|  |
| --- |
| Cornell University, School of Electrical and Computer ENgineering |
| Multiprocessor Drum Synthesis |
| ECE5760 |
|  |
| **Jeremy Blum, Sima Mitra, Jason Wright** |
| **Thursday Lab Section** |

|  |
| --- |
| A parallelized array of nodes approximating points on a drum where the wave equation was evaluated were used to generate a high quality representation of various drum sounds in real time. |

# Introduction

# Design and Testing Methods

Below, we describe each of the key components to the larger system in detail, and how each individual component works

## Inputs and Outputs

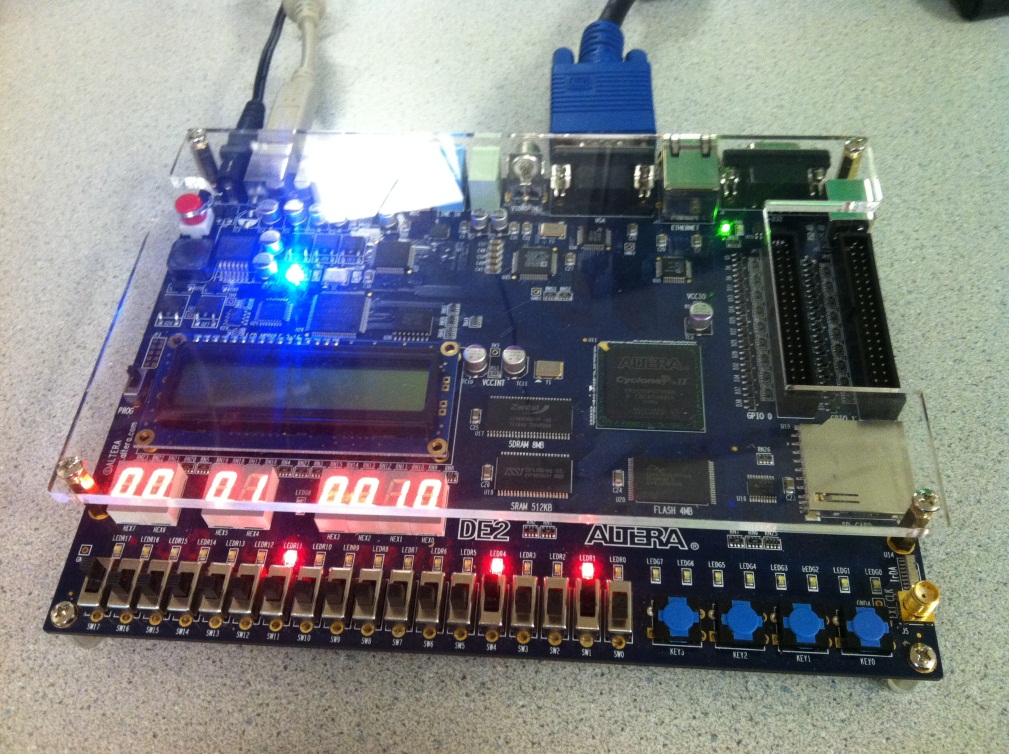


Figure : The DE2 Board that we used for this lab

## High-Level Structure

Figure 2 below shows the basic high-level system architecture:

PUT A PRETTY DIAGRAM HERE

Figure : High-Level System Organization Diagram

We choose to implement a completely parallelized state machine using combinational logic to represent each node on the drum head. Each node is connected together on four sides, and uses information going back in time by two timesteps. The relationships between these nodes are determined by the mathematical considerations that are listed in the next section.

On each clock cycle, all the node states are updated using information from the surrounding nodes, and from the previous two timesteps. In order to increase the total number of nodes in the system, we took advantage of the 4-way symmetry of a square drum-head. Figure 3 shows how the mirroring is setup. Because the wave equation is even about the origin, it’s possible to simulate the sound of a 15x15 node drum using only 8x8 nodes. Note that the full simulation is not 16x16, because the nodes along the origin are counted by both sides of the symmetry. At points along internal symmetry, the neighboring node value is set to the node value of the point on the opposite side of the internal boundary. Along the outer boundaries, all nodes are fixed to simulate the effect of a drum head being stretched over the edges of the drum.

