24 Series

Serial Control Protocol

CLOUD ELECTRONICS LIMITED

140 Staniforth Road,
Sheffield S9 3HF,
England
Phone +44 (0)114 244 7051
Fax +44 (0)114 242 5462

E-mail technical@cloud.co.uk

Website www.cloud.co.uk

UPDATES

V1.0 – First revision

INTRODUCTION

This document describes the serial protocol used to communicate with the 24-120 & 24-240 mixer-amplifiers.

As the serial command protocol is the same for the two products, throughout the rest of the document the two products will be referred to as the 24 Series.

Details for configuring the Serial Interface are in the 24 Series installation guide.

Throughout this document all message text is shown in a `mono spaced font`.

MESSAGE STRUCTURE

There are three types of messages in the 24 Series control protocol:-

- Control messages.
- Response messages.
- Error messages.

Control messages are sent to the 24 Series. Response and error messages are transmitted from the 24 Series in reply to a control message.

All control messages use upper case letters; all response messages use lower case letters.

The message body is enclosed within a header character (“<”) and a terminator sequence (“/>”). The 24 Series will reset its message decoding software upon reception of a header character (“<”), discarding any previously un-decoded partial messages. The 24 Series will start to decode a message upon reception of the terminator sequence (“/>”). If the decoded message is valid it is executed and a response message returned. An error message is returned if the command message has been interrupted by a new command message, if it is too large, if it cannot be decoded or if the command cannot be executed.

THE CONTROL MESSAGE

Control messages are sent to the 24 Series in order to perform a function. The control message has two fields separated by a comma. They are called the destination field and the command field.

`<DESTINATION,COMMAND/>`

DESTINATION FIELD

The destination field is an abbreviated description of the mixer section the command is intended to affect. The field consists of an optional single character **default modifier**, the two character **main destination** and an optional two character **sub destination**. The sub destination is separated from the main destination by a full stop (period).

- **DEFAULT MODIFIER.** The character "D" is used to indicate the message is set to a default value.

- **MAIN DESTINATION.** The 24 Series has four possible main destinations;
  1. The zones.
  2. The microphones.
  3. The global music channel.
  4. The system.

- **SUB DESTINATION.** The sub destination is used to address the message to a particular part of a main destination. Only the zone and system destinations have sub destinations.

Examples

1. `<Z1.MU,L23/>` MU is the sub destination of music in the main destination of zone 1
2. `<DZ1.MU,S3/>` Default modifier applied. MU is the sub destination of music in the main destination of zone 1
3. `<Z1.M1,M/>` M1 is the sub destination of microphone 1 in zone 1

COMMAND FIELD

The command field identifies the command to be performed on the destination. The command field may have up to three parts; the **command identifier**, the **command modifier** and the **command value**. No white space is permitted between these parts.

- **COMMAND IDENTIFIER**. This part is always required. It has a single alphabetic character to identify the command type.

- **COMMAND MODIFIER.** This part is optional depending on the command type. Some commands take no modifier, some commands always take a modifier and some commands can have the option of a modifier or none. The modifier is a single alphabetic character.

- **COMMAND VALUE**. This part is optional depending on the command type and modifier. Source and level commands require a decimal numeric value. The text field command takes up to 32 characters of text.

Examples

1. `<Z1.MU,M/>` The command identifier is "M".

2. `<Z2.MU,SD/>` The command identifier is "S", the command modifier is "D".

3. `<Z3.MU,L22/>` The command identifier is "L", the command value is "22"

THE RESPONSE MESSAGE

The response message is the 24 Series reply to a valid message. Response messages are always in lower case. For mute, open and commands sent to the system destination the response messages are a lower case version of the message sent. The response message confirms the new state. For level and source commands the response message confirms the new value set; the response message is always a lower-case message of the absolute value resulting from the command.
The response message from the global music destination has a list of the value in each zone separated by a semicolon.

Examples

1. `<Z1.MU,L12/>` Set the music level in zone 1 to 12.
   `<z1.mu,l12/>` Response returns the new value of level in zone 1.

2. `<Z1.MU,LU3/>` Increment the music level in zone 1 by 3
   `<z1.mu,l9/>` Response returns the new value of level in zone 1.

