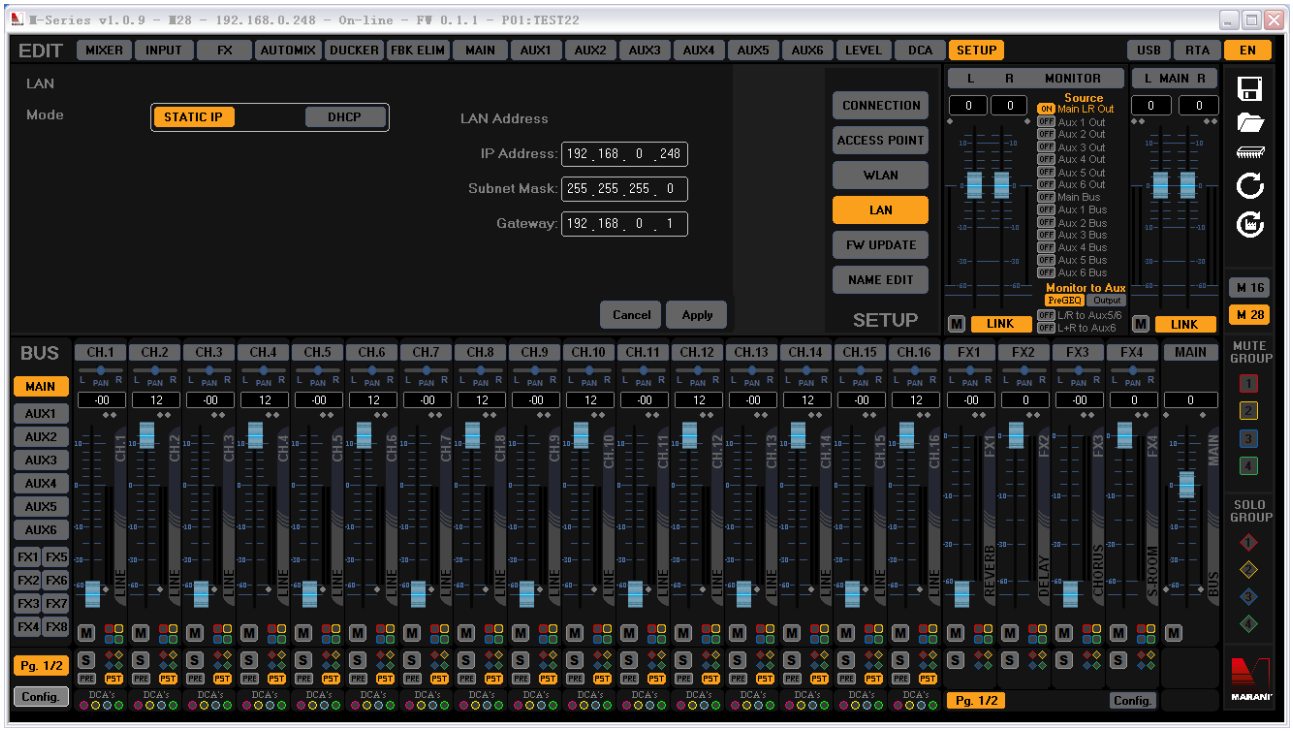
In this mode Static IP addressing should be used since the M28 Ethernet controller has no DHCP server capability.

<u>Connection Parameters (M28):</u>

- IP Addressing: Static
- IP Address: 192.168.0.100 (Default)
- Net Mask: 255.255.255.0 (Default)
- Gateway: 192.168.0.1 (Default)

<u>Connection Parameters (Remote Controller):</u>

- IP Addressing: Static
- IP Address: 192.168.0.xxx , where xxx is in the range [1,254] (and not equals to 100)
- Net Mask: 255.255.255.0
- Gateway: 192.168.0.1



2) LAN mode

In LAN mode both the M28 and the remote controller should be connected to an external router as shown below.

In this mode both Static and Dynamic IP addressing can be used since the M28 ethernet controller has DHCP client capability. Up to 10 remote controllers can connect the M28 at the same time.

<u>Connection Parameters (M28):</u>

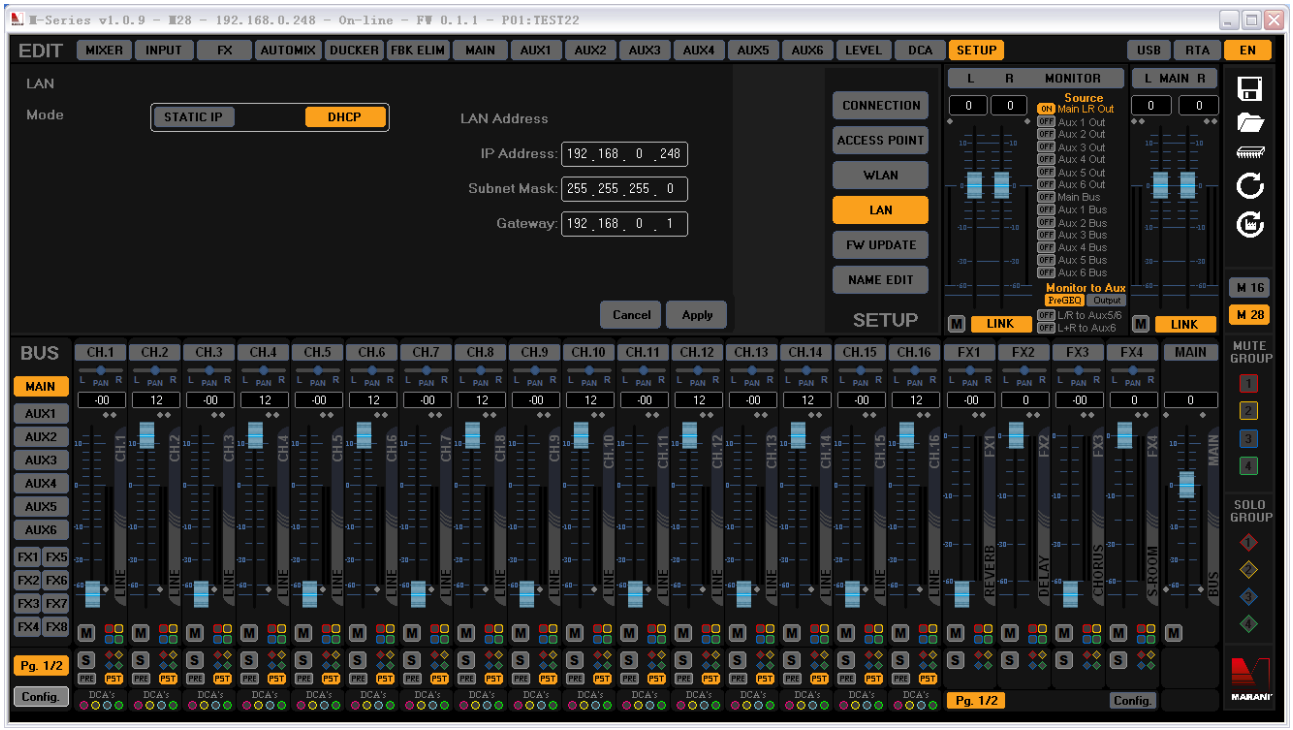
- IP Addressing: Dynamic (DHCP Client)
- IP Address: 192.168.0.100
- Net Mask: 255.255.255.0
- Gateway: 192.168.0.1

Note: IP Address, Net Mask and Gateway are automatically assigned by gthe DHCP server (i.e. the router). The default parameters above will be used only if the DHCP fails to assign the IP address.

Connection Parameters (Remote Controller):

- IP Addressing: Dynamic (DHCP Client)

Note: IP Address, Net Mask and Gateway are automatically assigned by the DHCP server (i.e. the router)



WiFi Connection

The M28 integrates a WiFi module for a full real-time remote control and monitor of the mixer processing functions.

The WiFi module can operate in two different modes:

1) Access Point mode

In Access Point mode the module acts as a WiFi router allowing to directly connect up to 4 clients (e.g. laptop, iPad, etc). The module includes a DHCP server to automatically assign the clients' IP addresses. This way, the remote control devices can easily connect the M28 mixer without any external WiFi router as shown below.

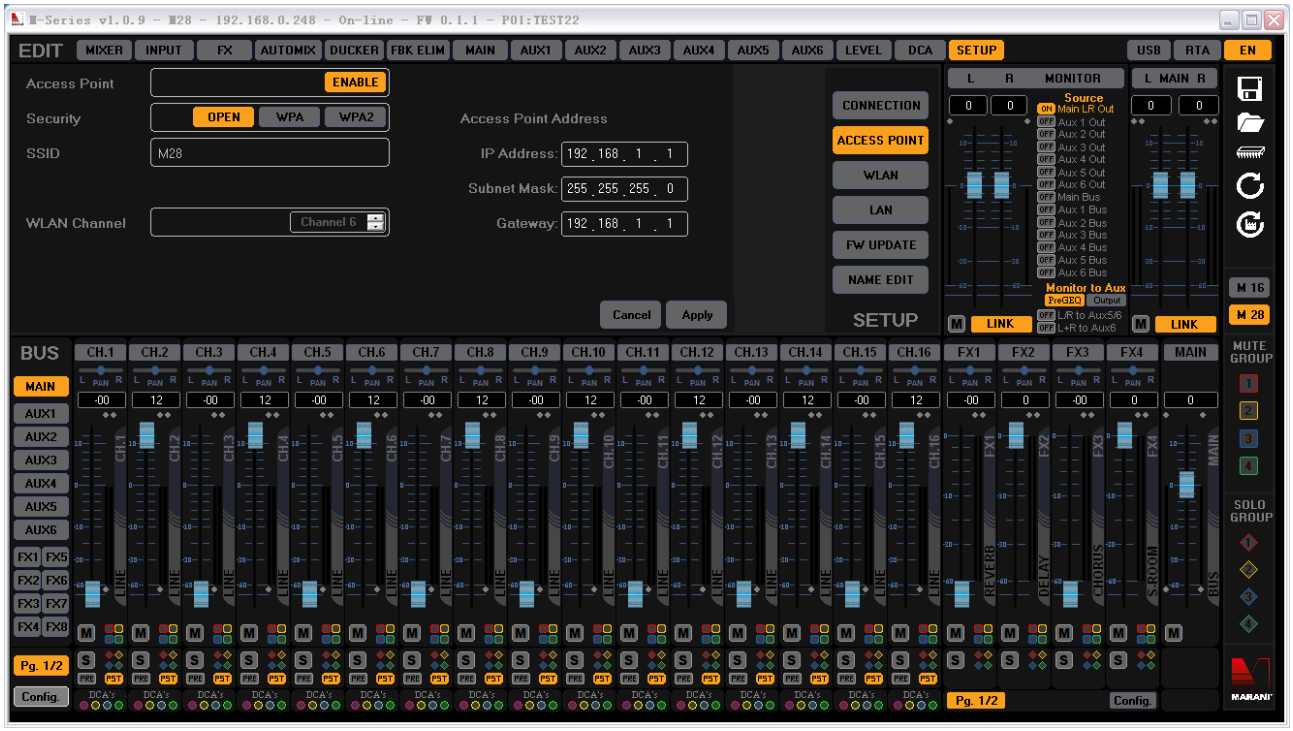
Connection Parameters (M28):

- IP Addressing: Dynamic (DHCP Server)
- IP Address: 192.168.0.1 (Default)
- Net Mask: 255.255.255.0 (Default)
- Gateway: 192.168.0.1 (Default)
- SSID: M28 (Default). WiFi network name
- Security: OPEN (Default). WPA/WPA2 supported
- Key: “0123456789” (Default). Used only with WPA/WPA2 encryption
- WiFi Channel: 6 (Default). WiFi radio channel.

Connection Parameters (Remote Controller):

- IP Addressing: Dynamic (DHCP Client)

Note: IP Address, Net Mask and Gateway are automatically assigned by the DHCP server (i.e. the M28 WiFi module)



2) WiFi Client (Station) mode

In WiFi Client mode the module connects to a WiFi network and supports DHCP client or static IP addressing. This way, both the M28 mixer and the remote control devices should be connected to an external WiFi router as shown below.
WiFi Client mode may be used to extend the WiFi range of the M28 mixer.

<u>Connection Parameters (M28):</u>

- IP Addressing: Dynamic (DHCP Client) or Static
- IP Address: 192.168.0.110 (Default). Used with Static IP addressing only
- Net Mask: 255.255.255.0 (Default). Used with Static IP addressing only
- Gateway: 192.168.0.1 (Default). Used with Static IP addressing only

Note: IP Address, Net Mask and Gateway are automatically assigned by the DHCP server (i.e. the WiFi router) if the Dynamic IP addressing is selected.