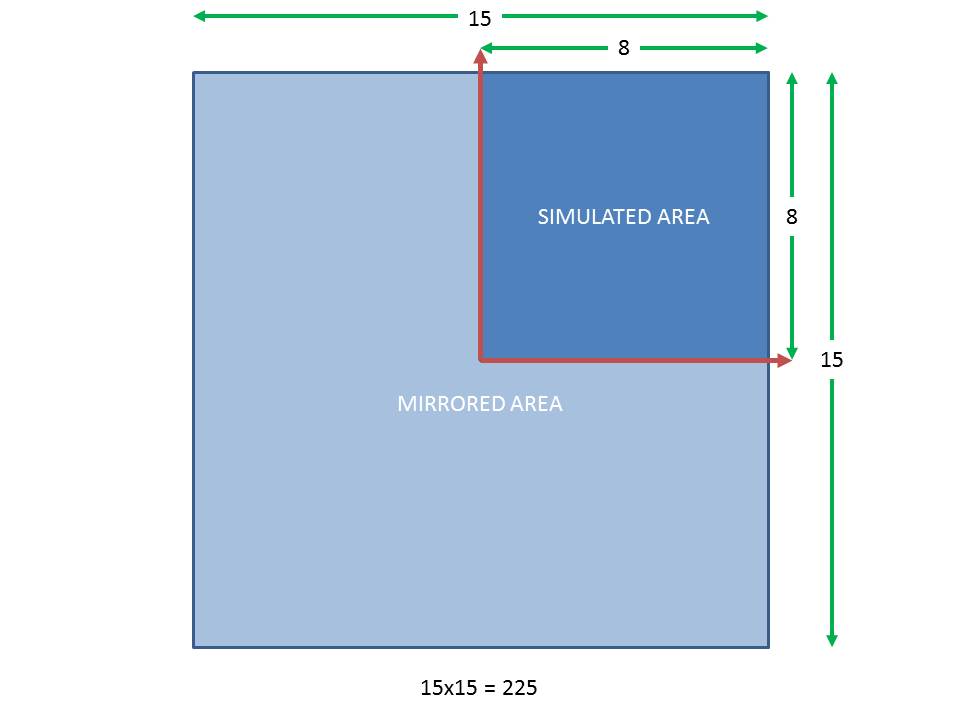


Figure : Drum Mirroring

## Mathematical Considerations

The 2D wave equation was used to emulate the states of the nodes on the drum. To facilitate ensuring that we can squeeze as many nodes as possible onto the FPGA, we adjusted the allowed inputs to the wave equation to only used powers of two when multiplying or dividing. While this reduces the number of variations you can have in the drum sound, it does mean that multiply and divide elements can be replaced with simple left and right arithmetic shifts. A multiplier takes up considerable space on the FPGA, whereas many more shift registers can be added to the logic.

PUT THE MAT FORMULAE HERE (I have no internet and can’t get it)

## Node Generation and Individual Nodes

A Matlab script is used to generate node modules. In the main Verilog file, a node module is defined as follows:

**module** node**(**left**,** right**,** up**,** down**,** clk**,** reset**,** resetval**,** value**,** sw**);**

Each node is given as an input the values of the nodes to the left, right top and bottom of it. It is also connected to the clock, the reset line, and the switches. The switches are used to control the coefficients of the wave equation, which permitted changing the drum sound on the fly.

The MATLAB script generates a module that creates all the node modules as defined by the previous code snippet. Edge nodes are assigned to the appropriate edge values. On the negative edge of the clock, all the wires withing the combinational logic nodes are written out to registers that can be used to update the nodes on the next step, and to generate the output waveform.

On the positive edge of the clock, all of these values are read from the registers, and assigned appropriately to the new nodes. Here too, the wave equation (redefined using shifts and additions) is calculated so it can be used in the new timestep to determine values. This version of the wave equation is shown below:

newval **=** **(**rhocoeff2 **+** **(**prev **+** prev**)** **-** prev2 **+** **(**prev2**>>>**sw**[**17**:**4**]))** **-** **((**rhocoeff2 **+** **(**prev **+** prev**)** **-** prev2 **+** **(**prev2**>>>**sw**[**17**:**4**]))>>>**sw**[**17**:**4**]);**

## Node Connectivity

As was previously mentioned, nodes are connected to each other on the top, bottom, left and right. The MATLAB script is responsible for determining the appropriate relations between nodes and previous timesteps, and calculating the new node values. Since each node module is given its neighbors values as an input, the new node can simply use these values to update its current state. To prevent infinite ringing, node values are set to zero below a threshold that is defined within the matlab script.

## Audio Codec Communication

The diagram below shows how the Audio Codec facilitates communication with the logic of our FPGA. A 16 bit signed value is transmitted to the codec representing the voltage of the audio signal from -1 to +1 volts. The Audio communication is clocked at <VALUE HERE>. Although we are capable of calculating nodes at a faster rate of 50MHz, we instead calculate them at the clock rate of the audio communication so that a buffer is not needed to store the calculated value.

PUT A PRETTY PICTURE HERE.

# Documentation

## Power Spectrum Plot

Comparison with the power spectrum confirms that our simulation is very close to what you would expect to see

PUT SPECTRUM SCREENSHOT HERE AND TALK ABOUT ALL THE HARMONICS WE HIT

## Demo Video

PUT LINK TO YOUTUBE VIDEO HERE

# Program Listing

There are two important programs that we wrote. A MATLAB script was used to generate a Verilog file containing the linkage information for all the nodes. Using this script, the node setup can easily be modified for the number of the nodes. The matlab script, and the code it generates are listed below. Also listed is the primary Verilog file which determines how the nodes actually function and interact with the audio codec to produce a sound.

