PLENA matrix API

en

Table of contents

1 PLENA Matrix Network API

This chapter summarizes the Ethernet protocol and commands used by the Bosch PLM-8M8 DSP matrix mixer and PLM-4Px2x DSP amplifier products. It is intended for the implementation of 3rd party remote control applications of the devices.

Revision Control

Date	Notes
26th October 2012	Initial release
8th February 2013	Added extra checksum byte to POBJ Write packet
21st February 2013	New information/changes to: WHAT and EXPL packets (new user hardware name Unicode string added) PASS packet (now only one password, in Unicode) SMON packet (sending hardware now includes its IP address)
10th March 2013	Changed all Unicode strings to UTF-8 strings

1.1 Protocol Information

The communications protocol used with the hardware devices is UDP. The user data of the UDP packet consists of a header followed by the data.
For both devices (amplifier and processor), the hardware receives data on UDP port **12128** and transmits data on UDP port **12129**.
The header for each packet is 10 bytes in length and consists of the following:

Field	Size	Description
Protocol ID	2 bytes	0x5E40 (processor) or 0x5E41 (amplifier)
Sub Type	2 bytes	Set to 0x0001 for packets from a master Set to 0x0100 for packets from a slave
Sequence Number	2 bytes	Any value from 0x0001 to 0xFFFF
Reserved	2 bytes	Set to 0x0000
Chunk Length	2 bytes	Length in bytes of data after this header (does not include the length of the header)

The ProtocolId field differentiates this packet from others with a similar format used by other Secure Electronics products. Its value is specific and unique to these devices. The matrix processor responds to an ID of 0x5E40. Note that while there are two amplifier types (120W and 220W), both respond to an ID of 0x5E41.
The SubType field should be set to 0x0001 for all packets originating from a controller (or master). A master is typically a software GUI, iPad/iPhone application or Crestron/AMX controller. The hardware units, in their reply packets, will have this field set to 0x0100. Masters should discard any packets received that do not have this field set to 0x0100.

The SequenceNumber is a field that can be used by masters to place a unique ID on the packet. The hardware units will reply to a particular command with a matching SequenceNumber such that a master can match up a device reply to a previously sent packet. The sequence number can be any 16-bit number except 0.
The Reserved field should be set to 0x0000.
Lastly, the ChunkLength field contains the length of ALL SUBSEQUENT data in the packet (in bytes). Data chunks following the header are restricted to 272 bytes.

Note that due to the lossy nature of UDP and the fact that the hardware can support communication with multiple masters at the same time, masters should continually poll for changes in hardware state if they need to remain in synchronization. Information on how to perform this synchronization is provided later in this document.

1.2 Network Discovery

The easiest way to discover units on a network is to send a broadcast PING command on to the network. All hardware units that receive this packet will reply with an EXPL command that provides information on the device's IP details, MAC address, ID name and firmware version. As UDP is lossy, it is generally worth sending three or so PING commands within a second to maximize the chance of not missing any return packets.
The master can then select the hardware unit to communicate with (or allow the user to make this choice). Using a broadcast address of 255.255.255.255 will return all devices on the immediate network (note that network switches usually prevent this from going out on to the actual internet), but some of these devices will not be able to communicate if they are located on a different subnet. For this reason a subnet broadcast (for example 192.168.1.255) is recommended.
If the user has a fixed IP address to communicate with, then the master can send the PING in non-broadcast mode to that specific address and wait and see if a WHAT command is received. Non-broadcast PING packets reply with WHAT commands instead of EXPL commands.
Note that a master can screen the hardware devices found based on firmware version, such that if a master has been programmed to suit a particular firmware version it will not communicate with a device of a different firmware version. In this situation the user should be instructed that this is the case.
Note that the PING, EXPL and WHAT commands are defined further in this document.

1.3 Connection Initiation

To initiate a connection with a particular hardware unit, a master unit should perform the following tasks:

1. Request password information from the hardware. This is done by sending a PASS command packet, which results in the slave replying with password information. Note that this password information is unencrypted within the packet as the aim of the password is to prevent *unintended* access rather than *unauthorized* access.
2. If the password information is enforced then the master should prompt the user for the password. If the user cannot enter the password correctly the master should prevent the connection from being established and inform the user of the failure.

3. If the password was required and if so has been entered correctly, the master unit can then request the device send its current configuration. This can be done by requesting a complete set of SYNC command packets from the slave.
4. After all SYNC packets are received, the master unit should update any of its GUI controls and then allow the user access to start controlling the hardware.

As the system is a multi-master system, there are times when online masters may “seize” the hardware to do a bulk download of system parameters. In this situation there is a carefully arbitrated mechanism to ensure that all other masters can be informed of this and then resynchronize. If a master ever receives an IGNO command from the slave hardware in response to a packet this indicates that the master has been locked out from the unit temporarily. In this situation the master should prevent further user interaction until the hardware has been released, and it should then attempt to resynchronize by requesting a complete set of SYNC packets before allowing further user interaction. As there could be several devices performing this resynchronization when the lock out is released it is recommended that the SYNC requests are randomly within 3-5 seconds after the lock out release has occurred.
It is highly recommended that to detect lock outs, the master sends a SEIZ command with a poll sub type every 3 seconds. This packet returns whether or not the device is locked out and by which IP address (which allows a master to know if it has the lock out). It also contains a 32-bit number representing how many times the hardware has been “seized”. Masters should keep a record of this, and if for any reason this count received from the hardware is greater than the last known value the master has stored, it should immediately attempt to request a full set of SYNC packets as soon as the device has become available again (allowing for randomization as noted in the previous paragraph), and prevent user interaction until this has occurred.

1.4 Parameter Control & Signal Monitoring

Various parameters within the hardware can be controlled by the master unit. These are controlled with the GOBJ, POBJ and PSET commands which are further outlined in this document.
In general, master units should implement a speed restriction on which command updates can be sent to the hardware. The general recommendation is approximately no faster than every 150msec (0.15sec).
As an example of this, if the user is controlling a volume fader on an input, the first change should occur immediately and then further changes limited to the (say) 0.15 second recommendation.
When a command to change a parameter is sent to the hardware unit, it will reply with an ACKN packet to indicate successful reception. It is up to the master to ensure successful packet delivery given that UDP is lossy.

Signal monitoring information (required for metering purposes) can be requested by sending a SMON command every 5 seconds. When an ACKN packet is replied, the hardware unit will broadcast SMON reply packets containing the monitoring data over the UDP network. If no SMON command has been received by the hardware for around 13 seconds it will no longer broadcast the reply packets.

Reply packets are broadcast at a rate of approximately 8Hz (so every 0.125 seconds or so). This is to ensure that metering is responsive on any GUIs. Broadcast packets are used so that if multiple masters are communicating with the hardware the network is not overcome with monitoring packets. The SMON packet contains the IP address of the hardware sending the packet at the end. ALL MASTERS MUST CHECK THIS IP AND ENSURE THAT THE PACKET CAME FROM THE HARDWARE THEY ARE CONNECTED TO. This is essential to avoid other broadcast packets from Bosch hardware that may be on the network from being errantly displayed.

Hardware devices are capable of storing presets. These presets contain full copies of DSP parameters that upon being recalled are copied into the LIVE PRESET, which while not a stored preset is the current mirror of active parameters engaged in the device. Any parameters altered from the main controls of masters should change settings in the live preset.