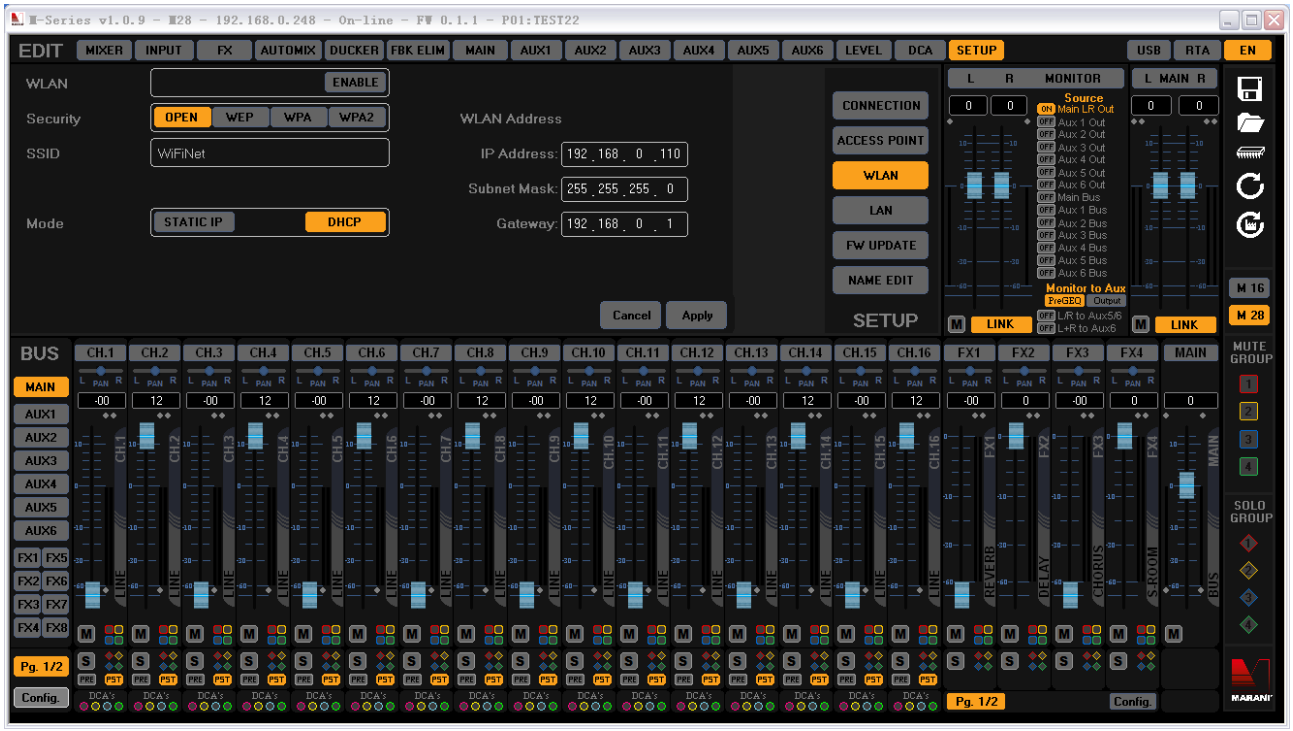
Connection Parameters (WiFi Router):

- SSID: WiFi Router network name
- Security: OPEN, WEP, WPA or WPA2
- Key: xxxxxxxx. Password of the WiFi router network

Connection Parameters (Remote Controller):

- IP Addressing: Dynamic (DHCP Client)

Note: IP Address, Net Mask and Gateway are automatically assigned by the DHCP server (i.e. the WiFi router)



Mixed Ethernet / WiFi Connection Examples

1) WiFi Remote Controllers using M28 Ethernet Port

2) Mixed WiFi/Ethernet Remote Controllers using M28 Ethernet Port and WiFi Module

Remote Control Communication Protocol

The communication between the M28 and a remote controller is based on a standard client/server TCP protocol. The M28 is acting as a server both for ethernet and WiFi connections, while the remote controller is the client.

The TCP server that runs on the M28 is always listening on TCP Port 1001 and allows to connect up to 10 clients through the Ethernet port and up to 4 concurrent clients through the Built-in WiFi module.

Remote Control Commands

The M28 and the remote controllers use a custom variable length command/response protocol to exchange data. The command and response format are described here below.

Command (from the remote controller)

STX	ID_M	ID_N	CMD	D0	D1	D2	---	DN-1	ETX
F0H	18H	xx	Cmd1	d0	d1	d2	---	dn-1	F7H

where:
- STX: Start of message. Always equals to F0 (Hex)
- ID_M: Device ID. Always equals to 16 (Hex)
- ID_N: Connection ID
- CMD: Command ID
- D0...DN-1: Command Data
- ETX: End of message. Always equals to F7 (Hex)

Note 1: All the command fields are 1 byte long (uint8 type).
For example with a data lenght N equals to 8 bytes, the total command length is 13 bytes (8 data fields + 5 control fields).

Note 2: The ID_N field (connection ID) is a number from 1 to 255 that is assigned by the mixer in the CMD_CONNECTION_OPEN response to identify the remote client. All the commands sent by the client after the CMD_CONNECTION_OPEN must contain the ID_N number assigned to the connection.

Response (from the M28)

STX	ID_M	ID_N	CMD	D0	D1	D2	---	DM-1	ETX
F0H	18H	Xx	Cmd1	d0	d1	d2	---	dm-1	F7H

where:
- STX: Start of message. Always equals to F0 (Hex)
- ID_M: Device ID. Always equals to 18 (Hex)
- ID_N: Connection ID
- CMD: Command ID
- D0...DM-1: Response Data
- ETX: End of message. Always equals to F7 (Hex)

Note 3: All the response fields are 1 byte long (uint8 type).
For example with a data lenght M equals to 8 bytes, the total response length is 13 bytes (8 data fields + 5 control fields).

Note 4: The ID_N field (connection ID) in the command response is a number from 1 to 255. The mixer returns the ID of the connection that changed the last parameter or it echoes the command ID_N field if no parameter is changed. The client has to check this field every time it receives a command response: if the returned ID_N is different from the ID_N sent in the command, it means another remote client changed a parameter and the actual client needs to resync the GUI with the new settings (by reading the device memory) or it has to simply inform the user that the GUI is not longer “in sync” with the mixer.

Note 5: the response length can be different from the corresponding command length.

Ack (from the remote controller) - Optional

ACK
00H

After receiving the response, the remote controller should always send the ACK = 00 (Hex). This allows a faster data exchange when concurrent WiFi connections are established.

Here below the remote control commands are described.

CMD_CONNECTION_OPEN (00H)

The remote controller should send this command only once and just after connecting to get device info and connection ID.

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	00H	00H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (30 bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	00H	IDM	IF	UDID1	UDID2	UDID3	UDID4

D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
UDID5	UDID6	Model	Name1	Name2	Name3	Name4	Name5	Name6	Name7

D16	D17	D18	D19	D20	D21	D22	D23	D24	ETX
Name8	Name9	Name10	Name11	Name12	Name13	Name14	Name15	Name16	F7H

where:
- ID_N = connection ID in the range [1,255].
- IDM = 18H
- IF: interface (0:Ethernet; 1:WiFi)
- UDID1-6: Unique Device ID
    - Example:
    - UDID1=12H; UDID2=34H; UDID3=56H; UDID4=78H; UDID5=9AH; UDID6=BCH
    - Unique Device ID = “123456789ABC”
- Model: Device model / extension card type (0: M16 Analog extension; 1: M16 Dante extension; 2: M16 SPDIF/Link extension ; 16: M28 Analog extension; 17: M28 Dante extension; 18: M28 SPDIF/Link extension)
- Name1-16: Device name string (16 chars)

Note: All the commands sent by the client after the CMD_CONNECTION_OPEN must contain the ID_N number returned in CMD_CONNECTION_OPEN command response.

CMD_SET_GAIN (01H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	01H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the Gain to set (e.g. 0:SetInpGain; 1: SetInpMasterGain; ...)
- Value: the Gain value (the range depends on type parameter)
- SrcCh: source channel (the range and meaning depend on type parameter)
- DestCh: destination channel (the range and meaning depend on type parameter)

Please refer to the table below

Type	Value Range	SrcCh Range	DestCh Range	Description
0	[0,72]	[0,29]	NotUsed	Input channel gain/sensitivity (analog) *Refer to "InpGain" param in the MemoryMap doc*
1	[0,191]	[0,29]	NotUsed	Input channel master gain (i.e. the master fader) *Refer to "InpMasterGain" param in the MemoryMap doc*
2	[0,60]	[0,29]	NotUsed	Input to MainLR bus pan control *Refer to "InpMasterSendPan" param in the MemoryMap doc*
4	[0,131]	[0,15]	[0,7]	Input to Efx1-8 send gain *Refer to "InpEfxSendGain" param in the MemoryMap doc*
5	[0,131]	[0,29]	[0,5]	Input to Aux1-6 send gain *Refer to "InpAuxSendGain" param in the MemoryMap doc*
10	[0,191]	[0,7]	NotUsed	Efx1-8 gain *Refer to "EfxGain" param in the MemoryMap doc*
11	[0,131]	[0,7]	NotUsed	Efx1-8 to MainLR bus gain (i.e. Efx1-8 return on MainLR bus) *Refer to "EfxMasterSendGain" param in the MemoryMap doc*
12	[0,60]	[0,7]	NotUsed	Efx1-8 to MainLR pan control *Refer to "EfxMasterSendPan" param in the MemoryMap doc*
13	[0,131]	[0,7]	[0,5]	Efx1-8 to Aux1-6 bus gain (i.e. Efx1-8 return on Aux1-6 bus) *Refer to "EfxAuxSendGain" param in the MemoryMap doc*
15	[0,191]	NotUsed	NotUsed	MainLR bus gain *Refer to "MasterGain" param in the MemoryMap doc*

16	[0,191]	[0,1]	NotUsed	MainLR output gain (SrcCh=0:MainL; SrcCh=1:MainR) *Refer to "MasterOutGain" param in the MemoryMap doc*
18	[0,191]	[0,5]	NotUsed	Aux1-6 bus gain *Refer to "AuxGain" param in the MemoryMap doc*
19	[0,191]	[0,5]	NotUsed	Aux1-6 output gain *Refer to "AuxOutGain" param in the MemoryMap doc*
26	[0,191]	[0,1]	NotUsed	MonitorLR output gain *(SrcCh=0:MonL; SrcCh=1:MonR)* *Refer to "MonGainL" and "MonGainR params in the MemoryMap doc*
28	[0,191]	[0,3]	NotUsed	DCA1-4 gain *Refer to "DcaAflGain" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	01H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_MUTE (02H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	02H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the parameter to set (e.g. 0:SetInpMasterMute)
- Value: 0: Unmuted; 1:Muted
- SrcCh: source channel (the range and meaning depend on type parameter)
- DestCh: destination channel (the range and meaning depend on type parameter)

Please refer to the table below

Type	Value Range	SrcCh Range	DestCh Range	Description
0	[0,1]	[0,29]	NotUsed	Input channel master mute *Refer to "InpMasterMute" param in the MemoryMap doc*
8	[0,1]	[0,29]	NotUsed	Input to Solo bus send mute *Refer to "InpSoloSendMute" param in the MemoryMap doc*
12	[0,1]	[0,7]	NotUsed	Efx1-8 Mute *Refer to "EfxMute" param in the MemoryMap doc*
13	[0,1]	[0,7]	NotUsed	Efx1-8 to MainLR bus send mute (i.e. Efx1-8 return on MainLR bus) *Refer to "EfxMasterSendMute" param in the MemoryMap doc*
14	[0,1]	[0,7]	[0,5]	Efx1-8 to Aux1-6 bus send mute (i.e. Efx1-8 return on Aux1-6 bus) *Refer to "EfxAuxSendMute" param in the MemoryMap doc*
18	[0,1]	[0,7]	NotUsed	Efx1-8 to Solo Bus send mute *Refer to "EfxSoloSendMute" param in the MemoryMap doc*
19	[0,1]	NotUsed	NotUsed	MainLR bus mute *Refer to "MasterMute" param in the MemoryMap doc*
20	[0,1]	NotUsed	NotUsed	MainLR output mute *Refer to "MasterOutMute" param in the MemoryMap doc*
23	[0,1]	[0,5]	NotUsed	Aux1-6 bus mute *Refer to "AuxMute" param in the MemoryMap doc*
24	[0,1]	[0,5]	NotUsed	Aux1-6 output mute *Refer to "AuxOutMute" param in the MemoryMap doc*
33	[0,1]	NotUsed	NotUsed	Monitor output mute *Refer to "MonMute" param in the MemoryMap doc*

35	[0,1]	[0,3]	NotUsed	DCA1-4 mute *Refer to "DcaAflMute" param in the MemoryMap doc*
37	[0,1]	[0,29]	[0,3]	DCA1-4 channel assignment/enable (0: Disabled; 1:Enabled / assigned) *Refer to "InpDcaAflEnable" param in the MemoryMap doc*
38	[0,1]	[0,3]	NotUsed	DCA1-4 to Solo bus send mute *Refer to "DcaAflSoloSendMute" param in the MemoryMap doc*
39	[0,1]	[0,29]	NotUsed	Input channel post fader mute *Refer to "InpPstMute" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	02H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_PAFL_SEL (03H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	03H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the parameter to set
- Value: 0: PRE (Pre Fader); 1:PST (Post Fader)
- SrcCh: source channel (the range and meaning depend on type parameter)
- DestCh: destination channel (the range and meaning depend on type parameter)

Please refer to the table below

Type	**Value Range**	**SrcCh Range**	**DestCh Range**	**Description**
1	[0,1]	[0,29]	[0,7]	Input to Efx1-8 send PRE/PST sel *Refer to "InpEfxSendPafl" param in the MemoryMap doc*
2	[0,1]	[0,29]	[0,5]	Input to Aux1-6 send PRE/PST sel *Refer to "InpAuxSendPafl" param in the MemoryMap doc*
3	[0,1]	[0,29]	NotUsed	Input to Solo bus send PRE/POST sel *Refer to "InpSoloSendPafl" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	03H	Type	Value	SrcCh	DestCh	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_POLARITY (04H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	04H	Type	Value	SrcCh	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the parameter to set (e.g. 0:SetInpPolarity)
- Value: 0: Polarity Normal; 1:Polarity Inverted
- SrcCh: source channel (the range and meaning depend on type parameter)

Please refer to the table below

Type	Value Range	SrcCh Range	Description
0	[0,1]	[0,29]	Input channel polarity *Refer to "InpPolarity" param in the MemoryMap doc*
1	[0,1]	NotUsed	MainLR output polarity *Refer to "MasterPolarity" param in the MemoryMap doc*
2	[0,1]	[0,5]	Aux1-6 output polarity *Refer to "AuxPolarity" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	04H	Type	Value	SrcCh	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_DELAY (05H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	05H	Type	SrcCh	DelayH	DelayL	DelayF	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the parameter to set (e.g. 0:SetInpDelay)
- SrcCh: source channel (the range and meaning depend on type parameter)
- DelayH: delay (high byte)
- DelayL: delay (low byte)
- DelayF: delay fine

Channel delay [ms] = DelayH*256 + DelayL + 0.05*DelayF

Please refer to the table below

Type	**SrcCh Range**	**DelayH**	**DelayL**	**DelayF**	**Description**
0	[0,29]	0	[0,15]	[0,19]	Input channel delay [ms] *Refer to "InpDelayH", "InpDelayL", "InpDelayFine" param in the MemoryMap doc*
1	NotUsed	0	[0,150]	[0,19]	MainLR output delay [ms] *Refer to "MasterDelayH", "MasterDelayL", "MasterDelayFine" param in the MemoryMap doc*
2	[0,5]	0	[0,100]	[0,19]	Aux1-6 output delay [ms] *Refer to "AuxDelayH", "AuxDelayL", "AuxDelayFine" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	05H	Type	SrcCh	DelayH	DelayL	DelayF	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_NOISE_GATE (06H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	06H	00H	SrcCh	Thr	Rel	Atk	Hold