## Matlab Script

quarterGridNodes1D **=** 8**;** %length of one dimension of the quarter grid

fileName **=** 'nodes.v'**;** %The Filename to save as

format long %We've got some long numbers.

fileID **=** fopen**(**fileName**,** 'wt'**);** %Opens the verilog file for writing

fprintf**(**fileID**,** 'module nodes (restart, clk, audio\_out, sw);\n'**);**

fprintf**(**fileID**,** '\toutput wire signed [15:0] audio\_out;\n'**);**

fprintf**(**fileID**,** '\tinput clk, restart;\n'**);**

fprintf**(**fileID**,** '\tinput [17:0] sw;\n'**);**

fprintf**(**fileID**,** '\n\n'**);**

%Norm PDF is used to generate simple gaussian distribution for the initial drum hit.

v **=** normpdf**([**0 0**;** 0 1**;** 0 2**;** 0 3**;** 1 0**;** 1 1**;** 1 2**;** 1 3**;** 2 0**;** 2 1**;**2 2**;** 2 3**;** 3 0**;** 3 1**;** 3 2**;** 3 3 **],[**0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**;** 0 0**],** **[**1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**;** 1 1**]);**

%We generate node connectivity info for all the nodes, and apply appropriate edge conditions

**for** x **=** 0**:**quarterGridNodes1D**-**1

**for** y **=** 0**:**quarterGridNodes1D**-**1

fprintf**(**fileID**,** strcat**(**'\twire signed[17:0] vwire\_'**,** int2str**(**x**),** '\_'**,** int2str**(**y**),** ';\n'**));**

fprintf**(**fileID**,** strcat**(**'\treg signed[17:0] vreg\_'**,** int2str**(**x**),** '\_'**,** int2str**(**y**),** ';\n'**));**

%Deal with Edge Cases and center cases for X. Note that 1 Quarter Symmetrry is being Used

**if(**x **==** 0**)**

left **=** strcat**(**'vreg\_' **,** int2str**(**x**+**1**)** **,** '\_' **,** int2str**(**y**));**

right **=** strcat**(**'vreg\_' **,** int2str**(**x**+**1**)** **,** '\_' **,** int2str**(**y**));**

**elseif** **(**x **==** quarterGridNodes1D**-**1**)**

left **=** strcat**(**'vreg\_' **,** int2str**(**x**-**1**)** **,** '\_' **,** int2str**(**y**));**

right **=** '18''b0'**;**

**else**

left **=** strcat**(**'vreg\_' **,** int2str**(**x**-**1**)** **,** '\_' **,** int2str**(**y**));**

right **=** strcat**(**'vreg\_' **,** int2str**(**x**+**1**)** **,** '\_' **,** int2str**(**y**));**

**end**

%Deal with Edge Cases and center cases for Y. Note that 1 Quarter Symmetrry is being Used

**if(**y **==** 0**)**

up **=** strcat**(**'vreg\_' **,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**+**1**));**

down **=** strcat**(**'vreg\_' **,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**+**1**));**

**elseif** **(**y **==** quarterGridNodes1D**-**1**)**

up **=** '18''b0'**;**

down **=** strcat**(**'vreg\_' **,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**-**1**));**

**else**

up **=** strcat**(**'vreg\_' **,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**+**1**));**

down **=** strcat**(**'vreg\_' **,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**-**1**));**

**end**

value **=** strcat**(**'vwire\_'**,** int2str**(**x**)** **,** '\_' **,** int2str**(**y**));**

%Prevent Ringing by setting values to zero when they get really small

resetnum **=** ''**;**

**if(**x**<**4**)** **&&** **(**y**<**4**)**

val **=** normpdf**([**x**,**y**],**0**,**1**);**

num **=** val**(**1**)\***val**(**2**)\***4**;**

**if** num **<**.0001

num **=** 0**;**

**end**

num **=** min**(**num**,** 1**)\*(**2**^**16**);**

resetnum **=** dec2bin**(**round**(**num**),** 16**);**

**end**

resetval **=** **[**'18''b00'**,** resetnum**];**

%Generate the Node Module

fprintf**(**fileID**,** **[**'\tnode n'**,** int2str**(**x**),** '\_'**,** int2str**(**y**),** '(.left('**,** left**,** '), .right('**,** right**,** '), .up('**,** up**,** '), .down('**,** down**,** '), .clk(clk), .reset(restart), .resetval('**,** resetval**,** '), .value('**,** value**,** '), .sw(sw));\n'**]);**

**end**

**end**

%Handle the Update

fprintf**(**fileID**,** '\talways @ (negedge clk)\n'**);**

fprintf**(**fileID**,** '\tbegin\n'**);**

**for** x **=** 0**:**quarterGridNodes1D**-**1

**for** y **=** 0**:**quarterGridNodes1D**-**1

fprintf**(**fileID**,** **[**'\t\tvreg\_'**,** int2str**(**x**),** '\_'**,** int2str**(**y**),** ' <= vwire\_'**,** int2str**(**x**),** '\_'**,** int2str**(**y**),** ';\n'**]);**