1.5 Command Packet: PING (Ping Hardware)

Description:
This command is used to discover information about a device (or devices) that are present on the network. It includes an option to force the hardware to return a broadcast (instead of unicast) UDP response packet for use in detecting hardware devices that may exist on the same physical network but on a different subnet.
Packet Structure (unicast reply requested):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	PING (ASCII 0x50494E47)

Packet Structure (broadcast reply requested):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	PING (ASCII 0x50494E47)
Broadcast Flag	1 byte	Set to 0x11 (all other values will result in a unicast reply)

The reply to a PING command when a unicast reply is requested is a WHAT command packet. The reply to a PING command when a broadcast reply is requested is an EXPL command packet.
Note that PING packets will be replied to by a hardware unit, regardless of whether or not the hardware has been “seized” by another master (via the SEIZ packet). This allows masters to continue to use PING packets to determine if the hardware unit is still online.

1.6 Command Packet: WHAT (Hardware ID Reply)

Description:
This packet is received from a hardware unit in response to a PING packet (in a non-broadcast request). It contains device information including firmware version, MAC address and IP details as well as any build-state considerations.
Packet Structure:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	WHAT (ASCII 0x57484154)
Firmware Info (Major)	1 byte	Value ranges from 0-255
Firmware Info (Minor)	1 byte	Value ranges from 0-255
Firmware Info (Rev)	2 bytes	Value ranges from 0-65535
MAC Address	6 bytes	Network MAC address of the hardware unit
IP Address	4 bytes	Network IP address (IPv4) of the hardware unit
Subnet Mask	4 bytes	Subnet mask IP address (IPv4) of the hardware unit
Default Gateway	4 bytes	Default gateway address (IPv4) of the hardware unit
DHCP Enabled Flag	1 byte	0 if the hardware unit has its DHCP client disabled 1 if the hardware unit has its DHCP client enabled
Custom Mode Info	1 byte	Data byte used by the hardware unit for identification purposes
Lock Out Flag	1 byte	0 if the hardware unit is free to communicate with the master 1 if this master is locked out from the hardware unit
Device Name	32 bytes	The manufacturer/product ID string (ASCII string)
User Hardware Name	81 bytes	81 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

Note that the Custom Mode Info field varies depending on the type of hardware unit.
For the matrix processor the value can be disregarded. For the amplifier, the quad 120W variant will set this field to 0x00 and the quad 220W variant will set this field to 0x01.
For master applications displaying a device name, the “User Hardware Name” field is for this purpose.

1.7 Command Packet: EXPL (Hardware ID Reply)

Description:
This packet is received from a hardware unit in response to a PING packet (in a broadcast request). It contains device information including firmware version, MAC address and IP details as well as any build-state considerations. It differs from the WHAT packet by including the IP address of the original requesting master in the data chunk, such that a master can determine if the reply was sent to it (which would normally be impossible to tell from the standard UDP header due to the fact that this packet is broadcast).
Packet Structure:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	EXPL (ASCII 0x4558504C)
Firmware Info (Major)	1 byte	Value ranges from 0-255
Firmware Info (Minor)	1 byte	Value ranges from 0-255
Firmware Info (Rev)	2 bytes	Value ranges from 0-65535
MAC Address	6 bytes	Network MAC address of the hardware unit
IP Address	4 bytes	Network IP address (IPv4) of the hardware unit
Subnet Mask	4 bytes	Subnet mask IP address (IPv4) of the hardware unit
Default Gateway	4 bytes	Default gateway address (IPv4) of the hardware unit
DHCP Enabled Flag	1 byte	0 if the hardware unit has its DHCP client disabled 1 if the hardware unit has its DHCP client enabled
Custom Mode Info	1 byte	Data byte used by the hardware unit for identification purposes
Lock Out Flag	1 byte	0 if the hardware unit is free to communicate with the master 1 if this master is locked out from the hardware unit
Device Name	32 bytes	The user defined name of the hardware unit (ASCII string)
Target IP Address	4 bytes	Network IP address (IPv4) of the master that this packet has been specifically sent at
User Hardware Name	81 bytes	81 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

Note that the Custom Mode Info field varies depending on the type of hardware unit.

For the matrix processor the value can be disregarded. For the amplifier, the quad 120W variant will set this field to 0x00 and the quad 220W variant will set this field to 0x01.
For master applications displaying a device name, the “User Hardware Name” field is for this purpose.

1.8 Command Packet: PASS (Password Information)

Description:
This packet is sent by a master to request the password information stored by a device.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	PASS (ASCII 0x50415353)

The hardware unit will then reply with a PASS packet, defined as:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	PASS (ASCII 0x50415353)
Hardware Password Enforced	1 byte	0 if the hardware password is not required to access the unit 1 if the hardware password is required to access the unit
Hardware Password	31 bytes	Hardware password - 31 byte length UTF-8 encoded string. Passwords not requiring the full length are padded with bytes set to 0x00

1.9 Command Packet: SEIZ (Hardware Seize/Lock-Out)

Description:
This packet is sent by a master to perform a variety of lock out functions. For 3rd-party systems this command is only required to poll the lock-out state of the hardware.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	SEIZ (ASCII 0x5345495A)
Sub Command	1 byte	0x1F (Lock-out state poll)

The hardware unit will then reply with a SEPL packet, defined as:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SEPL (ASCII 0x5345504C)
Lock-Out State	1 byte	0 if the hardware has not been locked-out 1 if the hardware has been locked out by a particular master
Lock-Out Owner	4 bytes	Network IP address (IPv4) of the master that the hardware has been locked out by
Lock-Out Count	4 bytes	The number of times the hardware has been locked out This is an unsigned 32-bit integer

1.10 Command Packet: SMON (Signal Monitoring)

Description:
This packet is sent by a master to instruct the hardware unit to start (or maintain) sending broadcast monitoring packets. These packets contain data that represents the current signal level at various points of the audio structure in the device.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	SMON (ASCII 0x534D4F4E)
Sub Command	1 byte	0x11 (Enable broadcast monitoring)

The hardware unit will then reply with an ACKN packet to confirm the command was received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	ACKN (ASCII 0x41434B4E)

Once triggered the monitoring packets will be sent from the hardware automatically for 13 seconds. Upon receiving another SMON packet as outlined above, this counter will restart and packets will continue.
These broadcast packets containing the signal monitoring data have the following structure:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SMON (ASCII 0x534D4F4E)

Packet Field	Size	Description
Monitoring Data	<x bytes>	Monitoring data Specific to hardware
Sending IP Address	4 bytes	The IP address of the hardware sending this reply packet

Note that the exact length and information contained in the monitoring information is not described in this document as it is hardware specific. It is contained in a separate document written specifically for that hardware.

1.11 Command Packet: GOBJ (Global Object) Write

Description:
This packet is sent by a master to set (write) a global object parameter in the hardware.
Packet Structure (master sent, WRITE):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	GOBJ (ASCII 0x474F424A)
Is Read Flag	1 byte	0x00 (write)
Global Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)
NV Commit Flag	1 byte	0x00 (do not commit to device NV memory) 0x01 (commit to NV memory)
Object Data	<x> bytes	Data for the object - length depends on the data object

The hardware unit will then reply with an ACKN packet to confirm the command was received and processed correctly:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	ACKN (ASCII 0x41434B4E)

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00030001 = Corrupt packet error

NACK Code 0x00030002 = Bad global object ID
NACK Code 0x00030003 = NV operation failure (the NV EEPROM failed to store the value)
NACK Code 0x00030004 = RAM operation failure (the RAM failed to store the value)
NACK Code 0x00030005 = Incorrect hardware state (the hardware is not ready to use this packet)

Exact global object information is dependent on the hardware device and is contained in a separate document written specifically for that hardware.