D6	D7	ETX
00H	00H	F7H

where:
- SrcCh: input channel [0,29]
- Thr: noise gate threshold/bypass (refer to “InpNoiseGateThr” param in the MemoryMap doc
  - - bit0-bit6 --> threshold [0,7]
  - - bit7 --> bypass (0:Normal; 1:Bypassed)
- Rel: noise gate release time (refer to “InpNoiseGateRel” param in the MemoryMap doc
- Atk: noise gate attack time (refer to “InpNoiseGateAtk” param in the MemoryMap doc
- Hold: noise gate hold time (refer to “InpNoiseGateHold” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	06H	00H	SrcCh	Thr	Rel	Atk	Hold

D6	D7	ETX
00H	00H	F7H

CMD_SET_COMPRESSOR (07H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	07H	Type	SrcCh	Ratio	Knee	Atk	Rel

D6	D7	ETX
Thr	00H	F7H

where:
- Type: select the compressor to set
  - 0: Set input channel compressor
  - 1: Set MainLR output compressor
  - 2: Set Aux1-6 output compressor
- SrcCh: source channel
  - if Type = 0, SrcCh = [0,29]
  - if Type = 1, SrcCh = NotUsed
  - if Type = 2, SrcCh = [0,5]
- Ratio: compressor ratio/bypass (refer to “InpCompressorRatio”, “MasterCompressorRatio”, “AuxCompressorRatio” param in the MemoryMap doc
  - - bit0-bit6 --> ratio [0,31]
  - - bit7 --> bypass (0:Normal; 1:Bypassed)
- Knee: compressor knee (refer to “InpCompressorKnee”, “MasterCompressorKnee”, “AuxCompressorKnee” param in the MemoryMap doc
- Atk: noise gate attack time (refer to “InpCompressorAtk”, “MasterCompressorAtk”, “AuxCompressorAtk” param in the MemoryMap doc
- Rel: noise gate release time (refer to “InpCompressorRel”, “MasterCompressorRel”, “AuxCompressorRel” param in the MemoryMap doc
- Thr: noise gate threshold (refer to “InpCompressorThr”, “MasterCompressorThr”, “AuxCompressorThr” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	07H	Type	SrcCh	Ratio	Knee	Atk	Rel

D6	D7	ETX
Thr	00H	F7H

CMD_SET_PEQ_BYPASS (08H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	08H	Type	SrcCh	Bypass	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the parameter to set
- SrcCh: source channel (the range and meaning depend on type parameter)
- Bypass: PEQ bypass (0:Active; 1:Bypassed)

Please refer to the table below

Type	**SrcCh Range**	**Bypass**	**Description**
0	[0,29]	[0,1]	Input channel PEQ bypass *Refer to "InpPeqBypass" param in the MemoryMap doc*
1	NotUsed	[0,1]	MainLR PEQ bypass *Refer to "MasterPeqBypass" param in the MemoryMap doc*
2	[0,5]	[0,1]	Aux1-6 PEQ bypass *Refer to "AuxPeqBypass" param in the MemoryMap doc*
3	[0,7]	[0,1]	Efx1-8 PEQ bypass *Refer to "EfxPeqBypass" param in the MemoryMap doc*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	08H	Type	SrcCh	Bypass	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_PEQ_FLAT (09H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	09H	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the PEQ to flatten
- SrcCh: source channel (the range and meaning depend on type parameter)

Please refer to the table below

Type	**SrcCh Range**	**Description**
0	[0,29]	Flatten input channel PEQ
1	NotUsed	Flatten MainLR PEQ
2	[0,5]	Flatten Aux1-6 PEQ
3	[0,29]	Flatten input channel Mid-Morph PEQ

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	09H	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_PEQ_FILTER (0AH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	Type	SrcCh	FiltNum	FiltType	Gain	FreqH

D6	D7	ETX
FreqL	Q	F7H

where:
- Type: select the PEQ to set
- SrcCh: source channel (the range and meaning depend on type parameter)
- FiltNum: PEQ filter number (the range and meaning depend on type parameter)
- FiltType: PEQ filter type (0:Peak; 1:Hi-Shv; 2:Lo-Shv)
- Gain: Bit0..6 PEQ filter gain = [0,48]; Bit7 bypass (0: filter enabled; 1: filter bypassed)
- Q; PEQ filter Q = [0,197]
- FreqH: PEQ filter frequency (high byte)
- FreqL: PEQ filter frequency (low byte)

PEQ filter frequency [Hz] = FreqH*256 + FreqL

Please refer to the table below

Type	**SrcCh Range**	**FiltNum**	**Description**
0	[0,29]	[0,3]	Set input channel PEQ filter *Refer to "InpPeqFilterType", "InpPeqFilterFreqH", "InpPeqFilterFreqL", "InpPeqFilterGain", "InpPeqFilterQ" param in the MemoryMap doc*
1	NotUsed	[0,7]	Set MainLR output PEQ filter *Refer to "MasterPeqFilterType", "MasterPeqFilterFreqH", "MasterPeqFilterFreqL", "MasterPeqFilterGain", "MasterPeqFilterQ" param in the MemoryMap doc*
2	[0,5]	[0,2]	Set Aux1-6 output PEQ filter *Refer to "AuxPeqFilterType", "AuxPeqFilterFreqH", "AuxPeqFilterFreqL", "AuxPeqFilterGain", "AuxPeqFilterQ" param in the MemoryMap docSet input channel PEQ filter*
3	[0,29]	[0,3]	Set input channel Mid-Morph PEQ filter *Refer to "InpMidMorphPeqFilterGain" param in the MemoryMap doc*
4	[0,1,2, 4,5,6]	[0,3] if SrcCh = 0,4 0 if SrcCh = [1,2,5,6]	Set Efx1-3; Efx5-7 PEQ Filter *Refer to "EfxXPeqFilterType", "EfxXPeqFilterFreqH", "EfxXPeqFilterFreqL", "EfxXPeqFilterGain", "EfxXPeqFilterQ" param in the MemoryMap docSet input channel PEQ filter*

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	Type	SrcCh	FiltNum	FiltType	Gain	FreqH

D6	D7	ETX
FreqL	Q	F7H

Input Mid-Morph EQ

The Mid-Morph EQ can be selected on each input channel using the CMD_SET_INP_PEQ_TYPE command. Once the Mid-Morph is active, the 4 parametric filters work with fixed type, frequency and Q as specified here below:

- Filter 0:
  - Type = Lo-Shv
  - Freq = 150 Hz
  - Q = 1.0
  - Gain: from -12 to +12dB step 0.5dB

- Filter 1:
  - Type = Peak
  - Freq = 250 Hz
  - Q = 1.2
  - Gain: from -12 to 0dB step 0.5dB

- Filter 2:
  - Type = Peak
  - Freq = 4000 Hz
  - Q = 1.2
  - Gain: from 0 to +12dB step 0.5dB

- Filter 3:
  - Type = Hi-Shv
  - Freq = 4000 Hz
  - Q = 1.0
  - Gain: from -12 to +12dB step 0.5dB

To set the 4 filters gain and bypass, in Mid-Morph mode, the remote controller should send the CMD_SET_PEQ_FILTER command with the format described above. The M28 firmware will ignore the FiltType, FreqH, FreqL, Q parameters.

The remote control GUI should implement filters 1 and 2 as a single “mid” filter (with a single gain and bypass control). The remote controller should select filter 1 and filter 2 depending on the gain value of the “mid” filter:

- Filter 1 should be used for “mid” filter gain values in the range [-12dB, 0dB]; Filter 2 should be

set with flat gain (0dB)

- Filter 2 should be used for “mid” filter gain values in the range [0.5dB, 12dB]; Filter 1 should be set with flat gain (0dB)

As a result, when the user changes the “mid” filter gain or bypass the remote controller must send 2 CMD_SET_PEQ_FILTER commands (one for the filter 1 and the other for the filter 2) as in the following examples:

- *EXAMPLE 1: “Mid” filter gain = -12dB (bypass OFF)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	00H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	00H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	18H	NotUsed

D6	D7	ETX

NotUsed	NotUsed	F7H

- *EXAMPLE 2: "Mid" filter gain = +12dB (bypass OFF)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	30H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	30H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

- *EXAMPLE 3: "Mid" filter gain = 0dB (bypass OFF)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	18H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

- *EXAMPLE 4: "Mid" filter gain = -12dB (bypass ON)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	80H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	80H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

- *EXAMPLE 5: "Mid" filter gain = +12dB (bypass ON)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	B0H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	B0H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

- *EXAMPLE 6: "Mid" filter gain = 0dB (bypass ON)*

Command (13 Bytes) – Filter 1

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	01H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Command (13 Bytes) – Filter 2

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0AH	03H	SrcCh	02H	NotUsed	98H	NotUsed

D6	D7	ETX
NotUsed	NotUsed	F7H

CMD_SET_GEQ_BYPASS (0BH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0BH	Type	SrcCh	Bypass	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the GEQ to set (0: MainLR GEQ; 1:Aux1-6 GEQ)
- SrcCh: source channel (NotUsed if Type = 0; [0,5] if Type = 1)
- Bypass: GEQ bypass (0:Active; 1:Bypassed)

Please refer to “MasterGeqBypass” and “AuxGeqBypass” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0BH	Type	SrcCh	Bypass	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_GEQ_FLAT (0CH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0CH	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the QEQ to flatten (0: MainLR GEQ; 1:Aux1-6 GEQ)
- SrcCh: source channel (NotUsed if Type = 0; [0,5] if Type = 1)

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0CH	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_GEQ_FILTER (0DH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0DH	Type	SrcCh	Band	Gain	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the GEQ to set (0: MainLR GEQ; 1:Aux1-6 GEQ)
- SrcCh: source channel (NotUsed if Type = 0; [0,5] if Type = 1)
- Band: select the GEQ filter [0,30]
- Gain: GEQ filter gain [0,48]

Please refer to “MasterGeqFilterGain” and “AuxGeqFilterGain” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0DH	Type	SrcCh	Band	Gain	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_HIGHPASS (0EH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0EH	00H	SrcCh	Enable	FreqH	FreqL	00H

D6	D7	ETX
00H	00H	F7H

where:
- SrcCh: input channel [0,29]
- Enable: input highpass filter enable (0:Disabled; 1:Enabled)
- FreqH: input highpass filter frequency (high byte)
- FreqL: input highpass filter frequency (low byte)

Filter frequency [Hz] = FreqH*256 + FreqL

Please refer to “InpHighPassEnable”, “InpHighPassFreqH” and “InpHighPassFreqL” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0EH	00H	SrcCh	Enable	FreqH	FreqL	00H

D6	D7	ETX
00H	00H	F7H

CMD_CONNECTION_CLOSE (0FH)

The remote controller should send this command just before disconnecting.

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0FH	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	0FH	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_READ_MEMORY (10H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	10H	AddrH	AddrL	nBytesH	nBytesL	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- AddrH: memory address (high byte)
- AddrL: memory address (low byte)
- nBytesH: number of bytes to read (high byte)
- nBytesL: number of bytes to read (low byte)

Addr = AddrH*256 + AddrL
nBytes (N) = nBytesH*256 + nBytesL (nBytes should be in the range [1,256]

Response (N+5 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	10H	Data0	Data1	Data2	Data3	Data4	Data5

...	DN-1	ETX
...	DataN-1	F7H

CMD_WRITE_MEMORY (11H)

Command (N+9 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	11H	AddrH	AddrL	nBytesH	nBytesL	Data0	Data1

...	DN-1	ETX
...	DataN-1	F7H

where:
- AddrH: memory address (high byte)
- AddrL: memory address (low byte)
- nBytesH: number of bytes to write (high byte)
- nBytesL: number of bytes to write (low byte)

Addr = AddrH*256 + AddrL
nBytes (N) = nBytesH*256 + nBytesL (nBytes should be in the range [1,256]

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	11H	AddrH	AddrL	nBytesH	nBytesL	Data0	Data1

D6	D7	ETX
Data2	Data3	F7H

CMD_SET_SRC_TYPE (12H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	12H	00H	SrcCh	SrcType	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- SrcCh: input channel [0,29]
- SrcType: channel source
  - 0: LINE; 1: MIC (if SrcCh = [0,15])
  - 0 (if SrcCh = [16,23])
  - 0: LINE/Dante (Ana or Dante); (if SrcCh = [24,27]) //not send
  - USB (if SrcCh = [28,29]) //not send

Please refer to “InpSrc” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	12H	00H	SrcCh	SrcType	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_DYNAMIC_FILTER (13H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	13H	00H	00H	Bypass	LowFreqBoost	HighFreqBoost	00H

D6	D7	ETX
00H	00H	F7H

where:
- Bypass: MainLR dynamic filter bypass (0:Active; 1:Bypassed)
- LowFreqBoost: dynamic filter low frequency boost [%]
- HighFreqBoost: dynamic filter high frequency boost [%]

Please refer to “MasterDynamicFilterBypass”, “MasterDynamicFilterLowFreqBoost” and “MasterDynamicFilterHighFrequencyBoost” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	13H	00H	00H	Bypass	LowFreqBoost	HighFreqBoost	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_PHANTOM (14H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	14H	00H	SrcCh	Phantom	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- SrcCh: input channel [0,15]
- Phantom: phatom power (+48V) enable (0: Disabled; 1: Enabled)

Please refer to “InpPhantom” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	14H	00H	SrcCh	Phantom	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_LINK (15H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	15H	Type	SrcCh	Link	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type:
  - 0 Inp Link
  - 1: Efx Link
- SrcCh:
  - if type=0: input stereo channel [0,15] (0: CH1-2; 1: CH3-4; … ; 15: CH27:28)
  - if type=1: Effect stereo channel [0,3] (0: FX1-4; 1: FX2-5; … ; 3: FX4-8)
- Link: stereo link enable (0: Disabled; 1: Enabled)

Example:
- Type = 0
- SrcCh = 0
- Link = 1

CH1-2 Link enabled (CH1: left channel; CH2: right channel)

When a link is enabled on a channel pair the left channel parameters is copied to the right channel. The channel pan is set to have CHX on the left and CHX+1 on the right,

Please refer to “InpLink” and “EfxLink” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	15H	Type	SrcCh	Link	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_RELOAD_PRESET (16H)

This command allows to reload all DSP parameters. To load a preset file, the remote controller should write all the preset parameters into the device memory (using the command CMD_WRITE_MEMORY) and then it should send the CMD_RELOAD_PRESET command to update the DSP with the new parameter set.

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	16H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	16H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_RESET_PRESET (17H)

This command allows to reinitialize the device preset memory (and the DSP) with the factory default parameters and reloads all DSP parameters. After sending the command the remote controller should read the device memory to get the default parameter values.