**end**

**end**

fprintf**(**fileID**,** '\tend\n'**);**

%Set the center node output to the audio output

fprintf**(**fileID**,** '\n\tassign audio\_out = vwire\_0\_0[17:2];\n'**);**

fprintf**(**fileID**,** 'endmodule\n'**);**

## Code Generated with Matlab Script

**module** nodes **(**restart**,** clk**,** audio\_out**,** sw**);**

**output** **wire** **signed** **[**15**:**0**]** audio\_out**;**

**input** clk**,** restart**;**

**input** **[**17**:**0**]** sw**;**

**wire** **signed[**17**:**0**]** vwire\_0\_0**;**

**reg** **signed[**17**:**0**]** vreg\_0\_0**;**

node n0\_0**(.**left**(**vreg\_1\_0**),** **.**right**(**vreg\_1\_0**),** **.**up**(**vreg\_0\_1**),** **.**down**(**vreg\_0\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b001010001011111010**),** **.**value**(**vwire\_0\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_1**;**

**reg** **signed[**17**:**0**]** vreg\_0\_1**;**

node n0\_1**(.**left**(**vreg\_1\_1**),** **.**right**(**vreg\_1\_1**),** **.**up**(**vreg\_0\_2**),** **.**down**(**vreg\_0\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000110001011011001**),** **.**value**(**vwire\_0\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_2**;**

**reg** **signed[**17**:**0**]** vreg\_0\_2**;**

node n0\_2**(.**left**(**vreg\_1\_2**),** **.**right**(**vreg\_1\_2**),** **.**up**(**vreg\_0\_3**),** **.**down**(**vreg\_0\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000001011000001110**),** **.**value**(**vwire\_0\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_3**;**

**reg** **signed[**17**:**0**]** vreg\_0\_3**;**

node n0\_3**(.**left**(**vreg\_1\_3**),** **.**right**(**vreg\_1\_3**),** **.**up**(**vreg\_0\_4**),** **.**down**(**vreg\_0\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000111001111**),** **.**value**(**vwire\_0\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_4**;**

**reg** **signed[**17**:**0**]** vreg\_0\_4**;**

node n0\_4**(.**left**(**vreg\_1\_4**),** **.**right**(**vreg\_1\_4**),** **.**up**(**vreg\_0\_5**),** **.**down**(**vreg\_0\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_0\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_5**;**

**reg** **signed[**17**:**0**]** vreg\_0\_5**;**

node n0\_5**(.**left**(**vreg\_1\_5**),** **.**right**(**vreg\_1\_5**),** **.**up**(**vreg\_0\_6**),** **.**down**(**vreg\_0\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_0\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_6**;**

**reg** **signed[**17**:**0**]** vreg\_0\_6**;**

node n0\_6**(.**left**(**vreg\_1\_6**),** **.**right**(**vreg\_1\_6**),** **.**up**(**vreg\_0\_7**),** **.**down**(**vreg\_0\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_0\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_0\_7**;**

**reg** **signed[**17**:**0**]** vreg\_0\_7**;**

node n0\_7**(.**left**(**vreg\_1\_7**),** **.**right**(**vreg\_1\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_0\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_0\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_0**;**

**reg** **signed[**17**:**0**]** vreg\_1\_0**;**

node n1\_0**(.**left**(**vreg\_0\_0**),** **.**right**(**vreg\_2\_0**),** **.**up**(**vreg\_1\_1**),** **.**down**(**vreg\_1\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000110001011011001**),** **.**value**(**vwire\_1\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_1**;**

**reg** **signed[**17**:**0**]** vreg\_1\_1**;**

node n1\_1**(.**left**(**vreg\_0\_1**),** **.**right**(**vreg\_2\_1**),** **.**up**(**vreg\_1\_2**),** **.**down**(**vreg\_1\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000011101111110100**),** **.**value**(**vwire\_1\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_2**;**

**reg** **signed[**17**:**0**]** vreg\_1\_2**;**

node n1\_2**(.**left**(**vreg\_0\_2**),** **.**right**(**vreg\_2\_2**),** **.**up**(**vreg\_1\_3**),** **.**down**(**vreg\_1\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000110101100001**),** **.**value**(**vwire\_1\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_3**;**

**reg** **signed[**17**:**0**]** vreg\_1\_3**;**

node n1\_3**(.**left**(**vreg\_0\_3**),** **.**right**(**vreg\_2\_3**),** **.**up**(**vreg\_1\_4**),** **.**down**(**vreg\_1\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000100011001**),** **.**value**(**vwire\_1\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_4**;**

