X AIR Mixer Series Remote Control Protocol

(firmware 1.11 or higher)

1. Description

The X AIR mixers are using a communication protocol that is compatible to standard OSC with MUSIC Group specific extensions (e.g. parameter enquiry, subscriptions).

OSC packets are received on **UDP port 10024** and replies are sent back to the requester's IP/port.

The corresponding Parameters.txt file contains all relevant details about OSC paths and parameter ranges.

2. Client messages

Operation	OSC address	Parameters	Comments
1.0 info request	/info or /xinfo	None	Server responds with /info or /xinfo message
1.1 status request	/status	None	Server responds with /status message
1.2 remote request	/xremote	None	Triggers server to send all parameter changes to maximum eight active clients. Timeout is 10 seconds.
1.3 set single parameter	<OSC path-formatted parameter name>	<string|int|float|blob value>	Sets the value of a console parameter, if it exists and value is in range.
1.4 get single parameter	<OSC path-formatted parameter name>	None	Requests the value of a console parameter. If it exists, new value is echoed back by server
1.5 subscribe to meter values	/meters	<string id> <int chnmeterid, optional>	Results in regular updates to batch of meter values as a single binary blob, to OSC address <id>. Meter values are signed integer 16 bit, resolution 1/256 dB see description for "*Meter subscriptions*"

Server messages

Operation	OSC address	Parameters	Comments
1.0 info request	/info or /xinfo	<string server_version> (/xinfo: <sting ip_address>) <string device_name> <string device_model> <string device_version>	Name=value pair strings e.g. “device_version=1.2”. First four are defined, and required. Further args are implementation dependent
1.1 status request	/status	<string status> <string ip address> <string server name> ..	Status should be “active” or “standby” depending on whether server is currently able to fulfil subscriptions. IP address of the form “192.168.2.1”. Server name is to uniquely identify this console. Further args are implementation dependent
1.3 set single 1.4 get single parameter	<OSC path-formatted parameter name>	<string|int|float|blob value>	Echoes the value of a parameter in response to a get, set or fulfilment of single parameter subscription, e.g. /ch/01/mix/fader
1.5 subscribe to meter values	<client-specified id>	<blob data>	Blob contains meter data (signed integer 16 bit, resolution 1/256 dB) as single binary blob

Type rules (get/set parameter)

- parameters must be big-endian and 4-byte aligned/padded, as per OSC specification
- float parameters must be in range 0.0 – 1.0
- integer parameters are signed 32-bit values
- boolean parameters will map to OSC integer type
- strings must be null-terminated

Meter subscriptions

meter subscriptions are used to request a set of meter values,e.g.:

/meters ,si “/meters/0” 8 *(meterID ,oscvalues <string> <integer>)*

Binary OSC message:

```
2f 6d 65 74 65 72 73 00 2c 73 69 00 2f 6d 65 74 ; /meters.,si./met
65 72 73 2f 30 00 00 00 00 00 00 08 ; ers/0.......
```

… returns 8 channel meters (pre-fader l/r, gate and comp gain reduction, post-fader l/r meters, gate key, comp key) of channel 9:

List of Meter IDs:

/meters/0 <chnmeterid>

CHANNEL:
8 channel meters (pre-fader l/r, gate and comp gain reduction, post-fader l/r meters, gate key, comp key)
-> returns one integer size value and 8 short signed integer values (16 bit) as single binary blob

/meters/1

ALL CHANNELS:
16 mono + 5x2 fx/aux + 6 bus + 4 fx send (all pre) + 2 st (post) + 2 monitor
-> returns one integer size value and 40 short signed integer values (16 bit) as single binary blob

/meters/2

ALL INPUTS:
16 mic, 2 aux, 18 usb
-> returns one integer size value and 36 short signed integer values (16 bit) as single binary blob

/meters/3

FX METERS:
4 x (2 x input, 10 x fx return, 2 x output)
-> returns one integer size value and 56 short signed integer values (16 bit) as single binary blob

/meters/4

RTA100:
100 bins RTA
-> returns one integer size value and 100 short signed integer values (16 bit) as single binary blob

/meters/5

ALL OUTPUTS:
6 aux, lr, 16 p16, 18 usb, headphones
-> returns one integer size value and 44 short signed integer values (16 bit) as single binary blob

/meters/6

ALL DYN:
16 gate, 16 dyn(ch), 6 dyn(bus), dyn(lr)
-> returns one integer size value and 39 short signed integer values (16 bit) as single binary blob

/meters/7

AUTOMIX GAINS:
16 gains (auto mixer)
-> returns one integer size value and 16 short signed integer values (16 bit) as single binary blob

/meters/8

DCAS:
4 dcas
-> returns one integer size value and 4 short signed integer values (16 bit) as single binary blob

/meters/9

RECORDER:
2 record, 2 play
-> returns one integer size value and 4 short signed integer values (16 bit) as single binary blob

Snapshot operations

- The first "name" or "index" values are put in the temporary buffer and may be used for a following save/load command
- The direct “name” or “scope” paths following "1....64/" are changing the respective snaphot's name/scope immediately (i.e. no save cmd required)
- The recall scope flags are a sequence of 59 characters, in the order that is listed below, and every "+" means "recall on", while "-" or anything else is "off".

/snap/

```
name (31)       // snapshot name for save, recently loaded snapshot name
index [1,64]    // recently loaded/saved snapshot number
load [1,64]     // 1..64 (triggers snapshot load)
save [1,64]     // 1..64 (triggers snapshot save)
delete [1,64]   // 1..64 (triggers snapshot delete)
01..64/
        name (31)     // snapshot name (31 chars max)
        scope (59)    // snapshot recall scope:
                      // ch1..16, aux 17-18, fxreturn1..4
                      // busmaster1..6, fxsendmaster1..4, main lr,
                      // dca1..4, fx1..4, source, input, config, eq, dyn,
                      // bussend1..6, fxsend1..4, fader-pan, mute,
                      // routing, console configuration
                      // (“+” = on, “-“ = off, 59 chars)
```