

```

/Scaler OSC patch for MLGI
//4.08.09 - Meason Wiley, Ajay Kapur, Dimitri Diakopoulos

CUI v;
v.initPort(2); //run chuck --probe& in Terminal to verify port number
v.init();

Hid hi;
HidMsg msg;

// which keyboard
0 => int device;
// get from command line
if( me.args() ) me.arg(0) => Std.atoi => device;

// open keyboard (get device number from command line)
if( !hi.openKeyboard( device ) ) me.exit();
//<<< "keyboard " + hi.name() + " ready", "" >>>;

//OSC initialization
"localhost" => string hostname;
10001 => int port;
OscSend xmit;
// aim the transmitter
xmit.setHost( hostname, port );

float pin[6];

2 => int mode;
4000.0 => float maxval;
1/maxval => float onedivmax;

//initializing variables
float secderiv0;
float secderiv1;
float secderiv2;
float secderiv3;
float secderiv4;
float secderiv5;

float sectemp0;
float sectemp1;
float sectemp2;
float sectemp3;
float sectemp4;
float sectemp5;

float deriv0;
float deriv1;
float deriv2;
float deriv3;
float deriv4;
float deriv5;

pin[0] => float temp0;
pin[1] => float temp1;
pin[2] => float temp2;
pin[3] => float temp3;
pin[4] => float temp4;
pin[5] => float temp5;

spork ~ modeselect();

while ( true )
{
    for ( 0 => int y; y < 6; y++){

```

```

    scaler(v.a2d(y)) => pin[y];
}

pin[0] - temp0 => deriv0;
pin[0] => temp0;
deriv0 - sectemp0 => secderiv0;
secderiv0 => sectemp0;
<<<pin[0]>>>; //print pin 1 to confirm OSC transmit

pin[1] - temp1 => deriv1;
pin[1] => temp1;
deriv1 - sectemp1 => secderiv1;
secderiv1 => sectemp1;

pin[2] - temp2 => deriv2;
pin[2] => temp2;
deriv2 - sectemp2 => secderiv2;
secderiv2 => sectemp2;

pin[3] - temp3 => deriv3;
pin[3] => temp3;
deriv3 - sectemp3 => secderiv3;
secderiv3 => sectemp3;

pin[4] - temp4 => deriv4;
pin[4] => temp4;
deriv4 - sectemp4 => secderiv4;
secderiv4 => sectemp4;

pin[5] - temp5 => deriv5;
pin[5] => temp5;
deriv5 - sectemp5 => secderiv5;
secderiv5 => sectemp5;

20::ms => now;

if (mode == 1) //trigger
{
  if (deriv0 > .2)
  {
    sendOSC ("/1trig", 1);
    sendOSC ("/1trig", 0);
    <<< "/1trig" >>>;
    20::ms => now;
  }

  else if (deriv1 > .2)
  {
    sendOSC ("/2trig", 1);
    sendOSC ("/2trig", 0);
    <<< "/2trig" >>>;
    20::ms => now;
  }

  else if (deriv2 > .2)
  {
    sendOSC ("/3trig", 1);
    sendOSC ("/3trig", 0);
    <<< "/3trig" >>>;
    20::ms => now;
  }

  else if (deriv3 > .2)
  {
    sendOSC ("/4trig", 1);
    sendOSC ("/4trig", 0);
  }
}

```

```

    <<< "/4trig" >>>;
    20::ms => now;
}

else if (deriv4 > .2)
{
    sendOSC ("/5trig", 1);
    sendOSC ("/5trig", 0);
    <<< "/5trig" >>>;
    20::ms => now;
}

else if (deriv5 > .2)
{
    sendOSC ("/6trig", 1);
    sendOSC ("/6trig", 0);
    <<< "/6trig" >>>;
    20::ms => now;
}
}

else if (mode == 2)
{
    sendOSC ("/1b", pin[0]);
    sendOSC ("/2b", pin[1]);
    sendOSC ("/3b", pin[2]);
    sendOSC ("/4b", pin[3]);
    sendOSC ("/5b", pin[4]);
    sendOSC ("/6b", pin[5]);
    <<<"/1b">>>;
}

else if (mode == 3)
{
    sendOSC ("/1c", pin[0]);
    sendOSC ("/2c", pin[1]);
    sendOSC ("/3c", pin[2]);
    sendOSC ("/4c", pin[3]);
    sendOSC ("/5c", pin[4]);
    sendOSC ("/6c", pin[5]);
    <<<"/1c">>>;
}
}

fun void modeselect()
{
    while (true)
    {
        // wait on event
        hi => now;

        // get one or more messages
        while( hi.recv( msg ) )
        {
            // check for action type
            if( msg.isButtonDown() )
            {
                if (msg.which == 7)
                {
                    2 => mode;
                }
                else if (msg.which == 22)
                {
                    1 => mode;
                }
                else if (msg.which == 9)

```

```
    {
      3 => mode;
    }
    //<<< "down:", msg.which, "(code)", msg.key, "(usb key)", msg.ascii, "(ascii)", mode>>>;
  }
}
}

fun void sendOSC (string s, float x)
{
  xmit.startMsg( s, "f" );
  x => xmit.addFloat;
}

fun float scaler(int x)
{
  x*ondivmax => float z;
  return(z*z*z*127*.0077);
  //return Math.exp(0.2);
}
```