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	17H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	17H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_MON_TO_AUX34 (19H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	19H	00H	00H	Mode	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Mode: [0,2]
  - 0: Normal Mode
  - 1: Monitor Left to Aux5 and Monitor Righr to Aux6
  - 2: (Monitor Left + Monitor Right) / 2 to Aux6

Please refer to “MonToAux56” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	19H	00H	00H	Mode	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_GET_VUMETER (1AH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1AH	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (600+5 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1AH	Data0	Data1	Data2	Data3	Data4	Data5

...	D285	ETX
...	Data599	F7H

where:
- Data0 … Data127, Data128,..,Data133, Data138… Data557: VuMeter
    - Bit6...Bit0: signal value in the range [0,120] (from -60dBFS to 0dBFS step 0.5dBFS)
    - Bit7: signal clip or clip limiter activity (0: normal; 1: signal clipped or clip limiter active)
- Data135: noise gate activity (CH25-CH30)
    - Bit0: CH25 noise gate (0: inactive; 1: active)
    - Bit1: CH26 noise gate (0: inactive; 1: active)
    - Bit2: CH27 noise gate (0: inactive; 1: active)
    - Bit3: CH28 noise gate (0: inactive; 1: active)
    - Bit4: CH29 noise gate (0: inactive; 1: active)
    - Bit5: CH30 noise gate (0: inactive; 1: active)
    - Bit6:
    - Bit7:
- Data135: noise gate activity (CH17-CH24)
    - Bit0: CH17 noise gate (0: inactive; 1: active)
    - Bit1: CH18 noise gate (0: inactive; 1: active)
    - Bit2: CH19 noise gate (0: inactive; 1: active)
    - Bit3: CH20 noise gate (0: inactive; 1: active)
    - Bit4: CH21 noise gate (0: inactive; 1: active)
    - Bit5: CH22 noise gate (0: inactive; 1: active)
    - Bit6: CH23 noise gate (0: inactive; 1: active)
    - Bit7: CH24 noise gate (0: inactive; 1: active)
- Data136: noise gate activity (CH09-CH16)
    - Bit0: CH09 noise gate (0: inactive; 1: active)
    - Bit1: CH10 noise gate (0: inactive; 1: active)

- Bit2: CH11 noise gate (0: inactive; 1: active)
- Bit3: CH12 noise gate (0: inactive; 1: active)
- Bit4: CH13 noise gate (0: inactive; 1: active)
- Bit5: CH14 noise gate (0: inactive; 1: active)
- Bit6: CH15 noise gate (0: inactive; 1: active)
- Bit7: CH16 noise gate (0: inactive; 1: active)

- Data137: noise gate activity (CH01-CH08)
  - Bit0: CH01 noise gate (0: inactive; 1: active)
  - Bit1: CH02 noise gate (0: inactive; 1: active)
  - Bit2: CH03 noise gate (0: inactive; 1: active)
  - Bit3: CH04 noise gate (0: inactive; 1: active)
  - Bit4: CH05 noise gate (0: inactive; 1: active)
  - Bit5: CH06 noise gate (0: inactive; 1: active)
  - Bit6: CH07 noise gate (0: inactive; 1: active)
  - Bit7: CH08 noise gate (0: inactive; 1: active)
- Data558 … Data589: InpCh01-30 compression value (gain reduction) in the range [0,120] (from -60dB (max compression) to 0dB (no compression) step 0.5dB)
- Data590 … Data595: Aux1-6 compression value (gain reduction) in the range [0,120] (from -60dB (max compression) to 0dB (no compression) step 0.5dB)
- Data596 … Data597: MailL-R compression value (gain reduction) in the range [0,120] (from -60dB (max compression) to 0dB (no compression) step 0.5dB)
- Data598 … Data599: USB REC L/R VuMeter

For more details about Data0...Data599 fields please refer to the following table.

Data Field	**Description**
Data0	CH01 VuMeter (before input gain)
Data1	CH01 VuMeter (after input gain and before input processes)
Data2	CH01 VuMeter (after input processes and before channel fader)
Data3	CH01 VuMeter (after channel fader)
Data4	CH02 VuMeter (before input gain)
Data5	CH02 VuMeter (after input gain and before input processes)
Data6	CH02 VuMeter (after input processes and before channel fader)
Data7	CH02 VuMeter (after channel fader)
Data8	CH03 VuMeter (before input gain)
Data9	CH03 VuMeter (after input gain and before input processes)
Data10	CH03 VuMeter (after input processes and before channel fader)
Data11	CH03 VuMeter (after channel fader)
Data12	CH04 VuMeter (before input gain)
Data13	CH04 VuMeter (after input gain and before input processes)
Data14	CH04 VuMeter (after input processes and before channel fader)
Data15	CH04 VuMeter (after channel fader)
Data16	CH05 VuMeter (before input gain)
Data17	CH05 VuMeter (after input gain and before input processes)

Data18	CH05 VuMeter (after input processes and before channel fader)
Data19	CH05 VuMeter (after channel fader)
Data20	CH06 VuMeter (before input gain)
Data21	CH06 VuMeter (after input gain and before input processes)
Data22	CH06 VuMeter (after input processes and before channel fader)
Data23	CH06 VuMeter (after channel fader)
Data24	CH07 VuMeter (before input gain)
Data25	CH07 VuMeter (after input gain and before input processes)
Data26	CH07 VuMeter (after input processes and before channel fader)
Data27	CH07 VuMeter (after channel fader)
Data28	CH08 VuMeter (before input gain)
Data29	CH08 VuMeter (after input gain and before input processes)
Data30	CH08 VuMeter (after input processes and before channel fader)
Data31	CH08 VuMeter (after channel fader)
Data32	CH09 VuMeter (before input gain)
Data33	CH09 VuMeter (after input gain and before input processes)
Data34	CH09 VuMeter (after input processes and before channel fader)
Data35	CH09 VuMeter (after channel fader)
Data36	CH10 VuMeter (before input gain)
Data37	CH10 VuMeter (after input gain and before input processes)
Data38	CH10 VuMeter (after input processes and before channel fader)
Data39	CH10 VuMeter (after channel fader)
Data40	CH11 VuMeter (before input gain)
Data41	CH11 VuMeter (after input gain and before input processes)
Data42	CH11 VuMeter (after input processes and before channel fader)
Data43	CH11 VuMeter (after channel fader)
Data44	CH12 VuMeter (before input gain)
Data45	CH12 VuMeter (after input gain and before input processes)
Data46	CH12 VuMeter (after input processes and before channel fader)
Data47	CH12 VuMeter (after channel fader)
Data48	CH13 VuMeter (before input gain)
Data49	CH13 VuMeter (after input gain and before input processes)
Data50	CH13 VuMeter (after input processes and before channel fader)
Data51	CH13 VuMeter (after channel fader)
Data52	CH14 VuMeter (before input gain)
Data53	CH14 VuMeter (after input gain and before input processes)
Data54	CH14 VuMeter (after input processes and before channel fader)

Data55	CH14 VuMeter (after channel fader)
Data56	CH15 VuMeter (before input gain)
Data57	CH15 VuMeter (after input gain and before input processes)
Data58	CH15 VuMeter (after input processes and before channel fader)
Data59	CH15 VuMeter (after channel fader)
Data60	CH16 VuMeter (before input gain)
Data61	CH16 VuMeter (after input gain and before input processes)
Data62	CH16 VuMeter (after input processes and before channel fader)
Data63	CH16 VuMeter (after channel fader)
Data64	CH17 VuMeter (before input gain)
Data65	CH17 VuMeter (after input gain and before input processes)
Data66	CH17 VuMeter (after input processes and before channel fader)
Data67	CH17 VuMeter (after channel fader)
Data68	CH18 VuMeter (before input gain)
Data69	CH18 VuMeter (after input gain and before input processes)
Data70	CH18 VuMeter (after input processes and before channel fader)
Data71	CH18 VuMeter (after channel fader)
Data72	CH19 VuMeter (before input gain)
Data73	CH19 VuMeter (after input gain and before input processes)
Data74	CH19 VuMeter (after input processes and before channel fader)
Data75	CH19 VuMeter (after channel fader)
Data76	CH20 VuMeter (before input gain)
Data77	CH20 VuMeter (after input gain and before input processes)
Data78	CH20 VuMeter (after input processes and before channel fader)
Data79	CH20 VuMeter (after channel fader)
Data80	CH21 VuMeter (before input gain)
Data81	CH21 VuMeter (after input gain and before input processes)
Data82	CH21 VuMeter (after input processes and before channel fader)
Data83	CH21 VuMeter (after channel fader)
Data84	CH22 VuMeter (before input gain)
Data85	CH22 VuMeter (after input gain and before input processes)
Data86	CH22 VuMeter (after input processes and before channel fader)
Data87	CH22 VuMeter (after channel fader)
Data88	CH23 VuMeter (before input gain)
Data89	CH23 VuMeter (after input gain and before input processes)
Data90	CH23 VuMeter (after input processes and before channel fader)
Data91	CH23 VuMeter (after channel fader)

Data92	CH24 VuMeter (before input gain)
Data93	CH24 VuMeter (after input gain and before input processes)
Data94	CH24 VuMeter (after input processes and before channel fader)
Data95	CH24 VuMeter (after channel fader)
Data96	CH25 VuMeter (before input gain)
Data97	CH25 VuMeter (after input gain and before input processes)
Data98	CH25 VuMeter (after input processes and before channel fader)
Data99	CH25 VuMeter (after channel fader)
Data100	CH26 VuMeter (before input gain)
Data101	CH26 VuMeter (after input gain and before input processes)
Data102	CH26 VuMeter (after input processes and before channel fader)
Data103	CH26 VuMeter (after channel fader)
Data104	CH27 VuMeter (before input gain)
Data105	CH27 VuMeter (after input gain and before input processes)
Data106	CH27 VuMeter (after input processes and before channel fader)
Data107	CH27 VuMeter (after channel fader)
Data108	CH28 VuMeter (before input gain)
Data109	CH28 VuMeter (after input gain and before input processes)
Data110	CH28 VuMeter (after input processes and before channel fader)
Data111	CH28 VuMeter (after channel fader)
Data112	CH29 VuMeter (before input gain)
Data113	CH29 VuMeter (after input gain and before input processes)
Data114	CH29 VuMeter (after input processes and before channel fader)
Data115	CH29 VuMeter (after channel fader)
Data116	CH30 VuMeter (before input gain)
Data117	CH30 VuMeter (after input gain and before input processes)
Data118	CH30 VuMeter (after input processes and before channel fader)
Data119	CH30 VuMeter (after channel fader)
Data120	CH31 VuMeter (before input gain)
Data121	CH31 VuMeter (after input gain and before input processes)
Data122	CH31 VuMeter (after input processes and before channel fader)
Data123	CH31 VuMeter (after channel fader)
Data124	CH32 VuMeter (before input gain)
Data125	CH32 VuMeter (after input gain and before input processes)
Data126	CH32 VuMeter (after input processes and before channel fader)
Data127	CH32 VuMeter (after channel fader)
Data128	MainLeft output VuMeter (before output fader)

Data129	MainLeft output VuMeter (after output fader)
Data130	MainRight output VuMeter (before output fader)
Data131	MainRight output VuMeter (after output fader)
Data132	MonitorLeft output VuMeter (after output fader)
Data133	MonitorRight output VuMeter (after output fader)
Data134	Noise gate activity (CH25-CH32)
Data135	Noise gate activity (CH17-CH24)
Data136	Noise gate activity (CH09-CH16)
Data137	Noise gate activity (CH01-CH08)
Data138	Efx1 Bus VuMeter (before Efx gain)
Data139	Efx2 Bus VuMeter (before Efx gain)
Data140	Efx3 Bus VuMeter (before Efx gain)
Data141	Efx4 Bus VuMeter (before Efx gain)
Data142	Efx5 Bus VuMeter (before Efx gain)
Data143	Efx6 Bus VuMeter (before Efx gain)
Data144	Efx7 Bus VuMeter (before Efx gain)
Data145	Efx8 Bus VuMeter (before Efx gain)
Data146	MainLeft Bus VuMeter (before bus gain)
Data147	MainRight Bus VuMeter (before bus gain)
Data148	MainLeft Bus VuMeter (after bus gain)
Data149	MainRight Bus VuMeter (after bus gain)
Data150	Aux1 output VuMeter (before output fader)
Data151	Aux1 output VuMeter (after output fader)
Data152	Aux2 output VuMeter (before output fader)
Data153	Aux2 output VuMeter (after output fader)
Data154	Aux3 output VuMeter (before output fader)
Data155	Aux3 output VuMeter (after output fader)
Data156	Aux4 output VuMeter (before output fader)
Data157	Aux4 output VuMeter (after output fader)
Data158	Aux5 output VuMeter (before output fader)
Data159	Aux5 output VuMeter (after output fader)
Data160	Aux6 output VuMeter (before output fader)
Data161	Aux6 output VuMeter (after output fader)
Data162	Aux1 Bus VuMeter (before bus gain)
Data163	Aux2 Bus VuMeter (before bus gain)
Data164	Aux3 Bus VuMeter (before bus gain)
Data165	Aux4 Bus VuMeter (before bus gain)

Data166	Aux5 Bus VuMeter (before bus gain)
Data167	Aux6 Bus VuMeter (before bus gain)
Data168	Aux1 Bus VuMeter (after bus gain)
Data169	Aux2 Bus VuMeter (after bus gain)
Data170	Aux3 Bus VuMeter (after bus gain)
Data171	Aux4 Bus VuMeter (after bus gain)
Data172	Aux5 Bus VuMeter (after bus gain)
Data173	Aux6 Bus VuMeter (after bus gain)
Data174	Efx1 Bus VuMeter (after Efx gain)
Data175	Efx2 Bus VuMeter (after Efx gain)
Data176	Efx3 Bus VuMeter (after Efx gain)
Data177	Efx4 Bus VuMeter (after Efx gain)
Data178	Efx5 Bus VuMeter (after Efx gain)
Data179	Efx6 Bus VuMeter (after Efx gain)
Data180	Efx7 Bus VuMeter (after Efx gain)
Data181	Efx8 Bus VuMeter (after Efx gain)
Data182-213	Aux1 – Ch1,⋯,ch32 SEND after Fader
Data214-245	Aux2 – Ch1,⋯,ch32 SEND after Fader
Data246-277	Aux3 – Ch1,⋯,ch32 SEND after Fader
Data278-309	Aux4 – Ch1,⋯,ch32 SEND after Fader
Data310-241	Aux5 – Ch1,⋯,ch32 SEND after Fader
Data342-373	Aux6 – Ch1,⋯,ch32 SEND after Fader
Data374-389	FX1 – Ch1,⋯,ch16 SEND after Fader
Data390-405	FX5 – Ch1,⋯,ch16 SEND after Fader
Data406-421	FX2 – Ch1,⋯,ch16 SEND after Fader
Data422-437	FX6 – Ch1,⋯,ch16 SEND after Fader
Data438-453	FX3 – Ch1,⋯,ch16 SEND after Fader
Data454-469	FX7 – Ch1,⋯,ch16 SEND after Fader
Data470-485	FX4 – Ch1,⋯,ch16 SEND after Fader
Data486-501	FX8 – Ch1,⋯,ch16 SEND after Fader
Data502	Efx1 return on Main bus (after return fader)
Data503	Efx2 return on Main bus (after return fader)
Data504	Efx3 return on Main bus (after return fader)
Data505	Efx4 return on Main bus (after return fader)
Data506	Efx5 return on Main bus (after return fader)
Data507	Efx6 return on Main bus (after return fader)