1.12 Command Packet: GOBJ (Global Object) Read

Description:
This packet is sent by a master to set (read) a global object parameter in the hardware.
Packet Structure (master sent, READ):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	GOBJ (ASCII 0x474F424A)
Is Read Flag	1 byte	0x01 (read)
Global Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)

The hardware unit will then reply with a GOBJ packet to confirm the command was received which contains the data for the requested global object.

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	GOBJ (ASCII 0x474F424A)
Read Flag Confirm	1 byte	0x01 (confirms read operation)
Global Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)
Object Data	<x> bytes	Data for the object - length depends on the data object

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00030001 = Corrupt packet error
NACK Code 0x00030002 = Bad global object ID

NACK Code 0x00030005 = Incorrect hardware state (the hardware is not ready to use this packet)

Exact global object information is dependent on the hardware device and is contained in a separate document written specifically for that hardware.

1.13 Command Packet: POBJ (Preset Object) Write

Description:
This packet is sent by a master to set (write) a preset object parameter in the hardware.
Packet Structure (master sent, WRITE):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	POBJ (ASCII 0x504F424A)
Is Read Flag	1 byte	0x00 (write)
Preset Number	1 byte	0 = Live Preset 1-5 = Preset Number
Preset Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)
NV Commit Flag	1 byte	0x00 (do not commit to device NV memory) 0x01 (commit to NV memory)
Object Data	<x> bytes	Data for the object - length depends on the data object
Object Checksum	1 byte	Set to 0x00 (different values only required when the object being sent is not in the quick set list which is beyond the scope of this document)

The hardware unit will then reply with an ACKN packet to confirm the command was received and processed correctly:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	ACKN (ASCII 0x41434B4E)

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00040001 = Corrupt packet error
NACK Code 0x00040002 = Bad preset object ID
NACK Code 0x00040003 = NV operation failure (the NV EEPROM failed to store the value)
NACK Code 0x00040004 = RAM operation failure (the RAM failed to store the value)
NACK Code 0x00040005 = Incorrect hardware state (the hardware is not ready to use this packet)
NACK Code 0x00040006 = Bad preset number

Exact preset object information is dependent on the hardware device and is contained in a separate document written specifically for that hardware.

1.14 Command Packet: POBJ (Preset Object) Read

Description:
This packet is sent by a master to get (read) a global object parameter in the hardware.
Packet Structure (master sent, READ):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	POBJ (ASCII 0x504F424A)
Is Read Flag	1 byte	0x01 (read)
Preset Number	1 byte	0 = Live Preset 1-5 = Preset Number
Preset Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)

The hardware unit will then reply with a POBJ packet to confirm the command was received which contains the data for the requested global object.

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	POBJ (ASCII 0x504F424A)
Read Flag Confirm	1 byte	0x01 (confirms read operation)
Preset Number	1 byte	0 = Live Preset 1-5 = Preset Number
Preset Object ID	2 bytes	0x0000 - 0xFFFF (hardware device dependent)
Object Data	<x> bytes	Data for the object - length depends on the data object

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00040001 = Corrupt packet error
NACK Code 0x00040002 = Bad preset object ID
NACK Code 0x00040005 = Incorrect hardware state (the hardware is not ready to use this packet)
NACK Code 0x00040006 = Bad preset number

Exact preset object information is dependent on the hardware device and is contained in a separate document written specifically for that hardware.

1.15 Command Packet: PSET (Preset) Change

Description:
This packet is sent by a master to engage a stored preset in the hardware into the live preset.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	PSET (ASCII 0x50534554)
Preset Sub Cmd 1	1 byte	1-5 = Preset Number
Preset Sub Cmd 2	1 byte	Same as preset sub cmd 1
Clear Seize Flag	1 byte	0x00 (not used by remote masters)

The hardware unit will then reply with an ACKN packet to confirm the command was received and processed correctly:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	ACKN (ASCII 0x41434B4E)

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00090001 = Incorrect hardware state (the hardware is in a state that prevents a preset change)
NACK Code 0x00090002 = Bad preset number requested

1.16 Command Packet: PSET (Preset) In-Use Request

Description:
This packet is sent by a master to request which presets stored in the device are currently valid. As presets slow down bulk state read and write operations between masters and slaves, this allows a master to copy only what is being used. Alternatively, it also allows a master to allow preset changes only if presets are in use.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	PSET (ASCII 0x50534554)
Preset Sub Cmd 1	1 byte	0xFF
Preset Sub Cmd 2	1 byte	0xFF
Clear Seize Flag	1 byte	0x00 (not used by remote masters)

The hardware unit will then reply with a PSET packet to confirm the command was received and processed correctly:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	PSET (ASCII 0x50534554)
Preset Sub Cmd 1	1 byte	0xFF
Preset Sub Cmd 2	1 byte	0xFF
Preset 1 Valid	1 byte	0x00 = not in use 0x01 = in use
Preset 2 Valid	1 byte	0x00 = not in use 0x01 = in use
Preset 3 Valid	1 byte	0x00 = not in use 0x01 = in use
Preset 4 Valid	1 byte	0x00 = not in use 0x01 = in use
Preset 5 Valid	1 byte	0x00 = not in use 0x01 = in use

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00090001 = Incorrect hardware state (the hardware has not yet determined the valid flags)

1.17 Command Packet: SYNC (Device Synchronization)

Description:
This packet is sent by a master to request a bulk load of synchronization data containing many global objects or preset objects from the live preset. It allows masters to ensure they are in sync with the current state of the hardware.
Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	ID of the SYNC return packet

The hardware unit will then reply with a SYNC packet to confirm the command was received and processed correctly:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Data	<x> bytes	The data contents stored in the sync packet

If the command could not be processed (but was received OK), a NACK packet will be sent:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	NACK (ASCII 0x4E41434B)
NACK Code	4 bytes	Unsigned 32-bit integer

NACK Code 0x00070001 = Incorrect hardware state (The hardware is not ready to send a SYNC packet)
NACK Code 0x00070002 = Corrupt packet error
NACK Code 0x00070003 = Bad SYNC request

Exact synchronization packet information is dependent on the hardware device and is contained in a separate document written specifically for that hardware.

2 PLM-8M8 DSP Matrix Mixer Specific API

This chapter summarizes parameters, commands and packets specific to the PLM-8M8 DSP matrix mixer hardware. These include preset objects, global objects, signal monitoring and synchronization control aspects.

Revision Control

Date	Notes
5th December 2012	Initial release
22nd January 2012	Updated “Signal Monitoring” section to suit device firmware 0.7.0 changes
8th February 2013	Added checksum byte reference in POBJ Write packet structure
21st February 2013	SMON packet altered to include sending IP address to data payload GOBJ_GLOBAL_MUTE_ALLOWED global object number changed GOBJ_FORCE_GLOBAL_MUTE global object number changed SYNC packet changes - now four SYNC packet types, with different data layout Firmware release at v1.0.0
10th March 2013	Changed all Unicode strings to UTF-8 strings

2.1 Signal Monitoring

Signal monitoring data enables applications to display the current value of audio signal levels at various signal nodes in the hardware. This data is represented in 32-bit unsigned integers that can be converted into a double type value via the following C code:

```
double ConvertToMonitoring(UInt32 rawValue)
{
    if (rawValue > 0x0007FFFF)
    {
        return 0.0;
    }
    else
    {
        double frac = (double)rawValue / (double)0x0007FFFF;
        double db = 20.0 * Math.Log10(frac);
        if (db < -100.0)
        {
            return -100.0;
        }
        else
        {
            return db;
```

```
            }
        }
    }
```

In audio terms, returning 0.0dB indicates full-scale audio (this level would result in analog audio in/out being clipped). The scale is negative, with lower values indicating lower audio levels. Below -50.0dB this monitoring starts to become inaccurate, so use of monitoring data should be restricted between 0.0 and -50.0dB.
Consult the master API document for details on how applications start requesting SMON packets from device hardware.
The matrix processor SMON packet that gets sent to master applications is structured as per the table below. There are 41 monitoring signal values contained, as well as the current paging activity into each of the output zones and the IP address of the hardware that is sending the packet in reply.

