1 Client.ck

This file is ideally all that needs to be run on a client machine to enable easy communication. Note that it is not necessary to communicate over OSC, but translates MIDI into OSC to be sent over the network. As such there is some comunication required. The necessary elements are as follows.

a) The host, consisting of

   i) 4-byte IP address (eg. `192.168.0.1`) or hostname (eg. `localhost`)

   ii) a port number, which is just an unsigned integer.

b) The port on which you would like the server to send communications back (if different from above). This can be specified by prefixing it with `in=`

c) The clients IP address, formatted as above. Unfortunately the client has to actually know this to enable two-way communication, in ChucK there is no way of getting the address of a received message, so we have to specifically tell the server where we are. To avoid confusion this is specified as `self=`<*address*>.

d) The MIDI port to listen on – this must be prefixed by `midi=` can be a number or a name, if it is a name containing spaces it might be wise to surround it in quotes (eg. `midi="IAC Driver 1 Bus 1"`)

If these need to be set, they can be provided as arguments to `Client.ck`. In ChucK arguments are expected to be colon delimited and appended to the name of the file, so the command to run `Client.ck` would become something like

```
%> chuck + Client.ck:192.0.16.8:8000:in=8001:midi="IAC Driver Bus 1"
```

These can be specified in any order. If you are running the file from the midiAudicle, you can add the argument list in the text field above the main editor.

If any of the above are not specified in the arguments, we will attempt to use default values, which should be the same as the default values for the server.

1.1 Standard Interface

The standard interface for which all instruments must do something consists only of two standard messages: note and control. Refer to per instrument specifications for detail, they may interpret these slightly differently.

Note This message is intended to initiate a note event. The MIDI representation is a standard Note On, a 3 byte message in which the first four bits of the first byte are always binary 1001 and the second four bits are the channel (note on channel 1 is therefore 144 and channel 16 159). The channel determines the instrument and this should be enumerated shortly after startup. The next two bytes represent the data, typically the first byte gives the note and the second some other information, for example the desired volume. This MIDI mesage is translated into an OSC message with the address pattern `/instrument/note,ii` where `instrument` is the name of the instrument and the two integers specified in the type tag are the two midi data bytes, in the same order.

Control This message is intended to supply some kind of control data. The MIDI representation is a Control Change message, again this is 3 bytes, with the first four bits of the first channel in this case 1011 (giving the whole byte a range from 176–191). The next byte typically specifies which control and the third the value, which are defined per-instrument. The OSC representation is `/instrument/control,ii`, exactly as above.

Instruments will define exactly what these do and may define more complicated systems for control if required.