Data508	Efx7 return on Main bus (after return fader)
Data509	Efx8 return on Main bus (after return fader)
Data510	Efx1 return on Aux1 bus (after return fader)
Data511	Efx2 return on Aux1 bus (after return fader)
Data512	Efx3 return on Aux1 bus (after return fader)
Data513	Efx4 return on Aux1 bus (after return fader)
Data514	Efx5 return on Aux1 bus (after return fader)
Data515	Efx6 return on Aux1 bus (after return fader)
Data516	Efx7 return on Aux1 bus (after return fader)
Data517	Efx8 return on Aux1 bus (after return fader)
Data518	Efx1 return on Aux2 bus (after return fader)
Data519	Efx2 return on Aux2 bus (after return fader)
Data520	Efx3 return on Aux2 bus (after return fader)
Data521	Efx4 return on Aux2 bus (after return fader)
Data522	Efx5 return on Aux2 bus (after return fader)
Data523	Efx6 return on Aux2 bus (after return fader)
Data524	Efx7 return on Aux2 bus (after return fader)
Data525	Efx8 return on Aux2 bus (after return fader)
Data526	Efx1 return on Aux3 bus (after return fader)
Data527	Efx2 return on Aux3 bus (after return fader)
Data528	Efx3 return on Aux3 bus (after return fader)
Data529	Efx4 return on Aux3 bus (after return fader)
Data530	Efx5 return on Aux3 bus (after return fader)
Data531	Efx6 return on Aux3 bus (after return fader)
Data532	Efx7 return on Aux3 bus (after return fader)
Data533	Efx8 return on Aux3 bus (after return fader)
Data534	Efx1 return on Aux4 bus (after return fader)
Data535	Efx2 return on Aux4 bus (after return fader)
Data536	Efx3 return on Aux4 bus (after return fader)
Data537	Efx4 return on Aux4 bus (after return fader)
Data538	Efx5 return on Aux4 bus (after return fader)
Data539	Efx6 return on Aux4 bus (after return fader)
Data540	Efx7 return on Aux4 bus (after return fader)
Data541	Efx8 return on Aux4 bus (after return fader)
Data542	Efx1 return on Aux5 bus (after return fader)
Data543	Efx2 return on Aux5 bus (after return fader)
Data544	Efx3 return on Aux5 bus (after return fader)

Data545	Efx4 return on Aux5 bus (after return fader)
Data546	Efx5 return on Aux5 bus (after return fader)
Data547	Efx6 return on Aux5 bus (after return fader)
Data548	Efx7 return on Aux5 bus (after return fader)
Data549	Efx8 return on Aux5 bus (after return fader)
Data550	Efx1 return on Aux6 bus (after return fader)
Data551	Efx2 return on Aux6 bus (after return fader)
Data552	Efx3 return on Aux6 bus (after return fader)
Data553	Efx4 return on Aux6 bus (after return fader)
Data554	Efx5 return on Aux6 bus (after return fader)
Data555	Efx6 return on Aux6 bus (after return fader)
Data556	Efx7 return on Aux6 bus (after return fader)
Data557	Efx8 return on Aux6 bus (after return fader)
Data558-589	CH01,..,CH32 compression value
Data590	Aux1 compression value
Data591	Aux2 compression value
Data592	Aux3 compression value
Data593	Aux4 compression value
Data594	Aux5 compression value
Data595	Aux6 compression value
Data596	MainL compression value
Data597	MainR compression value
Data598	USB Left recording value
Data599	USB Right recording value

CMD_SET_AUTOMIXER (1CH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1CH	Type	Value	Value1	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: [0,6]
  - 0: Set Automixer enable
  - 1: Set Automixer channel enable
  - 2: Set Automixer type
  - 3: Set Automixer attack time
  - 4: Set Automixer release time
  - 5: Set Automixer threshold time
  - 6: Set Gain Sharing level
- Value:
  - [0,1] if Type = 0 (Please refer to “AutoMixerEnable” param in the MemoryMap doc
  - [0,255] if Type = 1 (Please refer to “AutoMixerChEnable[2]” param in the MemoryMap doc AutoMixerChEnable[0]:
    - Bit0: CH09 enable (0: Disabled; 1: Enabled). It can be enabled only if CH01 source is MIC
    - Bit1: CH10 enable (0: Disabled; 1: Enabled). It can be enabled only if CH02 source is MIC
    - Bit2: CH11 enable (0: Disabled; 1: Enabled). It can be enabled only if CH03 source is MIC
    - Bit3: CH12 enable (0: Disabled; 1: Enabled). It can be enabled only if CH04 source is MIC
    - Bit4: CH13 enable (0: Disabled; 1: Enabled). It can be enabled only if CH05 source is MIC
    - Bit5: CH14 enable (0: Disabled; 1: Enabled). It can be enabled only if CH06 source is MIC
    - Bit6: CH15 enable (0: Disabled; 1: Enabled). It can be enabled only if CH07 source is MIC
    - Bit7: CH16 enable (0: Disabled; 1: Enabled). It can be enabled only if CH08 source is MIC
  - [0,1] if Type = 2 (0: NOM; 1:GAIN SHARING). Please refer to “AutoMixerType” param in the MemoryMap doc
  - [0,2] if Type = 3 (0: Slow; 1:Mid; 2:Fast). Please refer to “AutoMixerAtk” param in the MemoryMap doc
  - [0,2] if Type = 4 (0: Slow; 1:Mid; 2:Fast). Please refer to “AutoMixerRel” param in the MemoryMap doc
  - [0,44] if Type = 5. Threshold in the range [-56,-12] dBFs step 1dB. Please refer to

"AutoMixerThr" param in the MemoryMap doc

- [0,24] if Type = 6. Level in the range [-24,0] dBFs with 1 dB step. Please refer to "GainSharingLevel" param in the MemoryMap doc

- Value1:

- [0,255] if Type = 1 (Please refer to "AutoMixerChEnable[2]" param in the MemoryMap doc
  AutoMixerChEnable[1]:
  - Bit0: CH01 enable (0: Disabled; 1: Enabled). It can be enabled only if CH01 source is MIC
  - Bit1: CH02 enable (0: Disabled; 1: Enabled). It can be enabled only if CH02 source is MIC
  - Bit2: CH03 enable (0: Disabled; 1: Enabled). It can be enabled only if CH03 source is MIC
  - Bit3: CH04 enable (0: Disabled; 1: Enabled). It can be enabled only if CH04 source is MIC
  - Bit4: CH05 enable (0: Disabled; 1: Enabled). It can be enabled only if CH05 source is MIC
  - Bit5: CH06 enable (0: Disabled; 1: Enabled). It can be enabled only if CH06 source is MIC
  - Bit6: CH07 enable (0: Disabled; 1: Enabled). It can be enabled only if CH07 source is MIC
  - Bit7: CH08 enable (0: Disabled; 1: Enabled). It can be enabled only if CH08 source is MIC
- [0] if Type < > 1

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1CH	Type	Value	Value1	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_DUCKER (1FH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1FH	Type	SrcCh	Att	Rel	Att	Hold

D6	D7	ETX
Thr	Enable	F7H

where:
- Type: [0,1]
  - 0: Set ducker params
  - 1: Set channel attenuation
- Rel: Ducker release time = [0,28] (refer to “DuckerRel” param in the MemoryMap doc)
- Atk: Ducker attack time = [0,28] (refer to “DuckerAtk” param in the MemoryMap doc)
- Hold: Ducker hold time = [0,99] (refer to “DuckerHold” param in the MemoryMap doc)
- Hold: Ducker threshold = [0,44] (refer to “DuckerThr” param in the MemoryMap doc)
- Enable: Ducker enable (0: Disabled; 1: Enabled) (refer to “DuckerEnable” param in the MemoryMap doc)
- SrcCh: select a channel for which the attenuation has to be set = [0,29]
- Att: channel attenuation = [0,80] (refer to “DuckerAtt” param in the MemoryMap doc)

“SrcCh” and “Att” parameters are not used if Type = 0
“Rel”, “Atk”, “Hold”, “Thr” and “Enable” are not used if Type = 1

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	1FH	Type	SrcCh	Att	Rel	Att	Hold

D6	D7	ETX
Thr	Enable	F7H

CMD_SET_DUCKER_PRIORITY (20H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	20H	ChMsb	Ch24	Ch16	ChLsb	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- ChMsb: channel priority enable (high byte)
  - Bit 0 --> CH25 Priority (0:Disabled; 1:Enabled)
  - Bit 1 --> CH26 Priority (0:Disabled; 1:Enabled)
  - Bit 2 --> CH27 Priority (0:Disabled; 1:Enabled)
  - Bit 3 --> CH28 Priority (0:Disabled; 1:Enabled)
  - Bit 4 --> 0
  - Bit 5 --> 0
  - Bit 6 --> 0
  - Bit 7 --> 0
- Ch24: channel priority enable
  - Bit 0 --> CH17 Priority (0:Disabled; 1:Enabled)
  - Bit 1 --> CH18 Priority (0:Disabled; 1:Enabled)
  - Bit 2 --> CH19 Priority (0:Disabled; 1:Enabled)
  - Bit 3 --> CH20 Priority (0:Disabled; 1:Enabled)
  - Bit 4 --> CH21 Priority (0:Disabled; 1:Enabled)
  - Bit 5 --> CH22 Priority (0:Disabled; 1:Enabled)
  - Bit 6 --> CH23 Priority (0:Disabled; 1:Enabled)
  - Bit 7 --> CH24 Priority (0:Disabled; 1:Enabled)
- Ch16: channel priority enable
  - Bit 0 --> CH09 Priority (0:Disabled; 1:Enabled)
  - Bit 1 --> CH10 Priority (0:Disabled; 1:Enabled)
  - Bit 2 --> CH11 Priority (0:Disabled; 1:Enabled)
  - Bit 3 --> CH12 Priority (0:Disabled; 1:Enabled)
  - Bit 4 --> CH13 Priority (0:Disabled; 1:Enabled)
  - Bit 5 --> CH14 Priority (0:Disabled; 1:Enabled)
  - Bit 6 --> CH15 Priority (0:Disabled; 1:Enabled)
  - Bit 7 --> CH16 Priority (0:Disabled; 1:Enabled)
- ChLsb: channel priority enable (low byte)
  - Bit 0 --> CH01 Priority (0:Disabled; 1:Enabled)
  - Bit 1 --> CH02 Priority (0:Disabled; 1:Enabled)
  - Bit 2 --> CH03 Priority (0:Disabled; 1:Enabled)
  - Bit 3 --> CH04 Priority (0:Disabled; 1:Enabled)
  - Bit 4 --> CH05 Priority (0:Disabled; 1:Enabled)
  - Bit 5 --> CH06 Priority (0:Disabled; 1:Enabled)

- Bit 6 --> CH07 Priority (0:Disabled; 1:Enabled)
- Bit 7 --> CH08 Priority (0:Disabled; 1:Enabled)

Please refer to “DuckerPriorityChSel” param in the MemoryMap doc.

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	20H	ChMsb	Ch24	Ch16	ChLsb	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_RTA (21H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	21H	Type	Enable	Ch	WLen	NrAvg	MicLev

D6	D7	ETX
PinkLev	00H	F7H

where:
- Type: [0,1]
  - 0: Normal
  - 1: use the external Mic
- Enable: 0: RTA disabled; 1: RTA enabled
- Ch: RTA channel
  - 0: Monitor Left
  - 1: Monitor Right
  - 2: Monitor Left+Right
  - 3: Main Left
  - 4: Main Right
  - 5: Main Left+Right
  - 6: Aux1
  - 7: Aux2
  - 8: Aux3
  - 9: Aux4
  - 10: Aux5
  - 11: Aux6
- WLen: RTA analisys window length
  - 0: 2048 samples
  - 1: 4096 samples
  - 2: 8192 samples
  - 3: 16384 samples
  - 4: 32768 samples
- NrAvg: Fixed to 8
- MicLev: only for type=1 [0,..,72] =from -18dB to +18dB step 0.5dB
- PinkLev:only for type=1 [0,..,30] =from -30dBfs to 0dBfs step 1dB

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	21H	Type	Enable	Ch	WLen	NrAvg	MicLev

D6	D7	ETX

PinkLev	00H	F7H

CMD_GET_RTA_METER (22H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	22H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (36 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	22H	Data0	Data1	Data2	Data3	Data4	Data5

...	D30	ETX
...	Data30	F7H

where:
- Data0 … Data30: signal RMS value in the frequency Band0 … Band30, in the range [0,69] (from -69dB to 0dB step 1dB)

Band #	**Freq. Range [Hz]**
0	17,8 – 22,4
1	22,4 – 28,3
2	28,3 – 35,6
3	35,6 – 44,9
4	44,9 – 56,6
5	56,6 – 71,3
6	71,3 – 89,8
7	89,8 – 113,1
8	113,1 – 142,5
9	142,5 – 179,6
10	179,6 – 226,3
11	226,3 – 285,1
12	285,1 – 359,2
13	359,2 – 452,5

14	452,5 – 570,2
15	570,2 – 718,4
16	718,4 – 905,1
17	905,1 – 1140,4
18	1140,4 – 1436,8
19	1436,8 – 1810,2
20	1810,2 – 2280,7
21	2280,7 – 2873,5
22	2873,5 – 3620,4
23	3620,4 – 4561,4
24	4561,4 – 5747,0
25	2747,0 – 7240,8
26	7240,8 – 9122,8
27	9122,8 – 11494,0
28	11494,0 – 14481,5
29	14481,5 – 18245,6
30	18245,6 – 22988,0

CMD_SET_INP_PEQ_TYPE (23H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	23H	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: PEQ type (0: Normal; 1: Mid-Morph)
- SrcCh: input channel [0,29]