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SMON (ASCII 0x534D4F4E)
Monitoring Data Index 0	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Monitoring Data Index 1	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Monitoring Data Index 2	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
...	...	...
Monitoring Data Index 40	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Zone Paging Activity (Zones 1-8)	8 bytes	Each byte (in order, zone 1 to zone 8): 0 if no paging activity in zone 1-8 if paging is occurring, indicating ID of owning paging station
Sender IP Address	4 bytes	The 4 byte IP address of the hardware sending this packet back to the master

Of the 41 monitoring indexes contained in the matrix processor SMON packet, the following indexes are the ones useful to applications:

Monitoring Index	Description
0	Mic/Line Input 1 (Pre Zone Mixer)
1	Mic/Line Input 2 (Pre Zone Mixer)

Monitoring Index	Description
2	Mic/Line Input 3 (Pre Zone Mixer)
3	Mic/Line Input 4 (Pre Zone Mixer)
4	BGM Input 1 (Pre BGM Select)
5	BGM Input 2 (Pre BGM Select)
6	BGM Input 3 (Pre BGM Select)
7	Paging Input (Pre Paging Master)
8	Override Input (Pre Override Master)
17	Zone 1 Master Mix
18	Zone 2 Master Mix
19	Zone 3 Master Mix
20	Zone 4 Master Mix
21	Zone 5 Master Mix
22	Zone 6 Master Mix
23	Zone 7 Master Mix
24	Zone 8 Master Mix

2.2 Global Objects

The GOBJ packet is used to set or get the state of a system global object.
For 3rd-party applications, these packets should be used to control individual user parameters. Using the GOBJ packet to get values of parameters is not recommended as the SYNC packet contains more information and is faster for the same purpose.
The structure of a write (set) GOBJ packet is:

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	GOBJ (ASCII 0x474F424A)
Is Read Flag	1 byte	0x00 (write)
Global Object ID	2 bytes	16-bit number as defined in table below
NV Commit Flag	1 byte	0x00 (do not commit to device NV memory) 0x01 (commit to NV memory)

Packet Field	Size	Description
Object Data	<x> bytes	Data for the object - length depends on the data object

As NV memory has a limited lifespan, if an object is going to be toggled often by an external application, care should be taken to ensure the NV commit is performed once the object state has settled.
The table below lists global objects specifically designed to be accessed via 3rd-party control applications:

Global Object Number	Global Object Name	Data Description (Object Type)
46	GOBJ_STANDBY_ALLOWED	Boolean Block
47	GOBJ_FORCE_STANDBY	Boolean Block
50	GOBJ_GLOBAL_MUTE_ALLOWED	Boolean Block
51	GOBJ_FORCE_GLOBAL_MUTE	Boolean Block

In regards to the STANDBY global object, if GOBJ_STANDBY_ALLOWED is 0x00, then the object state of GOBJ_FORCE_STANDBY will be ignored (as the hardware is not allowed to enter standby mode).
GOBJ_FORCE_STANDBY is the global object to use to toggle the hardware into and out of standby mode.
In regards to the GLOBAL_MUTE global object, if GOBJ_GLOBAL_MUTE_ALLOWED is 0x00, then the object state of GOBJ_FORCE_GLOBAL_MUTE will be ignored (as the hardware is not allowed to engage the global mute).
GOBJ_FORCE_GLOBAL_MUTE is the global object to use to toggle the hardware into and out of global mute.
The C structure for the Boolean block is defined as below and is 2 bytes in length:

```
typedef struct
{
 uint8 P_BooleanValue;
 uint8 P_CheckField;
} BooleanBlock;
```

P_BooleanValue should be set to either 0x00 (false) or 0x01 (true). The value of P_CheckField is used as a verification for P_BooleanValue.
If P_BooleanValue = 0x00, set P_CheckField to 0x64
If P_BooleanValue = 0x01, set P_CheckField to 0x3A

2.3 Preset Objects

The POBJ packet is used to set or get the state of a preset object (usually in the live preset, but it can be used to do the same with stored presets as well).

For 3rd-party applications, these packets should be used to control individual user parameters. Using the POBJ packet to get values of parameters is not recommended as the SYNC packet contains more information and is faster for the same purpose.
The structure of a write (set) POBJ packet is:

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	POBJ (ASCII 0x504F424A)
Is Read Flag	1 byte	0x00 (write)
Preset Number	1 byte	0 = Live Preset
Preset Object ID	2 bytes	16-bit number as defined in table below
NV Commit Flag	1 byte	0x00 (see comment below)
Object Data	<x> bytes	Data for the object - length depends on the data object, e.g. Object Data: DSP Volume Lut Block Object Data: DSP Mutex Block
Object Checksum	1 byte	Set to 0x00 (not required as NV storage is managed by hardware)

Notice!
For all the preset objects accessible by external applications, the hardware automatically manages NV memory storage, so the NV Commit Flag should be set to 0x00.

The table below lists preset objects specifically designed to be accessed via 3rd-party control applications:

Preset object number	Preset object name	Data description (DSP block type)
81	POBJ_Z1_CB_ML1_LEVEL Zone 1 Mic/Line 1 Mix Level	DSP Volume LUT Block
82	POBJ_Z1_CB_ML2_LEVEL Zone 1 Mic/Line 2 Mix Level	DSP Volume LUT Block
83	POBJ_Z1_CB_ML3_LEVEL Zone 1 Mic/Line 3 Mix Level	DSP Volume LUT Block
84	POBJ_Z1_CB_ML4_LEVEL Zone 1 Mic/Line 4 Mix Level	DSP Volume LUT Block
85	POBJ_Z1_CB_BGM_MUTEX Zone 1 BGM Select & Level	DSP Mutex Block

Preset object number	Preset object name	Data description (DSP block type)
86	POBJ_Z1_CB_MASTER_LEVEL Zone 1 Master Mix Level	DSP Volume LUT Block
87	POBJ_Z1_CB_PAGING_LEVEL Zone 1 Paging Mix Level	DSP Volume LUT Block
104	POBJ_Z2_CB_ML1_LEVEL Zone 2 Mic/Line 1 Mix Level	DSP Volume LUT Block
105	POBJ_Z2_CB_ML2_LEVEL Zone 2 Mic/Line 2 Mix Level	DSP Volume LUT Block
106	POBJ_Z2_CB_ML3_LEVEL Zone 2 Mic/Line 3 Mix Level	DSP Volume LUT Block
107	POBJ_Z2_CB_ML4_LEVEL Zone 2 Mic/Line 4 Mix Level	DSP Volume LUT Block
108	POBJ_Z2_CB_BGM_MUTEX Zone 2 BGM Select & Level	DSP Mutex Block
109	POBJ_Z2_CB_MASTER_LEVEL Zone 2 Master Mix Level	DSP Volume LUT Block
110	POBJ_Z2_CB_PAGING_LEVEL Zone 2 Paging Mix Level	DSP Volume LUT Block
127	POBJ_Z3_CB_ML1_LEVEL Zone 3 Mic/Line 1 Mix Level	DSP Volume LUT Block
128	POBJ_Z3_CB_ML2_LEVEL Zone 3 Mic/Line 2 Mix Level	DSP Volume LUT Block
129	POBJ_Z3_CB_ML3_LEVEL Zone 3 Mic/Line 3 Mix Level	DSP Volume LUT Block
130	POBJ_Z3_CB_ML4_LEVEL Zone 3 Mic/Line 4 Mix Level	DSP Volume LUT Block
131	POBJ_Z3_CB_BGM_MUTEX Zone 3 BGM Select & Level	DSP Mutex Block
132	POBJ_Z3_CB_MASTER_LEVEL Zone 3 Master Mix Level	DSP Volume LUT Block
133	POBJ_Z3_CB_PAGING_LEVEL Zone 3 Paging Mix Level	DSP Volume LUT Block
150	POBJ_Z4_CB_ML1_LEVEL Zone 4 Mic/Line 1 Mix Level	DSP Volume LUT Block