**reg** **signed[**17**:**0**]** vreg\_1\_4**;**

node n1\_4**(.**left**(**vreg\_0\_4**),** **.**right**(**vreg\_2\_4**),** **.**up**(**vreg\_1\_5**),** **.**down**(**vreg\_1\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_1\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_5**;**

**reg** **signed[**17**:**0**]** vreg\_1\_5**;**

node n1\_5**(.**left**(**vreg\_0\_5**),** **.**right**(**vreg\_2\_5**),** **.**up**(**vreg\_1\_6**),** **.**down**(**vreg\_1\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_1\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_6**;**

**reg** **signed[**17**:**0**]** vreg\_1\_6**;**

node n1\_6**(.**left**(**vreg\_0\_6**),** **.**right**(**vreg\_2\_6**),** **.**up**(**vreg\_1\_7**),** **.**down**(**vreg\_1\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_1\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_1\_7**;**

**reg** **signed[**17**:**0**]** vreg\_1\_7**;**

node n1\_7**(.**left**(**vreg\_0\_7**),** **.**right**(**vreg\_2\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_1\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_1\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_0**;**

**reg** **signed[**17**:**0**]** vreg\_2\_0**;**

node n2\_0**(.**left**(**vreg\_1\_0**),** **.**right**(**vreg\_3\_0**),** **.**up**(**vreg\_2\_1**),** **.**down**(**vreg\_2\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000001011000001110**),** **.**value**(**vwire\_2\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_1**;**

**reg** **signed[**17**:**0**]** vreg\_2\_1**;**

node n2\_1**(.**left**(**vreg\_1\_1**),** **.**right**(**vreg\_3\_1**),** **.**up**(**vreg\_2\_2**),** **.**down**(**vreg\_2\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000110101100001**),** **.**value**(**vwire\_2\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_2**;**

**reg** **signed[**17**:**0**]** vreg\_2\_2**;**

node n2\_2**(.**left**(**vreg\_1\_2**),** **.**right**(**vreg\_3\_2**),** **.**up**(**vreg\_2\_3**),** **.**down**(**vreg\_2\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000001011111100**),** **.**value**(**vwire\_2\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_3**;**

**reg** **signed[**17**:**0**]** vreg\_2\_3**;**

node n2\_3**(.**left**(**vreg\_1\_3**),** **.**right**(**vreg\_3\_3**),** **.**up**(**vreg\_2\_4**),** **.**down**(**vreg\_2\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000000111111**),** **.**value**(**vwire\_2\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_4**;**

**reg** **signed[**17**:**0**]** vreg\_2\_4**;**

node n2\_4**(.**left**(**vreg\_1\_4**),** **.**right**(**vreg\_3\_4**),** **.**up**(**vreg\_2\_5**),** **.**down**(**vreg\_2\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_2\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_5**;**

**reg** **signed[**17**:**0**]** vreg\_2\_5**;**

node n2\_5**(.**left**(**vreg\_1\_5**),** **.**right**(**vreg\_3\_5**),** **.**up**(**vreg\_2\_6**),** **.**down**(**vreg\_2\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_2\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_6**;**

**reg** **signed[**17**:**0**]** vreg\_2\_6**;**

node n2\_6**(.**left**(**vreg\_1\_6**),** **.**right**(**vreg\_3\_6**),** **.**up**(**vreg\_2\_7**),** **.**down**(**vreg\_2\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_2\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_2\_7**;**

**reg** **signed[**17**:**0**]** vreg\_2\_7**;**

node n2\_7**(.**left**(**vreg\_1\_7**),** **.**right**(**vreg\_3\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_2\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_2\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_0**;**

**reg** **signed[**17**:**0**]** vreg\_3\_0**;**

node n3\_0**(.**left**(**vreg\_2\_0**),** **.**right**(**vreg\_4\_0**),** **.**up**(**vreg\_3\_1**),** **.**down**(**vreg\_3\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000111001111**),** **.**value**(**vwire\_3\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_1**;**

**reg** **signed[**17**:**0**]** vreg\_3\_1**;**

node n3\_1**(.**left**(**vreg\_2\_1**),** **.**right**(**vreg\_4\_1**),** **.**up**(**vreg\_3\_2**),** **.**down**(**vreg\_3\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000100011001**),** **.**value**(**vwire\_3\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_2**;**

**reg** **signed[**17**:**0**]** vreg\_3\_2**;**

node n3\_2**(.**left**(**vreg\_2\_2**),** **.**right**(**vreg\_4\_2**),** **.**up**(**vreg\_3\_3**),** **.**down**(**vreg\_3\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000000111111**),** **.**value**(**vwire\_3\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_3**;**

**reg** **signed[**17**:**0**]** vreg\_3\_3**;**

node n3\_3**(.**left**(**vreg\_2\_3**),** **.**right**(**vreg\_4\_3**),** **.**up**(**vreg\_3\_4**),** **.**down**(**vreg\_3\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b000000000000000000**),** **.**value**(**vwire\_3\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_4**;**