Please refer to “InpPeqType” param in the MemoryMap doc

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	23H	Type	SrcCh	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_HP_LP (24H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	24H	Type	SrcCh	HpLp	FType	FFreqH	FFreqL

D6	D7	ETX
00H	00H	F7H

where:
- Type: (0: Set Efx1-3/5-7 HP/LP; 1: Set MainLR HP/LP; 2: Set Aux1-6 HP/LP)
- SrcCh: source channel
  - [0,1,2, 4,5,6] if Type = 0
  - NotUsed if Type = 1
  - [0,5] if Type = 2
- HpLp: select the filter (0: highpass; 1:lowpass). If Type = 0 and SrcCh = 0,4, HpLp should be 0 (there is no lowpass filter for Efx1)
- FType: filter type
  - [0,2] (0: Bypass; 1: -6dB/Oct; 2: -12dB/Oct) if Type = 0
  - [0,6] (0: Bypass; 1: -6dB/Oct Butter; 2: -12dB/Oct Butter; 3: -12dB/Oct LinkwitzM; 4:-18dB/Oct Butter; 5: -24dB/Oct Butter; 6: -24dB Linkwitz) if Type = [1,2]
- FFreqH: filter frequency (high byte)
- FFreqL: filter frequency (low byte)

Filter frequency [Hz] = FFreqH*256 + FFreqL

Please refer to the following params in the MemoryMap doc:

- “Efx1HpFilterType[2]”, “Efx1HpFilterFreqH[2]”, “Efx1HpFilterFreqL[2]” for Type = 0 and SrcCh = 0,4

- “Efx2HpFilterType[2]”, “Efx2HpFilterFreqH[2]”, “Efx2HpFilterFreqL[2]”, “Efx2LpFilterType[2]”, “Efx2LpFilterFreqH[2]”, “Efx2LpFilterFreqL[2]” for Type = 0 and SrcCh = 1,5

- “Efx3HpFilterType[2]”, “Efx3HpFilterFreqH[2]”, “Efx3HpFilterFreqL[2]”, “Efx3LpFilterType[2]”, “Efx3LpFilterFreqH[2]”, “Efx3LpFilterFreqL[2]” for Type = 0 and SrcCh = 2,6

- “MasterHpFilterType”, “MasterHpFilterFreqH”, “MasterHpFilterFreqL”, “MasterLpFilterType”, “MasterLpFilterFreqH”, “MasterLpFilterFreqL” for Type = 1

- “AuxHpFilterType[6]”, “AuxHpFilterFreqH[6]”, “AuxHpFilterFreqL[6]”, “AuxLpFilterType[6]”, “AuxLpFilterFreqH[6]”, “AuxLpFilterFreqL[6]” for Type = 2

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	24H	Type	SrcCh	HpLp	FType	FFreqH	FFreqL

D6	D7	ETX
00H	00H	F7H

CMD_SET_EFX_TYPE (25H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	25H	00H	SrcCh	Type	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- SrcCh: [0,1] 0: Efx3, 1:Efx7
- Type: EFX3 type [0,6] (0: Chorus; 1: Flanger; 2: Phaser; 3: Tremolo; 4: Chorus+Tremolo; 5: Flanger+Tremolo; 6: Phaser+Tremolo)

Please refer to “Efx3Type” param in the MemoryMap doc.

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	25H	00H	SrcCh	Type	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_EFX_REVERB (26H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	26H	PreDelayL	DecayL	Density	PreDelayF	DecayH	Fx

D6	D7	ETX
00H	00H	F7H

where:
- PreDelayL: Reverb pre-delay low byte (refer to “Efx1RevPreDelayL[2]” param in the MemoryMap doc.
- DecayL: Reverb decay low byte (refer to “Efx1RevDecayL[2]” param in the MemoryMap doc.
- Density: Reverb density [0,100] (refer to “Efx1RevDensity[2]” param in the MemoryMap doc.
- PreDelayF: Reverb fine pre-delay(refer to “Efx1RevPreDelayFine[2]” param in the MemoryMap doc.
- DecayH: Reverb decay high byte (refer to “Efx1RevDecayH[2]” param in the MemoryMap doc.
- Fx: [0,1] 0: Efx1, 1:Efx4

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	26H	PreDelayL	DecayL	Density	PreDelayF	DecayH	Fx

D6	D7	ETX
00H	00H	F7H

CMD_SET_EFX_DELAY (27H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	27H	DelayH	DelayL	Fbk	DelayF	Fx	00H

D6	D7	ETX
00H	00H	F7H

where:
- DelayH: delay high byte (refer to “Efx2DelayH[2]” param in the MemoryMap doc)
- DelayL: delay low byte (refer to “Efx2DelayL[2]” param in the MemoryMap doc)
- Fbk: [0,100] (refer to “Efx2DelayFeedback[2]” param in the MemoryMap doc)
- DelayF: delay fine (refer to “Efx2DelayFine[2]” param in the MemoryMap doc)
- Fx: [0,1] 0: Efx2, 1:Efx5

Delay [ms] = DelayH*256 + DelayL + 0.05*DelayF

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	27H	DelayH	DelayL	Fbk	DelayF	Fx	00H

D6	D7	ETX
00H	00H	F7H

Beats Per Minute (BPM) to Delay [ms]

The delay value can be derived from the BPM using the following formula:

*Delay[ms] = (60000 / BPM) * K*

where K depends on the “Note” value as shown in the following table:

NOTE	K
1/4D	3/2 * 1
1/4	1
1/4T	2/3 * 1
1/8D	3/2 * 1/2

1/8	1/2
1/8T	2/3 * ½
1/16D	3/2 * ¼
1/16	¼
1/16T	2/3 * ¼
1/32	1/8

The BPM range must be limited to the following values:

- 89 <= BPM <= 180 if “Note” is equal to 1/4
- 133 <= BPM <= 180 if “Note” is equal to 1/4D
- 60 <= BPM <= 180 if “Note” is not equal to 1/4 or to 1/4D

CMD_SET_EFX_MODULATION (28H)

Command (16 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	27H	ChFlModType	ChFlModFreqL	FlFbk	PhModType

D4	D5	D6	D7	D8	D9	D10
PhModFreqL	PhFbk	TrModAmp	TrModFreqL	ChFlModFreqH	PhModFreqH	TrModFreqH

D11	EFX
Fx	F7H

where:
- ChFlModType: Efx3,7 Chorus/Flanger modulation type (refer to “Efx3ChFlModType” param in the MemoryMap doc)
    - 0: Sine
    - 1: Lookup table
- ChFlModFreqL: Efx3,7 Chorus/Flanger modulation freq low byte (refer to “Efx3ChFlModFreqL” param in the MemoryMap doc)
- FlFbk: Efx3 Flanger feedback [0,100] (refer to “Efx3FlFeedback” param in the MemoryMap doc)
- PhModType: Efx3 Phaser modulation type (refer to “Efx3PhModType” param in the MemoryMap doc)
    - 0: Sine
    - 1: Lookup table
- PhModFreqL: Efx3,7 Phaser modulation freq low byte (refer to “Efx3PhModFreqL” param in the MemoryMap doc)
- PhFbk: Efx3,7 Phaser feedback [0,100] (refer to “Efx3PhFeedback” param in the MemoryMap doc)
- TrModAmp: Efx3,7 Tremolo modulation amplitude [0,100] (refer to “Efx3TrModAmp” param in the MemoryMap doc)
- TrModFreqL: Efx3,7 Tremolo modulation freq low byte (refer to “Efx3TrModFreqL” param in the MemoryMap doc)
- ChFlModFreqH: Efx3,7 Chorus/Flanger modulation freq high byte (refer to “Efx3ChFlModFreqH” param in the MemoryMap doc)
- PhModFreqH: Efx3,7 Phaser modulation freq high byte (refer to “Efx3PhModFreqH” param in the MemoryMap doc)
- TrModFreqH: Efx3,7 Tremolo modulation freq high byte (refer to “Efx3TrModFreqH” param in the MemoryMap doc)

ChFlModFreq [Hz] : Chorus/Flanger frequency (rate) = (ChFlModFreqH * 256 + ChFlModFreqL)*0.01 + 0.01

PhModFreq [Hz] : Phaser frequency (rate) = (PhModFreqH * 256 + PhModFreqL)*0.01 + 0.01

TrModFreq [Hz] : Tremolo frequency (rate) = (TrModFreqH * 256 + TrModFreqL)*0.01 + 0.01

- Fx: [0,1] 0: Efx3, 1:Efx7

The maximum value of ChFlModFreq, PhModFreq and TrModFreq is 8Hz

Response (16 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	27H	ChFlModType	ChFlModFreqL	FlFbk	PhModType

D4	D5	D6	D7	D8	D9	D10
PhModFreqL	PhFbk	TrModAmp	TrModFreqL	ChFlModFreqH	PhModFreqH	TrModFreqH

D11	EFX
Fx	F7H

Beats Per Minute (BPM) to Frequency (Rate) [Hz]

The modulation frequency (rate) value can be derived from the BPM using the following formula:

*Freq[Hz] = BPM / (60 * K)*

where K depends on the “Note” value as shown in the following table:

NOTE	K
4	16
3	12
2	8
1	4
1/2D	3/2 * 2
1/2	2
1/2T	2/3 * 2
1/4D	3/2 * 1
1/4	1
1/4T	2/3 * 1
1/4D	3/2 * 1/2
1/4	1/2
1/4T	2/3 * 1/2
1/16	1/4

The BPM range must be limited to the following values:

- 15 <= BPM <= 120 if “Note” is equal to 1/16
- 15 <= BPM <= 160 if “Note” is equal to 1/8T
- 15 <= BPM <= 180 if “Note” is not equal to 1/16 or to 1/8T

CMD_SET_EFX_ROOM (29H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	29H	Type	DecayL	DecayH	Fx	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: Efx4.8 Room Type = [0,2] (refer to “Efx4RoomType[2]” param in the MemoryMap doc)
    - 0: Small Room
    - 1: Mid Room
    - 2: Large Room
- DecayL: Efx decay low byte (refer to “Efx4RoomDecayL[2]” param in the MemoryMap doc)
- DecayH: Efx decay high byte (refer to “Efx4RoomDecayH[2]” param in the MemoryMap doc)
- Fx: [0,1] 0: Efx4, 1:Efx8

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	29H	Type	DecayL	DecayH	Fx	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_FBK_ELIM (2AH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	2AH	Type	SrcCh	InpSel	Bypass	FreqDisp	00H

D6	D7	ETX
00H	00H	F7H

where:
- Type: select the Feedback Eliminator to set
  - 0: Set input feedback eliminator
  - 1: Set MainLR output feedback eliminator
  - 2: Set Aux1-6 output feedback eliminator
- SrcCh: source channel
  - [0,1] if Type = 0 (2 input feedback eliminators are available. They are shared between channels CH1-16
  - NotUsed if Type = 1
  - [0,5] if Type = 2
- InpSel:
  - [0,16] if Type = 0. It selects an input channel to be assigned to an input feedback eliminator (=0 No channel selected; 1: CH1 selected; … ; 16: CH16 selected)
  - NotUsed if Type = [1,2]
- Bypass: feedback eliminator bypass (0:Active; 1:Bypassed)
- FreqDisp; feedback eliminator frequency displacement = [0,12]

Please refer to the following params in the MemoryMap doc:

- “InpFbkElimBypass”, “InpFbkElimFreqDisp[2]” and “InpFbkElimSrcSel[2]” for Type = 0

- “MasterFbkElimBypass” and “MasterFbkElimFreqDisp” for Type = 1

- “AuxFbkElimBypass” and “AuxFbkElimFreqDisp[6]” for Type = 2

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	2AH	Type	SrcCh	InpSel	Bypass	FreqDisp	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_MON_SRC (2CH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	2CH	00H	00H	Src	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Src: monitor output source
  - 0: Main LR output
  - 1: Aux1 output
  - 2: Aux2 output
  - 3: Aux3 output
  - 4: Aux4 output
  - 5: Aux5 output
  - 6: Aux6 output
  - 7: Main Bus
  - 8: Aux1 Bus
  - 9: Aux2 Bus
  - 10: Aux3 Bus
  - 11: Aux4 Bus
  - 12: Aux5 output
  - 13: Aux6 output

Please refer to “MonSrc” param in the MemoryMap doc.

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	2CH	00H	00H	Src	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_BROWSE_USB (30H)

This command has a different format and meaning depending on the value assigned to D0 field.

Get Current Dir Name (D0 = 0)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	30H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (8 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	30H	Error	00H	NameLen	NameChar1

D4	...	...	DN	ETX
NameChar2	...	...	NameCharM	F7H

where
- Error:
  - 00H: No Error
  - FEH: Error in opening USB media (USB memory not available). If Error is equals to FEH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached. If Error is equals to FFH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
- NameLen: the lenght of the current dir absolute path name
- NameChar1..M: the current dir absolute path name (NameLen 8-bit ascii chars).