Preset object number	Preset object name	Data description (DSP block type)
151	POBJ_Z4_CB_ML2_LEVEL Zone 4 Mic/Line 2 Mix Level	DSP Volume LUT Block
152	POBJ_Z4_CB_ML3_LEVEL Zone 4 Mic/Line 3 Mix Level	DSP Volume LUT Block
153	POBJ_Z4_CB_ML4_LEVEL Zone 4 Mic/Line 4 Mix Level	DSP Volume LUT Block
154	POBJ_Z4_CB_BGM_MUTEX Zone 4 BGM Select & Level	DSP Mutex Block
155	POBJ_Z4_CB_MASTER_LEVEL Zone 4 Master Mix Level	DSP Volume LUT Block
156	POBJ_Z4_CB_PAGING_LEVEL Zone 4 Paging Mix Level	DSP Volume LUT Block
173	POBJ_Z5_CB_ML1_LEVEL Zone 5 Mic/Line 1 Mix Level	DSP Volume LUT Block
174	POBJ_Z5_CB_ML2_LEVEL Zone 5 Mic/Line 2 Mix Level	DSP Volume LUT Block
175	POBJ_Z5_CB_ML3_LEVEL Zone 5 Mic/Line 3 Mix Level	DSP Volume LUT Block
176	POBJ_Z5_CB_ML4_LEVEL Zone 5 Mic/Line 4 Mix Level	DSP Volume LUT Block
177	POBJ_Z5_CB_BGM_MUTEX Zone 5 BGM Select & Level	DSP Mutex Block
178	POBJ_Z5_CB_MASTER_LEVEL Zone 5 Master Mix Level	DSP Volume LUT Block
179	POBJ_Z5_CB_PAGING_LEVEL Zone 5 Paging Mix Level	DSP Volume LUT Block
196	POBJ_Z6_CB_ML1_LEVEL Zone 6 Mic/Line 1 Mix Level	DSP Volume LUT Block
197	POBJ_Z6_CB_ML2_LEVEL Zone 6 Mic/Line 2 Mix Level	DSP Volume LUT Block
198	POBJ_Z6_CB_ML3_LEVEL Zone 6 Mic/Line 3 Mix Level	DSP Volume LUT Block
199	POBJ_Z6_CB_ML4_LEVEL Zone 6 Mic/Line 4 Mix Level	DSP Volume LUT Block
200	POBJ_Z6_CB_BGM_MUTEX Zone 6 BGM Select & Level	DSP Mutex Block

Preset object number	Preset object name	Data description (DSP block type)
201	POBJ_Z6_CB_MASTER_LEVEL Zone 6 Master Mix Level	DSP Volume LUT Block
202	POBJ_Z6_CB_PAGING_LEVEL Zone 6 Paging Mix Level	DSP Volume LUT Block
219	POBJ_Z7_CB_ML1_LEVEL Zone 7 Mic/Line 1 Mix Level	DSP Volume LUT Block
220	POBJ_Z7_CB_ML2_LEVEL Zone 7 Mic/Line 2 Mix Level	DSP Volume LUT Block
221	POBJ_Z7_CB_ML3_LEVEL Zone 7 Mic/Line 3 Mix Level	DSP Volume LUT Block
222	POBJ_Z7_CB_ML4_LEVEL Zone 7 Mic/Line 4 Mix Level	DSP Volume LUT Block
223	POBJ_Z7_CB_BGM_MUTEX Zone 7 BGM Select & Level	DSP Mutex Block
224	POBJ_Z7_CB_MASTER_LEVEL Zone 7 Master Mix Level	DSP Volume LUT Block
225	POBJ_Z7_CB_PAGING_LEVEL Zone 7 Paging Mix Level	DSP Volume LUT Block
242	POBJ_Z8_CB_ML1_LEVEL Zone 8 Mic/Line 1 Mix Level	DSP Volume LUT Block
243	POBJ_Z8_CB_ML2_LEVEL Zone 8 Mic/Line 2 Mix Level	DSP Volume LUT Block
244	POBJ_Z8_CB_ML3_LEVEL Zone 8 Mic/Line 3 Mix Level	DSP Volume LUT Block
245	POBJ_Z8_CB_ML4_LEVEL Zone 8 Mic/Line 4 Mix Level	DSP Volume LUT Block
246	POBJ_Z8_CB_BGM_MUTEX Zone 8 BGM Select & Level	DSP Mutex Block
247	POBJ_Z8_CB_MASTER_LEVEL Zone 8 Master Mix Level	DSP Volume LUT Block
248	POBJ_Z8_CB_PAGING_LEVEL Zone 8 Paging Mix Level	DSP Volume LUT Block

2.3.1 Object Data: DSP Volume LUT Block

This object is used to set the volume (or level) of an audio node in 0.5dB steps. It includes a mute flag option. The C structure is defined as below and is 2 bytes in length:

```
typedef struct
{
 uint8 P_VolumeLutIndex;
 uint8 P_Flags;
} DspVolumeLutBlock;
```

The first byte, P_VolumeLutIndex, is a value ranging from 0 to 249. 0 indicates mute, and values 1-249 represent -100.0dB to +24.0dB in 0.5dB steps. The following table shows the arrangement:

Volume LUT Index	Audio dB Value
0	Mute
1	-100.0dB
2	-99.5dB
3	-99.0dB
...	...
177	-12.0dB
...	...
189	-6.0dB
...	...
201	0.0dB
...	...
248	23.5dB
249	24.0dB

The second byte, P_Flags, can be used to mute the volume block without changing the volume value stored in the LUT index (this allows a mute button in a GUI to simply change this field, and a fader control to change the LUT index field).
P_Flags = 0x00 = Unmuted
P_Flags = 0x01 = Muted

2.3.2 Object Data: DSP Mutex Block

This object is used to set the mutually exclusive BGM audio source as well as its volume (level). It includes a mute flag option. The C structure is defined as below and is 3 bytes in length:

```
typedef struct
{
 uint8 P_VolumeLutIndex;
 uint8 P_Flags;
 uint8 P_MutexValue;
}DspMutexBlock;
```

The first two fields (P_VolumeLutIndex and P_Flags) are defined identically as in the DSP Volume LUT Block. The third field, P_MutexValue, is defined as follows:
P_MutexValue = 0x00 = No source selected
P_MutexValue = 0x01 = BGM Source 1 selected
P_MutexValue = 0x02 = BGM Source 2 selected
P_MutexValue = 0x03 = BGM Source 3 selected