3. `<MU,LQ/>` Query level in global music (All zones)
   `<mu,l12;45;9;22/>` Return levels in semicolon separated list

THE ERROR MESSAGE

The error message is the 24 Series reply to an invalid message. The error message can have three fields separated by a space, the **ERROR IDENTIFIER**, the **RETURNED MESSAGE,** and the **ERROR TEXT**.

- **ERROR IDENTIFIER**. The error identifier has two characters; the first is an exclamation mark (“!”), the second is an upper case alphabetic character to identify the type of error.
- **RETURNED MESSAGE.** Not all error types use a returned message. When present the returned message is a copy of the sent message, either upper or lower case depending on how far the message has been decoded before the error is detected.
- **ERROR TEXT.** This is a brief description of the error.

ERRORS IN DETAIL

There are seven error types.

BUFFER FULL ERROR

This error is generated when a message has overflowed the 24 Series receive buffer. It is identified by the sequence (“!B”). It is used to indicate that the current message has been abandoned. It has no return message.

The receive buffer in the 24 Series is 64 characters long.

Example

```
<Z1.MU,L77777777777777777777777777777777777777777777777777
7<!B Message Buffer Full/>7777<Z1.MU,L8/><z1.mu,l8/>
```

NB All subsequent characters received by the 24 Series are ignored until it receives a header character (“<”) to start a new message.

EXECUTION ERROR

This error is generated if the execution function for the command fails. It is identified by the sequence (“!E”).

Example

```
<SY.RS,B38967/><!E SY.RS,B38967 Execution error/>
```

NB 38967 is not a valid Baud rate.

INTERRUPTED ERROR

This error is generated when a header character (“<”) is detected in the middle of a message. It is identified by the sequence (“!I”). It is used to indicate that the interrupted message has been abandoned and that the 24 Series has started to receive a new message. An interruption is detected when the opening angle bracket of the new message is received; the Interrupted Error message is transmitted immediately.

Example

```
<Z1.MU<<!I Message Interrupted/>Z2.MU,L23/><z2.mu,l23/>
```

NB the interrrupting message is preserved and processed normally.

NVM NOT READY ERROR

This error is generated if a control message is sent before the start up parameters have been read from Non Volatile Memory. It is identified by the sequence (“!N NVM not ready, please wait”). This error will clear shortly after power on; it is provided to acknowledge a message has been received but not acted upon.

OVERRUN ERROR

This error is identified by the sequence (“!A”). This indicates a fault condition within the software and should not be encountered during normal operation.

PARSE ERROR

This error is generated when a message is correctly formatted but the destination or command field cannot be decoded as valid. It is identified by the sequence (“!P”). Its return message is an upper case copy of the sent message.

Example

<Z1.MY,L8/><!P Z1.MY,L8 Parse error/>

NB “MY” is not a valid sub destination.

TOKEN ERROR

This error is generated when there is a problem with the format of the message so that it cannot be decoded into its destination and command tokens. It is identified by the sequence (“T”). Its return message is an upper case copy of the sent message up to the point of the detected error.

Example

<Z1;MU,L9/><!T Z1; Token error/>

NB The semicolon is not a valid sub destination separator.

TABLE OF ERRORS

Error Type	ID	Fault Condition	Example Message	
			Faulty Message	Response
Buffer Full	B	Message too long.	<GGGGGGGGG...	<!B Message Buffer Full/>
Execution	E	The execute function fails.	<SY.RS,B2345/>	<!E SY.RS,B2345 Execution error/>
Interrupted	I	Message interrupted before complete.	<Z1.MU<	<!I Message Interrupted/>
NVM	N	Waiting to finish reading Non Volatile Memory.	Any	<(“!N NVM not ready, please wait/>
Overrun	A	Internal fault.	Any	<!A/>
Parser	P	Error decoding message.	<Z1.MU,K89/>	<!P Z1.MU,K89 Parse error/>
Token	T	Error in message format.	<Z1M/>	<!T Z1M Token error/>