Get Current Dir Next Entry (D0 = 1)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	30H	01H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (20 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	30H	Error	EntryType	NameLen	NameChar1

D4	...	...	DN	DN+1	DN+2	DN+3	DN+4	DN+5
NameChar2	...	...	NameChar M	FS1	FS2	FS3	FS4	DT1

DN+6	DN+7	DN+8	DN+9	DN+10	DN+11	DN+12	ETX
DT2	DT3	DT4	FD1	FD2	FD3	FD4	F7H

where
- Error:
  - 01H: No Error
  - FEH: End-Of-Dir (No more entry in the current dir). If Error is equals to FEH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached. If Error is equals to FFH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
- EntryType:
  - 0: the entry is a sub dir. FSX and FDX fields are equal to 00H. DTX fields are the sub dir creation date
  - 1: the entry is a .wav file. FSX fields are the file size. FDX is the file duration and DTX fields are the file creation date
  - 2: the entry is a .lst file (playlist). FSX and FDX fields are equal to 00H. DTX fields are the playlist creation date
  - 3: the entry is a file, but it cannot be played (it should not appear in the dir contents list). In this case the response is 13 bytes long D1..D7 fields is equals to 00H.
- NameLen: the lenght of the dir entry
- NameChar1..M: the entry name (NameLen 8-bit ascii chars)
- FSX: File size bytes
  - File Size [bytes] = (FS4 << 24) + (FS3 << 16) + (FS2 << 8) + FS1
- DTX: DATE bytes
  - DATE (uint32) = (DT4<< 24) + (DT3 << 16) + (DT2 << 8) + DT1
- FDX: File duration bytes
  - File Duration [seconds] = (FD4<< 24) + (FD3 << 16) + (FD2 << 8) + FD1

DATE FORMAT:
DATE bits 4:0 --> Second / 2 (0..29)
DATE bits 10:5 --> Minute (0..59)
DATE bits 15:11 --> Hour (0..23)

DATE bits 20:16 --> Day (1..31)
DATE bits 24:21 --> Month (1..12)
DATE bits 31:25 --> Year origin from 1980 (0..127)

Enter Sub Dir (D0 = 2)

Command (8+NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	30H	02H	00H	NameLen	NameChar1

D4	...	...	DN	ETX
NameChar2	...	...	NameCharM	F7H

where:
- NameLen: the length of the sub dir name to enter
- NameChar1..M: the sub dir name to enter (NameLen 8-bit ascii chars)

Response (8 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	30H	Error	00H	NameLen	NameChar1

D4	...	...	DN	ETX
NameChar2	...	...	NameCharM	F7H

where
- Error:
  - 02H: No Error
  - FEH: Error (Can't enter dir or USB memory not available). If Error is equals to FEH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached. If Error is equals to FFH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
- NameLen: the lenght of the current dir name
- NameChar1..M: the current dir name (NameLen 8-bit ascii chars). The returned name is the absolute path name including the sub dir name

Leave Dir (D0 = 3)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5

F0H	18H	xx	30H	03H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (8 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	30H	Error	00H	NameLen	NameChar1

D4	...	...	DN	ETX
NameChar2	...	...	NameCharM	F7H

where
- Error:
  - 03H: No Error
  - FEH: Error in opening USB media (USB memory not available). If Error is equals to FEH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached. If Error is equals to FFH, the command response is 13 bytes long and D1..D7 fields is equals to 00H
- NameLen: the lenght of the current dir absolute path name
- NameChar1..M: the current dir absolute path name (NameLen 8-bit ascii chars).

Browse Completed (D0 = E0H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	30H	E0H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	30H	E0H	00H	00H	00H	00H	00H

D6	D7	ETX

00H	00H	F7H

GET CURRENT DIR CONTENTS PROCEDURE

1) Send the CMD_BROWSE_USB command with D0 = 0 to get the current dir absolute path name
2) Send the CMD_BROWSE_USB command with D0 = 1 untill End-Of-Dir is returned. to get the dir contents (i.e. the file list name)
3) Send the CMD_BROWSE_USB command with D0 = E0H to signal the browse procedure completion.

ENTER SUB DIR AND GET DIR CONTENTS PROCEDURE

1) Send the CMD_BROWSE_USB command with D0 = 2 to enter sub dir
2) Send the CMD_BROWSE_USB command with D0 = 1 untill End-Of-Dir is returned. to get the sub dir contents (i.e. the file list name)
3) Send the CMD_BROWSE_USB command with D0 = E0H to signal the browse procedure completion.

LEAVE CURRENT DIR AND GET UPLEVEL DIR CONTENTS PROCEDURE

1) Send the CMD_BROWSE_USB command with D0 = 3 to leave current dir
2) Send the CMD_BROWSE_USB command with D0 = 1 untill End-Of-Dir is returned. to get the uplevel dir contents (i.e. the file list name)
3) Send the CMD_BROWSE_USB command with D0 = E0H to signal the browse procedure completion.

CMD_USB_PLAY_REC (31H)

This command has a different format and meaning depending on the value assigned to D0 field.

Start Playback (D0 = 0)

Command (7+NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	31H	00H	NameLen	NameChar1	NameChar2

D4	...	...	DN	ETX
NameChar3	...	...	NameCharM	F7H

where:
- NameLen: the length of the file name to playback
- NameChar1..M: the file name to playback (NameLen 8-bit ascii chars)

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	31H	Error	NameLen	NameChar1	NameChar2

D4	...	...	D7	ETX
NameChar3	...	...	NameChar7	F7H

where
- Error:
  - 00H: No Error
  - FEH: Error in opening the requested file
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Start Recording (D0 = 1)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	01H	DT1	DT2	DT3	DT4	00H

D6	D7	ETX
00H	00H	F7H

where:
- DTX: DATE bytes

DATE (uint32) = (DT4<< 24) + (DT3 << 16) + (DT2 << 8) + DT1

DATE FORMAT:
DATE bits 4:0 --> Second / 2 (0..29)
DATE bits 10:5 --> Minute (0..59)
DATE bits 15:11 --> Hour (0..23)
DATE bits 20:16 --> Day (1..31)
DATE bits 24:21 --> Month (1..12)
DATE bits 31:25 --> Year origin from 1980 (0..127)

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	DT1	DT2	DT3	DT4	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 01H: No Error
  - FEH: Can't start recording or recording already active
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Pause ON/OFF (D0 = 2)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	02H	Pause	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:

- Pause = 0 Disable Pause; 1: Enable Pause

Note: this command should not be sent if the playback is not active

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 02H: No Error
  - FEH: Can't pause playback
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Forward ON/OFF (D0 = 3)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	03H	Forward	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Forward = 0 Disable Forward; 1: Enable Forward

Note: this command should not be sent if the playback is not active

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 03H: No Error
  - FEH: Can't forward playback
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Rewind ON/OFF (D0 = 4)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	04H	Rewind	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Rewind = 0 Disable Rewind; 1: Enable Rewind

Note: this command should not be sent if the playback is not active

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 04H: No Error
  - FEH: Can't rewind playback
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Stop Playback (D0 = 5)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	05H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Note: this command should not be sent if the playback is not active

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 05H: No Error
  - FEH: Can't stop playback
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.
  -

Stop Recording (D0 = 6)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	06H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Note: this command should not be sent if the recording is not active

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	Error	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where
- Error:
  - 06H: No Error
  - FEH: Can't stop recording
  - FFH: System busy. The remote controller should try resending the command every 100ms untill the system accepts it (Error not equals to FFH) or a max number of retries (typically 100) is reached.

Enable Playback Loop (D0 = 7)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	07H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	07H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Disable Playback Loop (D0 = 8)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	08H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	31H	08H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_GET_USB_STATUS (32H)

This command has a different format and meaning depending on the value assigned to D0 field.

Get Status Info (D0 = 0)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	32H	00H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (36 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	32H	Flags	PET1	PET2	PET3	PET4	PTT1

D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
PTT2	PTT3	PTT4	RET1	RET2	RET3	RET4	RRT1	RRT2	RRT3

D16	D17	D18	D19	D20	D21	D22	D23	D24	D25
RRT4	SrcType	InChE1	InChE2	InChE3	InChE4	InChE1	InChE2	InChE3	InChE4

D26	ETX	D26	ETX	D26	ETX
PGE1	PGE2	OE1	OE2	Span	F7H

where:
- Flags: USB status bitfield flags
  - bit0: recording status (0: rec inactive; 1: rec active)
  - bit1: playback status (0: playback inactive; 1: playback active)
  - bit2: playback loop status (0: playback loop inactive; 1: playback loop active)
  - bit4-3: pause/forward/rewind status (0: pause/forward/rewind inactive; 1: pause active; 2: forward active; 3: rewind active)
  - bit5: recording file changed flag (0: rec file not changed; 1: rec file changed). When this flag is 1, the remote controller should read the recording file name by sending the command CMD_GET_USB_STATUS with D0 field equals to 2.
  - bit6: playback file changed flag (0: playback file not changed; 1: playback file changed). When this flag is 1, the remote controller should read the playback file name by sending the

command CMD_GET_USB_STATUS with D0 field equals to 1.

- bit7: current dir changed flag (0: current dir not changed; 1: current dir changed). When this flag is 1, the remote controller should read the current dir name and the dir contents (the list of files name) by sending the command CMD_BROWSE_USB (please refer to the "*GET CURRENT DIR CONTENTS PROCEDURE*" in the CMD_BROWSE_USB section)

- PETX: Playback elapsed time bytes
  - Elapsed Time (seconds) = (PET4 << 24) + (PET3 << 16) + (PET2 << 8) + PET1
- PTTX: Playback total time bytes
  - Total Time (seconds) = (PTT4 << 24) + (PTT3 << 16) + (PTT2 << 8) + PTT1
- RETX: Recording elapsed time bytes
  - Elapsed Time (seconds) = (RET4 << 24) + (RET3 << 16) + (RET2 << 8) + RET1
- RRTX: Recording remaining time bytes

Remaining Time (seconds) = (RRT4 << 24) + (RRT3 << 16) + (RRT2 << 8) + RRT1

- SrcType: recording source type (please refer to the "UsbRecSrcType" param in the memory map doc)
- InChE1: recording input channel enable (please refer to the "UsbRecInpChEnable[0][0]" param in the memory map doc)
- InChE2: recording input channel enable (please refer to the "UsbRecInpChEnable[0][1]" param in the memory map doc)
- InChE3: recording input channel enable (please refer to the "UsbRecInpChEnable[0][2]" param in the memory map doc)
- InChE4: recording input channel enable (please refer to the "UsbRecInpChEnable[0][3]" param in the memory map doc)
- InChE5: recording input channel enable (please refer to the "UsbRecInpChEnable[1][0]" param in the memory map doc)
- InChE6: recording input channel enable (please refer to the "UsbRecInpChEnable[1][1]" param in the memory map doc)
- InChE7: recording input channel enable (please refer to the "UsbRecInpChEnable[1][2]" param in the memory map doc)
- InChE8: recording input channel enable (please refer to the "UsbRecInpChEnable[1][3]" param in the memory map doc)
- PGE1: recording PRE-GEQ output channel enable (please refer to the "UsbRecPreGeqEnable[0]" param in the memory map doc)
- PGE2: recording PRE-GEQ output channel enable (please refer to the "UsbRecPreGeqEnable[1]" param in the memory map doc)
- OE1: recording output channel enable (please refer to the "UsbRecOutEnable[0]" param in the memory map doc)
- OE2: recording output channel enable (please refer to the "UsbRecOutEnable[1]" param in the memory map doc)
- Span: recording file span time (please refer to the "UsbRecFileSpan" param in the memory map doc)

<u>Get Actual Playback File Name (D0 = 1)</u>

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	32H	01H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (6 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	32H	NameLen	NameChar1	NameChar2	NameChar3

D4	...	...	DN	ETX
NameChar4	...	...	NameCharM	F7H

where
- NameLen: the length of the actual playback file name
- NameChar1..M: the actual playback file name (NameLen 8-bit ascii chars)

<u>Get Actual Recording File Name (D0 = 2)</u>

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	32H	02H	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Response (6 + NameLen Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	32H	NameLen	NameChar1	NameChar2	NameChar3

D4	...	...	DN	ETX
NameChar4	...	...	NameCharM	F7H

where
- NameLen: the length of the actual recording file name
- NameChar1..M: the actual recording file name (NameLen 8-bit ascii chars)

CMD_SET_USB_REC_SRC (33H)

Command (14 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	33H	SrcType	EnableLeftH	EnableLeft24	EnableLeft16

D4	D5	D6	D7	D8	ETX
EnableLeftL	EnableRightH	EnableRight24	EnableRight16	EnableRightL	F7H

where:
- SrcType: USB recording source type
  - = 0: input channels recording
  - = 1: output channels recording
  - = 2: output PRE-GEQ channels recording

- EnableLeftH: USB recording left channel source (MSB)
  - if SrcType = 0: input channels recording enable
    - - Bit 0 --> CH25 Enable (0:Disabled; 1:Enabled)
    - - Bit 1 --> CH26 Enable (0:Disabled; 1:Enabled)
    - - Bit 2 --> CH27 Enable (0:Disabled; 1:Enabled)
    - - Bit 3 --> CH28 Enable (0:Disabled; 1:Enabled)
    - - Bit 4 --> CH29 Enable (0:Disabled; 1:Enabled)
    - - Bit 5 --> CH30 Enable (0:Disabled; 1:Enabled)
    - - Bit 6 --> 0
    - - Bit 7 --> 0
  - if SrcType > 0: unused

- EnableLeft24: USB recording left channel source
  - if SrcType = 0: input channels recording enable
    - - Bit 0 --> CH17 Enable (0:Disabled; 1:Enabled)
    - - Bit 1 --> CH18 Enable (0:Disabled; 1:Enabled)
    - - Bit 2 --> CH19 Enable (0:Disabled; 1:Enabled)
    - - Bit 3 --> CH20 Enable (0:Disabled; 1:Enabled)
    - - Bit 4 --> CH21 Enable (0:Disabled; 1:Enabled)
    - - Bit 5 --> CH22 Enable (0:Disabled; 1:Enabled)
    - - Bit 6 --> CH23 Enable (0:Disabled; 1:Enabled)
    - - Bit 7 --> CH24 Enable (0:Disabled; 1:Enabled)
  - if SrcType > 0: unused

- EnableLeft16: USB recording left channel source
  - if SrcType = 0: input channels recording enable
    - - Bit 0 --> CH09 Enable (0:Disabled; 1:Enabled)
    - - Bit 1 --> CH10 Enable (0:Disabled; 1:Enabled)
    - - Bit 2 --> CH11 Enable (0:Disabled; 1:Enabled)

    - Bit 3 --> CH12 Enable (0:Disabled; 1:Enabled)
    - Bit 4 --> CH13 Enable (0:Disabled; 1:Enabled)
    - Bit 5 --> CH14 Enable (0:Disabled; 1:Enabled)
    - Bit 6 --> CH15 Enable (0:Disabled; 1:Enabled)
    - Bit 7 --> CH16 Enable (0:Disabled; 1:Enabled)
- if SrcType > 0: unused

- EnableLeftL: USB recording left channel source (LSB)
- if SrcType = 0: input channels recording enable
    - Bit 0 --> CH01 Enable (0:Disabled; 1:Enabled)
    - Bit 1 --> CH02 Enable (0:Disabled; 1:Enabled)
    - Bit 2 --> CH03 Enable (0:Disabled; 1:Enabled)
    - Bit 3 --> CH04 Enable (0:Disabled; 1:Enabled)
    - Bit 4 --> CH05 Enable (0:Disabled; 1:Enabled)
    - Bit 5 --> CH06 Enable (0:Disabled; 1:Enabled)
    - Bit 6 --> CH07 Enable (0:Disabled; 1:Enabled)
    - Bit 7 --> CH08 Enable (0:Disabled; 1:Enabled)