2.4 Sync Packets

The SYNC packet is used by applications to ensure that parameters stored in the hardware can be synchronized across multiple masters all connected to the same hardware. SYNC packets can only be used to retrieve the state of data in the hardware - not set the data. When connecting to device hardware, master applications can request these four packets which should contain the state of all relevant parameters for the application.
For external master applications, the matrix processor has four specific types of SYNC packet that can be requested. These four packets contain all the information required to be able to display channel names and current zone control preset objects (from the live preset only). Data received back from the hardware in these packets represents the latest known parameter states, and connected master applications should update themselves to reflect this data.
It should be noted that if these packets are requested at a rate of 500msec, then it would take 2 seconds for the device to detect and display changes invoked into the hardware from other master applications. As it is likely that SYNC packet type 103 (which contains the audio settings) would change the most, this packet could be requested in between all other types leading to a 1 second update time.
To avoid race conditions with GUI controls and incoming SYNC data, it is highly recommended that SYNC information received for parameters changed within the last two seconds on the master application are ignored. This prevents controls from flickering to different unexpected values to a user. It is also recommended that if a control has focus on a GUI, that it is not updated with a SYNC value until it loses focus to prevent confusing value changes that may occur in the middle of a value edit.

Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	ID of the SYNC return packet: 100 = First type of SYNC packet 101 = Second type of SYNC packet 102 = Third type of SYNC packet 103 = Fourth type of SYNC packet

The four SYNC packets to receive have request numbers of 100, 101, 102 and 103.
For a SYNC packet of type 100, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	100
Hardware App State	1 byte	0x00 = Normal Mode 0x01 = Standby Mode 0x02 = Playing Global Alert Tone 0x03 = Playing Global Evac Tone 0x04 = Override Mode
Reserved 1	1 byte	This byte can be ignored
Reserved 2	1 byte	This byte can be ignored
Reserved 3	1 byte	This byte can be ignored
Mic/Line 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Mic/Line 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Mic/Line 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Mic/Line 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
BGM 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
BGM 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
BGM 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Standby Allowed	1 byte	0x00 = Standby mode is not allowed 0x01 = Standby mode is allowed
Force into Standby	1 byte	0x00 = Do not force into standby mode 0x01 = Force standby flag is set in hardware
Global Mute Allowed	1 byte	0x00 = Global mute is not allowed 0x01 = Global mute is allowed

Packet Field	Size	Description
Force Global Mute	1 byte	0x00 = Do not force into global mute 0x01 = Force global mute is set in hardware

For a SYNC packet of type 101, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	101
Hardware App State	1 byte	0x00 = Normal Mode 0x01 = Standby Mode 0x02 = Playing Global Alert Tone 0x03 = Playing Global Evac Tone 0x04 = Override Mode
Reserved 1	1 byte	This byte can be ignored
Reserved 2	1 byte	This byte can be ignored
Reserved 3	1 byte	This byte can be ignored
Zone 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 5 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 6 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Zone 7 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

Packet Field	Size	Description
Zone 8 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

For a SYNC packet of type 102, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	102
Hardware App State	1 byte	0x00 = Normal Mode 0x01 = Standby Mode 0x02 = Playing Global Alert Tone 0x03 = Playing Global Evac Tone 0x04 = Override Mode
Reserved 1	1 byte	This byte can be ignored
Reserved 2	1 byte	This byte can be ignored
Reserved 3	1 byte	This byte can be ignored
Preset 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 5 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 1 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 2 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user

Packet Field	Size	Description
Preset 3 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 4 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 5 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user

For a SYNC packet of type 103, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	103
Hardware App State	1 byte	0x00 = Normal Mode 0x01 = Standby Mode 0x02 = Playing Global Alert Tone 0x03 = Playing Global Evac Tone 0x04 = Override Mode
Reserved 1	1 byte	This byte can be ignored
Reserved 2	1 byte	This byte can be ignored
Reserved 3	1 byte	This byte can be ignored
Zone 1 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 1 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 1 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 1 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 1 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 1 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 1 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)

Packet Field	Size	Description
Zone 2 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 2 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 2 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 2 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 2 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 2 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 2 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 3 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 3 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 4 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 4 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)

Packet Field	Size	Description
Zone 5 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 5 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 5 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 5 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 5 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 5 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 5 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 6 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 6 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 7 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 7 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 7 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 7 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 7 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 7 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)

Packet Field	Size	Description
Zone 7 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 Mic/Line 1 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 Mic/Line 2 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 Mic/Line 3 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 Mic/Line 4 Mix Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 BGM Select and Level	3 bytes	DSP Mutex Block (see POBJ section)
Zone 8 Master Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Zone 8 Paging Level	2 bytes	DSP Volume LUT Block (see POBJ section)

3 PLM-4Px2x DSP Amplifier Specific API

This chapter summarizes parameters, commands and packets specific to the PLENA Matrix PLM-4Px2x DSP amplifier hardware. These include preset objects, global objects, signal monitoring and synchronization control aspects.

Revision Control

Date	Notes
5th December 2012	Initial Release
8th February 2013	Added checksum byte reference in POBJ Write packet structure
21st February 2013	SMON packet altered to include sending IP address to data payload GOBJ_GLOBAL_MUTE_ALLOWED global object number changed GOBJ_FORCE_GLOBAL_MUTE global object number changed SYNC packet changes - now three SYNC packet types, with different data layout Firmware release at v1.0.0
10th March 2013	Changed all Unicode strings to UTF-8 strings

3.1 Signal Monitoring

Signal monitoring data enables applications to display the current value of audio signal levels at various signal nodes in the hardware. This data is represented in 32-bit unsigned integers that can be converted into a double type value via the following C code:

```
double ConvertToMonitoring(UInt32 rawValue)
{
  if (rawValue > 0x0007FFFF)
  {
        return 0.0;
  }
  else
  {
        double frac = (double)rawValue / (double)0x0007FFFF;
        double db = 20.0 * Math.Log10(frac);
        if (db < -100.0)
              return -100.0;
        }
        else
        {
              return db;
        }
  }
```

}

In audio terms, returning 0.0dB indicates full-scale audio (this level would result in analog audio in/out being clipped). The scale is negative, with lower values indicating lower audio levels. Below -50.0dB this monitoring starts to become inaccurate, so use of monitoring data should be restricted between 0.0 and -50.0dB.
Consult the master API document for details on how applications start requesting SMON packets from device hardware.
The amplifier SMON packet that gets sent to master applications is structured as per the table below. There are 8 monitoring signal values contained and the IP address of the hardware that is sending the packet in reply.

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SMON (ASCII 0x534D4F4E)
Monitoring Data Index 0	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Monitoring Data Index 1	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Monitoring Data Index 2	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
...	...	...
Monitoring Data Index 7	4 bytes	Monitoring data 32-bit unsigned integer, big-endian
Sender IP Address	4 bytes	The 4 byte IP address of the hardware sending this packet back to the master

Of the 8 monitoring indexes contained in the amplifier SMON packet, the following indexes are the ones useful to applications:

Monitoring Index	Description
0	Amplifier Channel 1 Output
1	Amplifier Channel 2 Output
2	Amplifier Channel 3 Output
3	Amplifier Channel 4 Output

3.2 Global Objects

The GOBJ packet is used to set or get the state of a system global object.
For 3rd-party applications, these packets should be used to control individual user parameters. Using the GOBJ packet to get values of parameters is not recommended as the SYNC packet contains more information and is faster for the same purpose.