**reg** **signed[**17**:**0**]** vreg\_3\_4**;**

node n3\_4**(.**left**(**vreg\_2\_4**),** **.**right**(**vreg\_4\_4**),** **.**up**(**vreg\_3\_5**),** **.**down**(**vreg\_3\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_3\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_5**;**

**reg** **signed[**17**:**0**]** vreg\_3\_5**;**

node n3\_5**(.**left**(**vreg\_2\_5**),** **.**right**(**vreg\_4\_5**),** **.**up**(**vreg\_3\_6**),** **.**down**(**vreg\_3\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_3\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_6**;**

**reg** **signed[**17**:**0**]** vreg\_3\_6**;**

node n3\_6**(.**left**(**vreg\_2\_6**),** **.**right**(**vreg\_4\_6**),** **.**up**(**vreg\_3\_7**),** **.**down**(**vreg\_3\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_3\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_3\_7**;**

**reg** **signed[**17**:**0**]** vreg\_3\_7**;**

node n3\_7**(.**left**(**vreg\_2\_7**),** **.**right**(**vreg\_4\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_3\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_3\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_0**;**

**reg** **signed[**17**:**0**]** vreg\_4\_0**;**

node n4\_0**(.**left**(**vreg\_3\_0**),** **.**right**(**vreg\_5\_0**),** **.**up**(**vreg\_4\_1**),** **.**down**(**vreg\_4\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_1**;**

**reg** **signed[**17**:**0**]** vreg\_4\_1**;**

node n4\_1**(.**left**(**vreg\_3\_1**),** **.**right**(**vreg\_5\_1**),** **.**up**(**vreg\_4\_2**),** **.**down**(**vreg\_4\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_2**;**

**reg** **signed[**17**:**0**]** vreg\_4\_2**;**

node n4\_2**(.**left**(**vreg\_3\_2**),** **.**right**(**vreg\_5\_2**),** **.**up**(**vreg\_4\_3**),** **.**down**(**vreg\_4\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_3**;**

**reg** **signed[**17**:**0**]** vreg\_4\_3**;**

node n4\_3**(.**left**(**vreg\_3\_3**),** **.**right**(**vreg\_5\_3**),** **.**up**(**vreg\_4\_4**),** **.**down**(**vreg\_4\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_4**;**

**reg** **signed[**17**:**0**]** vreg\_4\_4**;**

node n4\_4**(.**left**(**vreg\_3\_4**),** **.**right**(**vreg\_5\_4**),** **.**up**(**vreg\_4\_5**),** **.**down**(**vreg\_4\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_5**;**

**reg** **signed[**17**:**0**]** vreg\_4\_5**;**

node n4\_5**(.**left**(**vreg\_3\_5**),** **.**right**(**vreg\_5\_5**),** **.**up**(**vreg\_4\_6**),** **.**down**(**vreg\_4\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_6**;**

**reg** **signed[**17**:**0**]** vreg\_4\_6**;**

node n4\_6**(.**left**(**vreg\_3\_6**),** **.**right**(**vreg\_5\_6**),** **.**up**(**vreg\_4\_7**),** **.**down**(**vreg\_4\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_4\_7**;**

**reg** **signed[**17**:**0**]** vreg\_4\_7**;**

node n4\_7**(.**left**(**vreg\_3\_7**),** **.**right**(**vreg\_5\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_4\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_4\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_0**;**

**reg** **signed[**17**:**0**]** vreg\_5\_0**;**

node n5\_0**(.**left**(**vreg\_4\_0**),** **.**right**(**vreg\_6\_0**),** **.**up**(**vreg\_5\_1**),** **.**down**(**vreg\_5\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_1**;**

**reg** **signed[**17**:**0**]** vreg\_5\_1**;**

node n5\_1**(.**left**(**vreg\_4\_1**),** **.**right**(**vreg\_6\_1**),** **.**up**(**vreg\_5\_2**),** **.**down**(**vreg\_5\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_2**;**

**reg** **signed[**17**:**0**]** vreg\_5\_2**;**

node n5\_2**(.**left**(**vreg\_4\_2**),** **.**right**(**vreg\_6\_2**),** **.**up**(**vreg\_5\_3**),** **.**down**(**vreg\_5\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_3**;**

**reg** **signed[**17**:**0**]** vreg\_5\_3**;**

node n5\_3**(.**left**(**vreg\_4\_3**),** **.**right**(**vreg\_6\_3**),** **.**up**(**vreg\_5\_4**),** **.**down**(**vreg\_5\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_4**;**

**reg** **signed[**17**:**0**]** vreg\_5\_4**;**

node n5\_4**(.**left**(**vreg\_4\_4**),** **.**right**(**vreg\_6\_4**),** **.**up**(**vreg\_5\_5**),** **.**down**(**vreg\_5\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_5**;**