TABLE OF COMMANDS

Command		Identifier	Example Message			
			Zone	Mic	Global Music	Global Mic
Mute		M	<Z1.MU,M/> <Z1.M1,M/>	<M1,M/>	<MU,M/>	<MI,M/>
Open		O	<Z3.MU,O/> <Z2.M1,O/>	<M1,O/>	<MU,O/>	<MI,O/>
Query		Q	<Z1.MU,Q/> <Z1.M1,Q/>	<M1,Q/>	<MU,Q/>	<MI,Q/>
Source Select	Up	SU	<Z1.MU,SU/>	**NA**	<MU,SU/>	**NA**
	Down	SD	<Z2.MU,SD/>		<MU,SD/>	
	Absolute	S	<Z1.MU,S5/>		<MU,S3/>	
	Enable	SE	<Z1.MU,SE/>		<MU,SE/>	
	Disable	SX	<Z1.MU,SX/>		<MU,SX/>	
	Query	SQ	<Z1.MU,SQ/>		<MU,Q/>	
Level	Up	LU	<Z1.MU,LU10/>	<Z1.M1,LU10/>	<MU,LU10/>	<MI,LU10/>
	Down	LD	<Z2.MU,LD8/>	<Z1.M1,LD8/>	<MU,LD10/>	<MI,LD10/>
	Absolute	L	<Z1.MU,L22>	<Z1.M1,L22/>	<MU,L32/>	<MI,L32/>
	Enable	LE	<Z1.MU,LE/>	<Z1.M1,LE/>	<MU,LE/>	<MI,LE/>
	Disable	LX	<Z1.MU,LX/>	<Z1.M1,LX/>	<MU,LX/>	<MI,LX/>
	Query	LQ	<Z1.MU,LQ/>	<Z1.M1,LQ/>	<MU,LQ/>	<MI,LQ/>
Default Commands						
Level		D[dst],L	<DZ2.MU,L10/>	**NA**	<DMU,L32/>	
Source		D[dst],S	<DZ1.MU,S4/>		<DMU,S4/>	
Mute		D[dst],M	<DZ3.MU,M/> <DZ1.M1,M/>	<DMI,M/>	<DMU,M/>	
Open		D[dst],O	<DZ3.MU,O/> <DZ2.M1,O/>	<DM2,O/>	<DMU,O/>	
Enable	Level	D[dst],LE	<DZ4.MU,LE/>	**NA**	<DMU,LE/>	
	Source	D[dst],SE	<DZ1.MU,SE/>		<DMU,SE/>	
Disable	Level	D[dst],LX	<DZ3.MU,LX/>		<DMU,LX/>	
	Source	D[dst],SX	<DZ3.MU,SX/>		<DMU,SX/>	

TABLE OF COMMANDS CONTINUED…

System Commands			
Command		**Identifier**	
Reset		R	<SY,R/>
Initialisation mode	Previous	IP	<SY,IP/>
	Default	ID	<SY,ID/>
	Factory	IF	<SY,IF/>
RS232	Baud	B	<SY.RS,B9600/>
Password	Set	K	<SY,K12345678/>
PING		?	<SY,?/>
System music mute socket	Query	MQ	<SY,MQ/>
Software version	Query	SV,Q	<SY.SV,Q/>
Hardware version	Query	HV,Q	<SY.HV,Q/>
Parent amplifier power control	Power down	PA,PD	<SY.PA,PD/>
	Power up	PA,PU	<SY.PA,PU/>
	Query	PA,Q	<SY.PA,Q/>
Text Field	Set	TX,S	<SY.TX,S=[Up to 32 characters]/>
	Query	TX.Q	<SY.TX,Q/>
Boot Load	Unlock	BU	<SY,BU1234/>
	Enable	BE	<SY,BE/>
	Reset	BR	<SY,BR/>
	Lock	BL	<SY,BL/>
	Query	BQ	<SY,BQ/>

EXAMPLES

Each example has a typical 24 Series reply in red.

MUSIC COMMANDS

Music Level

Set the music level in a specified zone

```
<Z1.MU,LE/><z1.mu,le/>          (* Zone 1 music level enable.
<Z1.MU,LX/><z1.mu,lx/>          (* Zone 1 music level disable.
<Z1.MU,L12/><z1.mu,l12/>        (* Zone 1 music level to -12dB.
<Z1.MU,LU7/><z1.mu,l5/>         (* Zone 1 music level up by 7dB to  -5dB.
<Z1.MU,LD3/><z1.mu,l8/>         (* Zone 1 music level down by 3dB to -8dB.
<Z2.MU,LQ/><z2.mu,l90/>         (* Zone 2 music level query.
```