The structure of a write (set) GOBJ packet is:

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	GOBJ (ASCII 0x474F424A)
Is Read Flag	1 byte	0x00 (write)
Global Object ID	2 bytes	16-bit number as defined in table below
NV Commit Flag	1 byte	0x00 (do not commit to device NV memory) 0x01 (commit to NV memory)
Object Data	<x> bytes	Data for the object - length depends on the data object

As NV memory has a limited lifespan, if an object is going to be toggled often by an external application, care should be taken to ensure the NV commit is performed once the object state has settled.
The table below lists global objects specifically designed to be accessed via 3rd-party control applications:

Global Object Number	Global Object Name	Data Description (Object Type)
1	GOBJ_FORCE_STANDBY	Boolean Block
15	GOBJ_GLOBAL_MUTE_ALLOWED	Boolean Block
16	GOBJ_FORCE_GLOBAL_MUTE	Boolean Block

GOBJ_FORCE_STANDBY is the global object to use to toggle the hardware into and out of standby mode. In regards to the GLOBAL_MUTE global object, if GOBJ_GLOBAL_MUTE_ALLOWED is 0x00, then the object state of GOBJ_FORCE_GLOBAL_MUTE will be ignored (as the hardware is not allowed to engage the global mute). GOBJ_FORCE_GLOBAL_MUTE is the global object to use to toggle the hardware into and out of global mute.
The C structure for the Boolean block is defined as below and is 2 bytes in length:

```
typedef struct
{
 uint8 P_BooleanValue;
 uint8 P_CheckField;
}BooleanBlock;
```

P_BooleanValue should be set to either 0x00 (false) or 0x01 (true). The value of P_CheckField is used as a verification for P_BooleanValue.
If P_BooleanValue = 0x00, set P_CheckField to 0x64
If P_BooleanValue = 0x01, set P_CheckField to 0x3A

3.3 Preset Objects

The POBJ packet is used to set or get the state of a preset object (usually in the live preset, but it can be used to do the same with stored presets as well).
For 3rd-party applications, these packets should be used to control individual user parameters. Using the POBJ packet to get values of parameters is not recommended as the SYNC packet contains more information and is faster for the same purpose.
The structure of a write (set) POBJ packet is:

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	POBJ (ASCII 0x504F424A)
Is Read Flag	1 byte	0x00 (write)
Preset Number	1 byte	0 = Live Preset
Preset Object ID	2 bytes	16-bit number as defined in table below
NV Commit Flag	1 byte	0x00 (see comment below)
Object Data	<x> bytes	Data for the object - length depends on the data object, e.g. Object Data: DSP Volume Lut Block Object Data: DSP Mutex Block
Object Checksum	1 byte	Set to 0x00 (not required as NV storage is managed by hardware)

Notice!
For all the preset objects accessible by external applications, the hardware automatically manages NV memory storage, so the NV Commit Flag should be set to 0x00.

The table below lists preset objects specifically designed to be accessed via 3rd-party control applications:

Preset Object Number	Preset Object Name	Data Description (DSP Block Type)
1	POBJ_AMPCH1_DSP_BASSENHANCE Amp Channel 1 Bass Enhance	DSP Bass Enhance Block
20	POBJ_AMPCH1_CB_MIXINPUT1 Amp Channel 1 Mix Input 1 Level	DSP Volume LUT Block
21	POBJ_AMPCH1_CB_MIXINPUT2 Amp Channel 1 Mix Input 2 Level	DSP Volume LUT Block
22	POBJ_AMPCH1_CB_MIXINPUT3 Amp Channel 1 Mix Input 3 Level	DSP Volume LUT Block

Preset Object Number	Preset Object Name	Data Description (DSP Block Type)
23	POBJ_AMPCH1_CB_MIXINPUT4 Amp Channel 1 Mix Input 4 Level	DSP Volume LUT Block
26	POBJ_AMPCH1_CB_OUTPUTLEVEL Amp Channel 1 Main Output Level	DSP Volume LUT Block
27	POBJ_AMPCH2_DSP_BASSENHANCE Amp Channel 2 Bass Enhance	DSP Bass Enhance Block
46	POBJ_AMPCH2_CB_MIXINPUT1 Amp Channel 2 Mix Input 1 Level	DSP Volume LUT Block
47	POBJ_AMPCH2_CB_MIXINPUT2 Amp Channel 2 Mix Input 2 Level	DSP Volume LUT Block
48	POBJ_AMPCH2_CB_MIXINPUT3 Amp Channel 2 Mix Input 3 Level	DSP Volume LUT Block
49	POBJ_AMPCH2_CB_MIXINPUT4 Amp Channel 2 Mix Input 4 Level	DSP Volume LUT Block
52	POBJ_AMPCH2_CB_OUTPUTLEVEL Amp Channel 2 Main Output Level	DSP Volume LUT Block
53	POBJ_AMPCH3_DSP_BASSENHANCE Amp Channel 3 Bass Enhance	DSP Bass Enhance Block
72	POBJ_AMPCH3_CB_MIXINPUT1 Amp Channel 3 Mix Input 1 Level	DSP Volume LUT Block
73	POBJ_AMPCH3_CB_MIXINPUT2 Amp Channel 3 Mix Input 2 Level	DSP Volume LUT Block
74	POBJ_AMPCH3_CB_MIXINPUT3 Amp Channel 3 Mix Input 3 Level	DSP Volume LUT Block
75	POBJ_AMPCH3_CB_MIXINPUT4 Amp Channel 3 Mix Input 4 Level	DSP Volume LUT Block
78	POBJ_AMPCH3_CB_OUTPUTLEVEL Amp Channel 3 Main Output Level	DSP Volume LUT Block
79	POBJ_AMPCH4_DSP_BASSENHANCE Amp Channel 4 Bass Enhance	DSP Bass Enhance Block
98	POBJ_AMPCH4_CB_MIXINPUT1 Amp Channel 4 Mix Input 1 Level	DSP Volume LUT Block
99	POBJ_AMPCH4_CB_MIXINPUT2 Amp Channel 4 Mix Input 2 Level	DSP Volume LUT Block

Preset Object Number	Preset Object Name	Data Description (DSP Block Type)
100	POBJ_AMPCH4_CB_MIXINPUT3 Amp Channel 4 Mix Input 3 Level	DSP Volume LUT Block
101	POBJ_AMPCH4_CB_MIXINPUT4 Amp Channel 4 Mix Input 4 Level	DSP Volume LUT Block
104	POBJ_AMPCH4_CB_OUTPUTLEVEL Amp Channel 4 Main Output Level	DSP Volume LUT Block

3.3.1 Object Data: DSP Volume LUT Block

This object is used to set the volume (or level) of an audio node in 0.5dB steps. It includes a mute flag option. The C structure is defined as below and is 2 bytes in length:

```
typedef struct
{
 uint8 P_VolumeLutIndex;
 uint8 P_Flags;
}DspVolumeLutBlock;
```

The first byte, P_VolumeLutIndex, is a value ranging from 0 to 249. 0 indicates mute, and values 1-249 represent -100.0dB to +24.0dB in 0.5dB steps. The following table shows the arrangement:

Volume LUT Index	Audio dB Value
0	Mute
1	-100.0dB
2	-99.5dB
3	-99.0dB
...	...
177	-12.0dB
...	...
189	-6.0dB
...	...
201	0.0dB
...	...
248	23.5dB
249	24.0dB

The second byte, P_Flags, can be used to mute the volume block without changing the volume value stored in the LUT index (this allows a mute button in a GUI to simply change this field, and a fader control to change the LUT index field).
P_Flags = 0x00 = Unmuted