**reg** **signed[**17**:**0**]** vreg\_5\_5**;**

node n5\_5**(.**left**(**vreg\_4\_5**),** **.**right**(**vreg\_6\_5**),** **.**up**(**vreg\_5\_6**),** **.**down**(**vreg\_5\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_6**;**

**reg** **signed[**17**:**0**]** vreg\_5\_6**;**

node n5\_6**(.**left**(**vreg\_4\_6**),** **.**right**(**vreg\_6\_6**),** **.**up**(**vreg\_5\_7**),** **.**down**(**vreg\_5\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_5\_7**;**

**reg** **signed[**17**:**0**]** vreg\_5\_7**;**

node n5\_7**(.**left**(**vreg\_4\_7**),** **.**right**(**vreg\_6\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_5\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_5\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_0**;**

**reg** **signed[**17**:**0**]** vreg\_6\_0**;**

node n6\_0**(.**left**(**vreg\_5\_0**),** **.**right**(**vreg\_7\_0**),** **.**up**(**vreg\_6\_1**),** **.**down**(**vreg\_6\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_1**;**

**reg** **signed[**17**:**0**]** vreg\_6\_1**;**

node n6\_1**(.**left**(**vreg\_5\_1**),** **.**right**(**vreg\_7\_1**),** **.**up**(**vreg\_6\_2**),** **.**down**(**vreg\_6\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_2**;**

**reg** **signed[**17**:**0**]** vreg\_6\_2**;**

node n6\_2**(.**left**(**vreg\_5\_2**),** **.**right**(**vreg\_7\_2**),** **.**up**(**vreg\_6\_3**),** **.**down**(**vreg\_6\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_3**;**

**reg** **signed[**17**:**0**]** vreg\_6\_3**;**

node n6\_3**(.**left**(**vreg\_5\_3**),** **.**right**(**vreg\_7\_3**),** **.**up**(**vreg\_6\_4**),** **.**down**(**vreg\_6\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_4**;**

**reg** **signed[**17**:**0**]** vreg\_6\_4**;**

node n6\_4**(.**left**(**vreg\_5\_4**),** **.**right**(**vreg\_7\_4**),** **.**up**(**vreg\_6\_5**),** **.**down**(**vreg\_6\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_5**;**

**reg** **signed[**17**:**0**]** vreg\_6\_5**;**

node n6\_5**(.**left**(**vreg\_5\_5**),** **.**right**(**vreg\_7\_5**),** **.**up**(**vreg\_6\_6**),** **.**down**(**vreg\_6\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_6**;**

**reg** **signed[**17**:**0**]** vreg\_6\_6**;**

node n6\_6**(.**left**(**vreg\_5\_6**),** **.**right**(**vreg\_7\_6**),** **.**up**(**vreg\_6\_7**),** **.**down**(**vreg\_6\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_6\_7**;**

**reg** **signed[**17**:**0**]** vreg\_6\_7**;**

node n6\_7**(.**left**(**vreg\_5\_7**),** **.**right**(**vreg\_7\_7**),** **.**up**(**18'b0**),** **.**down**(**vreg\_6\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_6\_7**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_0**;**

**reg** **signed[**17**:**0**]** vreg\_7\_0**;**

node n7\_0**(.**left**(**vreg\_6\_0**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_1**),** **.**down**(**vreg\_7\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_0**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_1**;**

**reg** **signed[**17**:**0**]** vreg\_7\_1**;**

node n7\_1**(.**left**(**vreg\_6\_1**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_2**),** **.**down**(**vreg\_7\_0**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_1**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_2**;**

**reg** **signed[**17**:**0**]** vreg\_7\_2**;**

node n7\_2**(.**left**(**vreg\_6\_2**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_3**),** **.**down**(**vreg\_7\_1**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_2**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_3**;**

**reg** **signed[**17**:**0**]** vreg\_7\_3**;**

node n7\_3**(.**left**(**vreg\_6\_3**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_4**),** **.**down**(**vreg\_7\_2**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_3**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_4**;**

**reg** **signed[**17**:**0**]** vreg\_7\_4**;**

node n7\_4**(.**left**(**vreg\_6\_4**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_5**),** **.**down**(**vreg\_7\_3**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_4**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_5**;**

**reg** **signed[**17**:**0**]** vreg\_7\_5**;**

node n7\_5**(.**left**(**vreg\_6\_5**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_6**),** **.**down**(**vreg\_7\_4**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_5**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_6**;**

**reg** **signed[**17**:**0**]** vreg\_7\_6**;**

node n7\_6**(.**left**(**vreg\_6\_6**),** **.**right**(**18'b0**),** **.**up**(**vreg\_7\_7**),** **.**down**(**vreg\_7\_5**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_6**),** **.**sw**(**sw**));**