Set the music level in both zones

```
<MU,LE/><mu,le/>                (* Global music level enable.
<MU,LX/><mu,lx/>                (* Global music level disable.
<MU,LQ/><mu,l8;90/>             (* Global music query.
<MU,LU3/><mu,l5;87/>            (* Global music up 3dB.
<MU,L12/><mu,l12;12/>           (* Global music  level to -12dB.
```

Music Source

Set the music source in a specified zone

```
<Z1.MU,SE/><z1.mu,se/>          (* Zone 1 source enable.
<Z1.MU,SX/><z1.mu,sx/>          (* Zone 1 source disable.
<Z1.MU,S2/><z1.mu,s2/>          (* Zone 1 source to 2.
<Z1.MU,SD/><z1.mu,s1/>          (* Zone 1 source down.
<Z1.MU,SU/><z1.mu,s2/>          (* Zone 1 source up.
```

Set the music source in both zones

```
<MU,SE/><mu,se/>                (* Global music source enable.
<MU,SX/><mu,sx/>                (* Global music source disable.
<MU,SU/><mu,s3;6/>              (* Global music source up.
<MU,SQ/><mu,s3;6/>              (* Global music source query.
<MU,S2/><mu,s2;2/>              (* Global music source to 2.
```

Mute / Open

Enable Music Mute in a specified zone

```
<Z1.MU,M/><z1.mu,m/>            (* Zone 1 music mute on.
<Z1.MU,O/><z1.mu,o/>            (* Zone 1 music mute off.
```

Enable Music Mute in a both zones

```
<MU,M/><mu,m/>                  (* Global music mute on.
<MU,O/><mu,o/>                  (* Global music mute off.
```

Query

Global music query

```
<MU,Q/><mu,q = o,sx,le;m,sx,le/>   (* Zone 1, unmuted, Source=disabled, Level=enabled
                                   (* Zone 2, muted, Source=disabled, Level=enabled
```

Zone 1 music query

```
<Z1.MU,Q/><z1.mu,q = o,sx,lx/>     (*Zone 1, unmuted, Source=disabled, Level=disabled
```

Defaults

Set default status of product parameters

<DZ1.MU,LE/><dz1.mu,le/> (* Default zone 1 music level enable.
<DZ1.MU,LX/><dz1.mu,lx/> (* Default zone 1 music level disable.
<DZ2.MU,SE/><dz2.mu,se/> (* Default zone 2 music source enable.
<DZ2.MU,SX/><dz2.mu,sx/> (* Default zone 2 music source disable.
<DZ1.MU,S3/><dz1.mu,s3/> (* Default zone 1 music source set to 3.
<DZ2.MU,L12/><dz2.mu,l12/> (* Default zone 2 music level set to -12dB.
<DZ2.MU,M/><dz2.mu,m/> (* Default zone 2 music mute state set to on.
<DZ2.MU,O/><dz2.mu,o/> (* Default zone 2 music mute state set to off.

<DMU,LE/><dmu,le/> (* Default global music level enable.
<DMU,LX/><dmu,lx/> (* Default global music level disable.
<DMU,SE/><dmu,se/> (* Default global music source enable.
<DMU,SX/><dmu,sx/> (* Default global music source disable.
<DMU,L2/><dmu,l2;2/> (* Default global music level set to -2dB.
<DMU,S3/><dmu,s3;3/> (* Default global music source set to 3.
<DMU,M/><dmu,m/> (* Default global music mute state set to on.
<DMU,O/><dmu,o/> (* Default global music mute state set to off.

MICROPHONE COMMANDS

Mute / Open

<MI,M/><mi,m/> (* Global microphone mute on.
<MI,O/><mi,o/> (* Global microphone mute off.
<Z2.M1,M/><z2.m1,m/> (* Zone 2 microphone mute on.
<Z2.M1,O/><z2.m1,o/> (* Zone 2 microphone mute off.

Mic Level

Set the mic level in a specified zone

<Z1.M1,LE/><z1.m1,le/> (* Zone 1 mic level enable.
<Z1.M1,LX/><z1.m1,lx/> (* Zone 1 mic level disable.
<Z1.M1,L12/><z1.m1,l12/> (* Zone 1 mic level to -12dB.
<Z1.M1,LU7/><z1.m1,l5/> (* Zone 1 mic level up by 7dB to -5dB.
<Z1.M1,LD3/><z1.m1,l8/> (* Zone 1 mic level down by 3dB to -8dB.
<Z2.M1,LQ/><z3.m1,l90/> (* Zone 2 mic level query.