P_Flags = 0x01 = Muted

3.3.2 Object Data: DSP Bass Enhance Block

This object is used to enable or disable the dynamic bass enhancement of the amplifier channel. The C structure is defined as below and is 1 byte in length:

```
typedef struct
{
 uint8 P_Enabled;
}DspBassEnhanceBlock;
```

P_Enabled = 0x00 = Bass enhancement turned off
P_Enabled = 0x01 = Bass enhancement active

3.4 Sync Packets

The SYNC packet is used by applications to ensure that parameters stored in the hardware can be synchronized across multiple masters all connected to the same hardware. SYNC packets can only be used to retrieve the state of data in the hardware - not set the data. When connecting to device hardware, master applications can request these three packets which should contain the state of all relevant parameters for the application.
For external master applications, the amplifier has three specific types of SYNC packet that can be requested. These three packets contain all the information required to be able to display channel names and current amplifier channel control preset objects (from the live preset only). Data received back from the hardware in these packets represents the latest known parameter states, and connected master applications should update themselves to reflect this data.
It should be noted that if these packets are requested at a rate of 500msec, then it would take 1.5 seconds for the device to detect and display changes invoked into the hardware from other master applications. As it is likely that SYNC packet type 102 (which contains the audio settings) would change the most, this packet could be requested in between all other types leading to a 1 second update time.
To avoid race conditions with GUI controls and incoming SYNC data, it is highly recommended that SYNC information received for parameters changed within the last two seconds on the master application are ignored. This prevents controls from flickering to different unexpected values to a user. It is also recommended that if a control has focus on a GUI, that it is not updated with a SYNC value until it loses focus to prevent confusing value changes that may occur in the middle of a value edit.

Packet Structure (master sent):

Packet Field	Size	Description
SE Master Header	10 bytes	Standard Secure Electronics UDP header (sent from master)
Command	4 bytes	SYNC (ASCII 0x53594E43)

Packet Field	Size	Description
Sync Request	1 byte	ID of the SYNC return packet: 100 = First type of SYNC packet 101 = Second type of SYNC packet 102 = Third type of SYNC packet

The three SYNC packets to receive have request numbers of 100, 101 and 102.
For a SYNC packet of type 100, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	100
Amplifier Fault Summary	1 byte	Bit 0 = Amp Channel 1 Thermal Fault (if bit is set) Bit 1 = Amp Channel 1 Shutdown Fault (if bit is set) Bit 2 = Amp Channel 2 Thermal Fault (if bit is set) Bit 3 = Amp Channel 2 Shutdown Fault (if bit is set) Bit 4 = Amp Channel 3 Thermal Fault (if bit is set) Bit 5 = Amp Channel 3 Shutdown Fault (if bit is set) Bit 6 = Amp Channel 4 Thermal Fault (if bit is set) Bit 7 = Amp Channel 4 Shutdown Fault (if bit is set)
Override State	1 byte	0x00 if not in override mode 0x01 if in override mode
Standby State	1 byte	0x00 if not in standby mode 0x01 if in standby mode
Input 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Input 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Input 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Input 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Output 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

Packet Field	Size	Description
Output 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Output 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Output 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Force into Standby	1 byte	0x00 = Do not force into standby mode 0x01 = Force standby flag is set in hardware
Global Mute Allowed	1 byte	0x00 = Global mute is not allowed 0x01 = Global mute is allowed
Force Global Mute	1 byte	0x00 = Do not force into global mute 0x01 = Force global mute is set in hardware

For a SYNC packet of type 101, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	101
Amplifier Fault Summary	1 byte	Bit 0 = Amp Channel 1 Thermal Fault (if bit is set) Bit 1 = Amp Channel 1 Shutdown Fault (if bit is set) Bit 2 = Amp Channel 2 Thermal Fault (if bit is set) Bit 3 = Amp Channel 2 Shutdown Fault (if bit is set) Bit 4 = Amp Channel 3 Thermal Fault (if bit is set) Bit 5 = Amp Channel 3 Shutdown Fault (if bit is set) Bit 6 = Amp Channel 4 Thermal Fault (if bit is set) Bit 7 = Amp Channel 4 Shutdown Fault (if bit is set)
Override State	1 byte	0x00 if not in override mode 0x01 if in override mode
Standby State	1 byte	0x00 if not in standby mode 0x01 if in standby mode
Preset 1 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00

Packet Field	Size	Description
Preset 2 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 3 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 4 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 5 Name	31 bytes	31 byte length UTF-8 encoded string. Strings not requiring the full length are padded with bytes set to 0x00
Preset 1 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 2 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 3 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 4 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user
Preset 5 In Use	1 byte	0x00 = Preset not used 0x01 = Preset is valid, and can be recalled by the user

For a SYNC packet of type 102, the following return packet from the slave will be received:

Packet Field	Size	Description
SE Slave Header	10 bytes	Standard Secure Electronics UDP header (sent from slave)
Command	4 bytes	SYNC (ASCII 0x53594E43)
Sync Request	1 byte	102
Amplifier Fault Summary	1 byte	Bit 0 = Amp Channel 1 Thermal Fault (if bit is set) Bit 1 = Amp Channel 1 Shutdown Fault (if bit is set) Bit 2 = Amp Channel 2 Thermal Fault (if bit is set) Bit 3 = Amp Channel 2 Shutdown Fault (if bit is set) Bit 4 = Amp Channel 3 Thermal Fault (if bit is set) Bit 5 = Amp Channel 3 Shutdown Fault (if bit is set) Bit 6 = Amp Channel 4 Thermal Fault (if bit is set) Bit 7 = Amp Channel 4 Shutdown Fault (if bit is set)

Packet Field	Size	Description
Override State	1 byte	0x00 if not in override mode 0x01 if in override mode
Standby State	1 byte	0x00 if not in standby mode 0x01 if in standby mode
Amp Channel 1 Bass Enhance	1 byte	DSP Bass Enhance Block (see POBJ section)
Amp Channel 1 Mix Input 1	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 1 Mix Input 2	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 1 Mix Input 3	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 1 Mix Input 4	2 bytes	DSP Volume LUT Block (see POBJ section)
Reserved	4 bytes	
Amp Channel 1 Main Output Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 2 Bass Enhance	1 byte	DSP Bass Enhance Block (see POBJ section)
Amp Channel 2 Mix Input 1	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 2 Mix Input 2	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 2 Mix Input 3	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 2 Mix Input 4	2 bytes	DSP Volume LUT Block (see POBJ section)
Reserved	4 bytes	
Amp Channel 2 Main Output Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 3 Bass Enhance	1 byte	DSP Bass Enhance Block (see POBJ section)
Amp Channel 3 Mix Input 1	2 bytes	DSP Volume LUT Block (see POBJ section)

Packet Field	Size	Description
Amp Channel 3 Mix Input 2	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 3 Mix Input 3	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 3 Mix Input 4	2 bytes	DSP Volume LUT Block (see POBJ section)
Reserved	4 bytes	
Amp Channel 3 Main Output Level	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 4 Bass Enhance	1 byte	DSP Bass Enhance Block (see POBJ section)
Amp Channel 4 Mix Input 1	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 4 Mix Input 2	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 4 Mix Input 3	2 bytes	DSP Volume LUT Block (see POBJ section)
Amp Channel 4 Mix Input 4	2 bytes	DSP Volume LUT Block (see POBJ section)
Reserved	4 bytes	
Amp Channel 4 Main Output Level	2 bytes	DSP Volume LUT Block (see POBJ section)

Bosch Security Systems B.V.
Torenallee 49
5617 BA Eindhoven
Netherlands
www.boschsecurity.com
© Bosch Security Systems B.V., 2018