- if SrcType = 1: ouput channels recording enable
    - Bit 0 --> Main output Enable (0:Disabled; 1:Enabled)
    - Bit 1 --> Aux1 output Enable (0:Disabled; 1:Enabled)
    - Bit 2 --> Aux2 output Enable (0:Disabled; 1:Enabled)
    - Bit 3 --> Aux3 output Enable (0:Disabled; 1:Enabled)
    - Bit 4 --> Aux4 output Enable (0:Disabled; 1:Enabled)
    - Bit 5 --> Aux5 output Enable (0:Disabled; 1:Enabled)
    - Bit 6 --> Aux6 output Enable (0:Disabled; 1:Enabled)
    - Bit 7 --> Monitor output Enable (0:Disabled; 1:Enabled)

- if SrcType = 2: ouput channels (PRE-GEQ) recording enable
    - Bit 0 --> Main output Enable (0:Disabled; 1:Enabled)
    - Bit 1 --> Aux1 output Enable (0:Disabled; 1:Enabled)
    - Bit 2 --> Aux2 output Enable (0:Disabled; 1:Enabled)
    - Bit 3 --> Aux3 output Enable (0:Disabled; 1:Enabled)
    - Bit 4 --> Aux4 output Enable (0:Disabled; 1:Enabled)
    - Bit 5 --> Aux5 output Enable (0:Disabled; 1:Enabled)
    - Bit 6 --> Aux6 output Enable (0:Disabled; 1:Enabled)
    - Bit 7 --> Monitor output Enable (0:Disabled; 1:Enabled)

- EnableRightH: USB recording right channel source (MSB)
- if SrcType = 0: input channels recording enable
    - Bit 0 --> CH25 Enable (0:Disabled; 1:Enabled)
    - Bit 1 --> CH26 Enable (0:Disabled; 1:Enabled)
    - Bit 2 --> CH27 Enable (0:Disabled; 1:Enabled)
    - Bit 3 --> CH28 Enable (0:Disabled; 1:Enabled)
    - Bit 4 --> CH29 Enable (0:Disabled; 1:Enabled)
    - Bit 5 --> CH30 Enable (0:Disabled; 1:Enabled)
- if SrcType > 0: unused

- EnableRight24: USB recording right channel source

- if SrcType = 0: input channels recording enable
  - - Bit 0 --> CH17 Enable (0:Disabled; 1:Enabled)
  - - Bit 1 --> CH18 Enable (0:Disabled; 1:Enabled)
  - - Bit 2 --> CH19 Enable (0:Disabled; 1:Enabled)
  - - Bit 3 --> CH20 Enable (0:Disabled; 1:Enabled)
  - - Bit 4 --> CH21 Enable (0:Disabled; 1:Enabled)
  - - Bit 5 --> CH22 Enable (0:Disabled; 1:Enabled)
  - - Bit 6 --> CH23 Enable (0:Disabled; 1:Enabled)
  - - Bit 7 --> CH24 Enable (0:Disabled; 1:Enabled)
- if SrcType > 0: unused

- EnableRight16: USB recording right channel source
- if SrcType = 0: input channels recording enable
  - - Bit 0 --> CH09 Enable (0:Disabled; 1:Enabled)
  - - Bit 1 --> CH10 Enable (0:Disabled; 1:Enabled)
  - - Bit 2 --> CH11 Enable (0:Disabled; 1:Enabled)
  - - Bit 3 --> CH12 Enable (0:Disabled; 1:Enabled)
  - - Bit 4 --> CH13 Enable (0:Disabled; 1:Enabled)
  - - Bit 5 --> CH14 Enable (0:Disabled; 1:Enabled)
  - - Bit 6 --> CH15 Enable (0:Disabled; 1:Enabled)
  - - Bit 7 --> CH16 Enable (0:Disabled; 1:Enabled)
- if SrcType > 0: unused

- EnableRightL: USB recording right channel source (LSB)
- if SrcType = 0: input channels recording enable
  - - Bit 0 --> CH01 Enable (0:Disabled; 1:Enabled)
  - - Bit 1 --> CH02 Enable (0:Disabled; 1:Enabled)
  - - Bit 2 --> CH03 Enable (0:Disabled; 1:Enabled)
  - - Bit 3 --> CH04 Enable (0:Disabled; 1:Enabled)
  - - Bit 4 --> CH05 Enable (0:Disabled; 1:Enabled)
  - - Bit 5 --> CH06 Enable (0:Disabled; 1:Enabled)
  - - Bit 6 --> CH07 Enable (0:Disabled; 1:Enabled)
  - - Bit 7 --> CH08 Enable (0:Disabled; 1:Enabled)

- if SrcType = 1: ouput channels recording enable
  - - Bit 0 --> Main output Enable (0:Disabled; 1:Enabled)
  - - Bit 1 --> Aux1 output Enable (0:Disabled; 1:Enabled)
  - - Bit 2 --> Aux2 output Enable (0:Disabled; 1:Enabled)
  - - Bit 3 --> Aux3 output Enable (0:Disabled; 1:Enabled)
  - - Bit 4 --> Aux4 output Enable (0:Disabled; 1:Enabled)
  - - Bit 5 --> Aux5 output Enable (0:Disabled; 1:Enabled)
  - - Bit 6 --> Aux6 output Enable (0:Disabled; 1:Enabled)
  - - Bit 7 --> Monitor output Enable (0:Disabled; 1:Enabled)

- if SrcType = 2: ouput channels (PRE-GEQ) recording enable
  - - Bit 0 --> Main output Enable (0:Disabled; 1:Enabled)
  - - Bit 1 --> Aux1 output Enable (0:Disabled; 1:Enabled)
  - - Bit 2 --> Aux2 output Enable (0:Disabled; 1:Enabled)
  - - Bit 3 --> Aux3 output Enable (0:Disabled; 1:Enabled)

- - Bit 4 --> Aux4 output Enable (0:Disabled; 1:Enabled)
- - Bit 5 --> Aux5 output Enable (0:Disabled; 1:Enabled)
- - Bit 6 --> Aux6 output Enable (0:Disabled; 1:Enabled)
- - Bit 7 --> Monitor output Enable (0:Disabled; 1:Enabled)

*Note: for SrcType = 1 or 2, only one source channel should be enabled at a time for left and right USB recording channel. For example it should be avoided to enable Aux1 and Aux2 bits in the "*EnableLeftL" (or "EnableRightL") command field. Moreover, *when the main or monitor output is enabled for left (or right) USB recording channel, the same selection should be forced in the right (or left) USB recording channel.*

Please refer to "UsbRecSrcType", "UsbRecInpChEnable", "UsbRecPreGeqEnable", "UsbRecOutEnable" params in the MemoryMap doc.

Response (14 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3
F0H	18H	xx	33H	SrcType	EnableLeftH	EnableLeft24	EnableLeft16

D4	D5	D6	D7	D8	ETX
EnableLeftL	EnableRightH	EnableRight24	EnableRight16	EnableRightL	F7H

CMD_SET_USB_REC_FILE_SPAN (34H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	34H	Span	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- Span: USB recording file span time in the range [1,12]

Span Time [minutes] = Span * 10

Please refer to “UsbRecFileSpan” param in the MemoryMap doc.

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	34H	Span	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_RECALL_PRESET (37H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	37H	PresetNr	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- PresetNr: the preset number to recall in the range [0,2]

Please refer to “ActualPreset” param in the MemoryMap doc.

Note: the remote controller should read the actual preset data (through the CMD_READ_MEMORY command) after sending the CMD_RECALL_PRESET command to sync the GUI with the parameter set of the recalled preset. The remote controller should also update its ActualPreset number variable (equals to the recalled preset number) or, even, read the device ActualPreset variable (through the CMD_READ_MEMORY command).

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	37H	PresetNr	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_STORE_PRESET (38H)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	38H	PresetNr	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- PresetNr: the preset number/position in the range [0,2] where to store the actual parameter set

Note: the remote controller should write the new preset name (through the CMD_WRITE_MEMORY command) before sending the CMD_STORE_PRESET command. The remote controller should also update its ActualPreset number variable (equals to the stored preset number) or, even, read the device ActualPreset variable (through the CMD_WRITE_MEMORY command) after sending the CMD_STORE_PRESET command

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	38H	PresetNr	00H	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

CMD_SET_FIR_ENABLE (3BH)

Command (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	3BH	00H	EnFir	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

where:
- EnFir: [0,1] 0:FIR Bypassed; 1:FIR Enabled

Response (13 Bytes)

STX	ID_M	ID_N	CMD	D0	D1	D2	D3	D4	D5
F0H	18H	xx	3BH	00H	EnFir	00H	00H	00H	00H

D6	D7	ETX
00H	00H	F7H

Appendix 1. Devices Scanning

In order to search a M28 device in the network, the remote controller can scan a set of IP addresses sending TCP connect requests on the remote port 1001.

For each IP address, if the connection is refused, it means there is no M28 device owning that address; otherwise, if the TCP connection is established, the remote host may be a M28 and the remote controller can send a CMD_CONNECTION_OPEN command to be sure the remote host is a M28 and to get the device info (please refer to the CMD_CONNECTION_OPEN section). After sending the command, the remote controller should wait a while (about 0.5sec) for the command response. If the remote host doesn't send the command response it means it isn't a M28 device; on the contrary, if the command response is received, the remote host is a M28 and the remote controller should send the CMD_CONNECTION_CLOSE command and finally close the TCP connection.

Scanning procedure example:

- IP address range: from 192.168.0.1 to 192.168.0.100

1a) Try connecting the host 192.168.0.1 on port 1001. Is the connection established?

- NO: the remote host is not a M28.

- YES: go to point 1b)

1b) Send the CMD_CONNECTION_OPEN command.

1c) Wait the command response for up to 0.5sec. Is the response received?

- NO: the remote host is not a M28. Go to point 1e)

- YES: go to point 1d)

1d) Send the CMD_CONNECTION_CLOSE command.

1e) Close the TCP connection.

2a) Try connecting the host 192.168.0.2 on port 1001. Is the connection established?

- NO: the remote host is not a M28.

- YES: go to point 2b)

2b) Send the CMD_CONNECTION_OPEN command.

2c) Wait the command response for up to 0.5sec. Is the response received?

- NO: the remote host is not a M28. Go to point 2e)

- YES: go to point 2d)

2d) Send the CMD_CONNECTION_CLOSE command.

2e) Close the TCP connection.

….

100a) Try connecting the host 192.168.0.100 on port 1001. Is the connection established?

- NO: the remote host is not a M28.

- YES: go to point 100b)

100b) Send the CMD_CONNECTION_OPEN command.

100c) Wait the command response for up to 0.5sec. Is the response received?

- NO: the remote host is not a M28. Go to point 100e)

- YES: go to point 100d)

100d) Send the CMD_CONNECTION_CLOSE command.

100e) Close the TCP connection.

To speed-up the scanning procedure, it is suggested to run points 1x) to 100x) in parallel by using a thread for each IP address in the scanning list.

Appendix 2. Automatic Device Announcement

The M28 sends a periodic UDP broadcast message on port 10002 to announce itself in the network.

The UDP message is sent every 5sec both through the WiFi and Ethernet interfaces. It has the following format:

Message (25 bytes)

D0	D1	D2	D3	D4	D5	D6	D7	D8	D9
IDM	IF	Model	UDID1	UDID2	UDID3	UDID4	UDID5	UDID6	Name1

D10	D11	D12	D13	D14	D15	D16	D17	D18	D19
Name2	Name3	Name4	Name5	Name6	Name7	Name8	Name9	Name10	Name11

D20	D21	D22	D23	D24
Name12	Name13	Name14	Name15	Name16

where:
- IDM = 18H
- IF: interface (0:Ethernet; 1:WiFi)
- Model: Device model / extension card type (0: M16 Analog extension; 1: M16 Dante extension; 2: M16 SPDIF/Link extension ; 16: M28 Analog extension; 17: M28 Dante extension; 18: M28 SPDIF/Link extension)
- UDID1-6: Unique Device ID
    - Example:
    - UDID1=12H; UDID2=34H; UDID3=56H; UDID4=78H; UDID5=9AH; UDID6=BCH
    - Unique Device ID = “123456789ABC”

- Name1-16: Device name string (16 chars)

The remote controller should run a background UDP server listening on port 10002 and waiting for UDP broadcast messages. Once a UDP message is received, the remote controller can retrieve the remote IP address from the UDP stack and the M28 device info from the UDP message data fields.

The remote controller should also send a “ping” to the remote IP address to verify that the local host is able to reach the device in the network, since the reception of the UDP broadcast message cannot ensure that the M28 is reachable through TCP (due to a mismatch in the network settings).

In example:

- Remote controller

- IP address = 192.168.1.100
- Netmask = 255.255.255.0
- Gateway = 192.168.1.1

- M28

- IP address = 192.168.0.101
- Netmask = 255.255.255.0
- Gateway = 192.168.0.1

In the case above the remote controller will receive the M28 UDP message, but it won't be able to connect (or ping) the device.

Appendix 3. Remote Controllers Sync Procedure

The M28 uses UDP broadcast messages to inform all the active remote controllers that a parameter set is changed.

Every time the M28 receives a “Write” command (i.e. a command that modifies a set of parameters), it broadcasts that command to both WiFi and Ethernet networks, on UDP port 10002, with the following message preamble:

D0	D1	D2	D3	D4
CTRL	LenH	LenL	CntH	CntL

where:
- CTRL = FFH (Control code to discriminate between “Write” command broadcast and Device Announcement message)
- LenH: Message lenght (MSB)
- LenL: Message lenght (LSB)
- CntH: Message counter (MSB)
- CntL: Message counter (LSB)

The message length ((LenH << 8) + LenL) includes the preample length (5 bytes).
The message counter ((CntH << 8) + CntL) is the broadcast message progressive number (currently it is used for test purpose only and can be discarded by the remote controller)

The remote controller should run a background UDP server listening on port 10002 and waiting for UDP broadcast packets. Once an UDP packet is received, the remote controller should:

a) Read the message preample

b) Check if the CTRL field (i.e. the first byte of the message) is equal to FFH

c) Get the message length by the LenH/LenL bytes

d) Get the remaining data (message length minus 5 bytes) to retrieve the “Write” command fields that a remote controller sent to the M28.

e) Parse the “Write” command fields to understand which parameters are changed and update the relevant GUI controls.

f) Repeat points a) to e) untill there are no more UDP messages in the UDP packet.

WARNING: An UDP packet may contain more than one message.

Note: since UDP protocol doesn't ensure packet delivery it could happen (especially for WiFi network) to lose some broadcast messages. For this reason the remote controller should periodically read the M28 memory (every 5-10 seconds) and compare the parameter set with the actual remote controller memory buffer. When the remote controller finds a difference in one or more parameters it should update the relevant GUI controls.