Set the mic level in both zones

<MI,LE/><mi,le/> (* Global mic level enable.
<MI,LX/><mi,lx/> (* Global mic level disable.
<MI,LQ/><mi,l8;90/> (* Global mic query.
<MI,LU3/><mi,l5;87/> (* Global mic up 3dB.
<MI,L12/><mi,l12;12/> (* Global mic level to -12dB.

Defaults

Set default status of product parameters

<DZ1.M1,LE/><dz1.m1,le/> (* Default zone 1 mic level enable.
<DZ1.M1,LX/><dz1.m1,lx/> (* Default zone 1 mic level disable.
<DZ1.M1,L40/><dz1.m1,l40/> (* Default zone 1 mic level set to -40dB.
<DZ2.M1,L12/><dz2.m1,l12/> (* Default zone 2 mic level set to -12dB.
<DZ1.M1,M/><dz1.m1,m/> (* Default zone 1 microphone mute on.
<DZ2.M1,O/><dz2.m1,o/> (* Default zone 2 microphone mute off.

<DMI,LE/><dmi,le/> (* Default global mic level enable.
<DMI,LX/><dmi,lx/> (* Default global mic level disable.
<DMI,L2/><dmi,l20;20/> (* Default global mic level set to -20dB.
<DMI,M/><dmi,m/> (* Default microphone mute state set to on.
<DMI,O/><dmi,o/> (* Default microphone mute state set to off.

Query

<Z1.M1,Q/><z1.m1,q = o/> (* Zone 1 microphone query.
<MI,Q/><mi,q = o;o/> (* Global microphones query.

SYSTEM COMMANDS

Initialisation

<SY,ID/><sy,id/> (* Set power up mode to default.
<SY,IP/><sy,ip/> (* Set power up mode to last (previous hence the 'P').
<SY,IF/><sy,if/> (* Set power up mode to factory defaults.

Reset

<SY,R/><sy,r/> (* Reset to factory settings.

Ping

<SY,?/><sy,?/> (* No effect. Trigger a response message.

Baud

<SY.RS,B4800/><sy.rs,b4800/> (* Set RS232 Baud rate to 4800.
<SY.RS,B9600/><sy.rs,b9600/> (* Set RS232 Baud rate to 9600.

Version

<SY.HV,Q/><sy.hv,q = V1.2/> (* Hardware version for 24 Series.
<SY.SV,Q/><sy.sv,q = V1.00/> (* Software version for 24 Series.

Power Control

<SY.PA,PU/><sy.pa,pu/> (* Release amp from forced power down mode.
(* NB amp may still be in auto power down.
<SY.PA,PD/><sy.pa,pd/> (* Force the amp into power down mode.
<SY.PA,Q/><sy.pa,q = pd/> (* Query state of forced power down.

System Boot Load Mode

```
<SY,BU1234/><sy,bu = ud/>              (* Unlock boot loader. NB PIN required.
<SY,BE/><sy,be = ue/>                  (* Enable boot loader.
<SY,BR/><sy,br/>                       (* Boot loader system reset. Boot loader now active.
<SY,BL/><sy,bl = ld/>                  (* Lock  boot loader. NB boot loader is also disabled.
<SY,BD/><sy,bd = ld/>                  (* Disable boot loader.
<SY,BQ/><sy,bq = ld/>                  (* Query boot loader.
<SY,BR/><!E SY,BR Execution error/>    (* System reset with boot loader disabled.
```

Password

```
<SY,K12345678/><sy,k12345678/>         (* Set password. Old password = 1234,
                                       (* new password = 5678.
```

Text Field

```
<SY.TX,S=Label Text/><sy.tx,s=Label Text/>     (* Set text field to “Label Text”
<SY.TX,Q/><sy.tx,q = Label Text/>              (* Query text field.
```

System Music Mute Socket Query

```
<SY,MQ/><sy,mq = o/>                   (* System music mute socket query, mute off.
<SY,MQ/><sy,mq = m/>                   (* System music mute socket query, mute on.
```