field and perfect circuit (redux)

a realization of *field and perfect circuit* (2009) where each square in the perfect squared square is assigned a sound for its corresponding duration such that no adjacent squares are assigned the same sound. this eliminates the need of interpreting several paths of the smith diagram that represents the perfect squared square and allows the piece to be performer with fewer players.

a performance with the redux version was premiered with cristian alvear in osorno, chile on november 23rd, 2016. in the performance, a computer program that generated the redux version was used together with a live interpretation of the original version.

the computer code used for the performance is provided on the following page and is available (as of this publication of the score on november 23rd, 2016) at:

htttp://www.github.com/mwinter80/field_and_perfect_circuit

michael winter (november, 2016)

```
(
//November 23, 2016
//Allows user to drop files into respective folders
~squares_samples_folder = thisProcess.nowExecutingPath.dirname +/+ "squares/";
~open_perc_samples_folder = thisProcess.nowExecutingPath.dirname +/+ "perc/open/";
~closed_perc_samples_folder = thisProcess.nowExecutingPath.dirname +/+ "perc/closed/";

//Loads files
~squares_samples = PathName(~squares_samples_folder).files.collect(
    {|file| postln(file); Buffer.read(s, file.fullPath)});
~open_perc_samples = PathName(~open_perc_samples_folder).files.collect(
    {|file| postln(file); Buffer.read(s, file.fullPath)});
~closed_perc_samples = PathName(~closed_perc_samples_folder).files.collect(
    {|file| postln(file); Buffer.read(s, file.fullPath)});

//Representation of perfect squared square where the dimensions of the squares are mapped to seconds
~squares=[[0,33],[0,37],[0,42],[33,37],[33,62],[37,53],[37,62],[42,60],[42,66],[53,60],[53,62],[60,62],[60,66],[60,77],[62,77],[62,112],
[66,77],[66,85],[77,85],[77,112],[85,112]];

/*
This could / should be done programatically. Each number corresponds to one of the files in the squares folder (in this case 12 files) and
the index in the array corresponds to the representation of the perfect squared square above. It is important that no adjacent squares are
assigned the same sample. TODO: program this.
*/
~order = [9, 2, 1, 5, 6, 4, 7, 12, 3, 2, 1, 9, 10, 11, 5, 8, 4, 12, 6, 8, 5];
)

//SynthDef - obvio
(
SynthDef(\circuit1, {
    var squares, trig, unitdur;
    trig = Impulse.kr(0);
    unitdur = 1;
    squares = { |i|
        var start, stop, dur, env1, env2, square, openperc, closedperc;
        start = (~squares[i][0])*unitdur;
        stop = (~squares[i][1])*unitdur;
        dur = (~squares[i][1]-~squares[i][0])*unitdur;
        env1 = EnvGen.kr(Env.sine(dur), TDelay.kr(trig, start)) > 0;
        env2 = EnvGen.kr(Env.sine(dur), TDelay.kr(trig, stop)) > 0;
        square = PlayBuf.ar(1, ~squares_samples[~order[i]-1], 1, TDelay.kr(trig, start), loop:1);
        openperc = PlayBuf.ar(1, ~open_perc_samples[~open_perc_samples.size.rand], 1, TDelay.kr(trig, start));
        closedperc = PlayBuf.ar(1, ~closed_perc_samples[~closed_perc_samples.size.rand], 1, TDelay.kr(trig, stop));
        (env1 * (square  + openperc)) + (env2 * closedperc)
    } ! ~squares.size;
    Out.ar([0,1], Mix.new(squares) * 0.5);
}).send(s);
)

//This starts immediately. Better trigger on a delay so it occurs within the 'field'
~circuitSynth = Synth(\circuit1);
```