**wire** **signed[**17**:**0**]** vwire\_7\_7**;**

**reg** **signed[**17**:**0**]** vreg\_7\_7**;**

node n7\_7**(.**left**(**vreg\_6\_7**),** **.**right**(**18'b0**),** **.**up**(**18'b0**),** **.**down**(**vreg\_7\_6**),** **.**clk**(**clk**),** **.**reset**(**restart**),** **.**resetval**(**18'b00**),** **.**value**(**vwire\_7\_7**),** **.**sw**(**sw**));**

**always** **@** **(negedge** clk**)**

**begin**

vreg\_0\_0 **<=** vwire\_0\_0**;**

vreg\_0\_1 **<=** vwire\_0\_1**;**

vreg\_0\_2 **<=** vwire\_0\_2**;**

vreg\_0\_3 **<=** vwire\_0\_3**;**

vreg\_0\_4 **<=** vwire\_0\_4**;**

vreg\_0\_5 **<=** vwire\_0\_5**;**

vreg\_0\_6 **<=** vwire\_0\_6**;**

vreg\_0\_7 **<=** vwire\_0\_7**;**

vreg\_1\_0 **<=** vwire\_1\_0**;**

vreg\_1\_1 **<=** vwire\_1\_1**;**

vreg\_1\_2 **<=** vwire\_1\_2**;**

vreg\_1\_3 **<=** vwire\_1\_3**;**

vreg\_1\_4 **<=** vwire\_1\_4**;**

vreg\_1\_5 **<=** vwire\_1\_5**;**

vreg\_1\_6 **<=** vwire\_1\_6**;**

vreg\_1\_7 **<=** vwire\_1\_7**;**

vreg\_2\_0 **<=** vwire\_2\_0**;**

vreg\_2\_1 **<=** vwire\_2\_1**;**

vreg\_2\_2 **<=** vwire\_2\_2**;**

vreg\_2\_3 **<=** vwire\_2\_3**;**

vreg\_2\_4 **<=** vwire\_2\_4**;**

vreg\_2\_5 **<=** vwire\_2\_5**;**

vreg\_2\_6 **<=** vwire\_2\_6**;**

vreg\_2\_7 **<=** vwire\_2\_7**;**

vreg\_3\_0 **<=** vwire\_3\_0**;**

vreg\_3\_1 **<=** vwire\_3\_1**;**

vreg\_3\_2 **<=** vwire\_3\_2**;**

vreg\_3\_3 **<=** vwire\_3\_3**;**

vreg\_3\_4 **<=** vwire\_3\_4**;**

vreg\_3\_5 **<=** vwire\_3\_5**;**

vreg\_3\_6 **<=** vwire\_3\_6**;**

vreg\_3\_7 **<=** vwire\_3\_7**;**

vreg\_4\_0 **<=** vwire\_4\_0**;**

vreg\_4\_1 **<=** vwire\_4\_1**;**

vreg\_4\_2 **<=** vwire\_4\_2**;**

vreg\_4\_3 **<=** vwire\_4\_3**;**

vreg\_4\_4 **<=** vwire\_4\_4**;**

vreg\_4\_5 **<=** vwire\_4\_5**;**

vreg\_4\_6 **<=** vwire\_4\_6**;**

vreg\_4\_7 **<=** vwire\_4\_7**;**

vreg\_5\_0 **<=** vwire\_5\_0**;**

vreg\_5\_1 **<=** vwire\_5\_1**;**

vreg\_5\_2 **<=** vwire\_5\_2**;**

vreg\_5\_3 **<=** vwire\_5\_3**;**

vreg\_5\_4 **<=** vwire\_5\_4**;**

vreg\_5\_5 **<=** vwire\_5\_5**;**

vreg\_5\_6 **<=** vwire\_5\_6**;**

vreg\_5\_7 **<=** vwire\_5\_7**;**

vreg\_6\_0 **<=** vwire\_6\_0**;**

vreg\_6\_1 **<=** vwire\_6\_1**;**

vreg\_6\_2 **<=** vwire\_6\_2**;**

vreg\_6\_3 **<=** vwire\_6\_3**;**

vreg\_6\_4 **<=** vwire\_6\_4**;**

vreg\_6\_5 **<=** vwire\_6\_5**;**

vreg\_6\_6 **<=** vwire\_6\_6**;**

vreg\_6\_7 **<=** vwire\_6\_7**;**

vreg\_7\_0 **<=** vwire\_7\_0**;**

vreg\_7\_1 **<=** vwire\_7\_1**;**

vreg\_7\_2 **<=** vwire\_7\_2**;**

vreg\_7\_3 **<=** vwire\_7\_3**;**

vreg\_7\_4 **<=** vwire\_7\_4**;**

vreg\_7\_5 **<=** vwire\_7\_5**;**

vreg\_7\_6 **<=** vwire\_7\_6**;**

vreg\_7\_7 **<=** vwire\_7\_7**;**

**end**

**assign** audio\_out **=** vwire\_0\_0**[**17**:**2**];**

**endmodule**

### Top Level Verilog Program

// --------------------------------------------------------------------

// Drum Synthesizer

// Jeremy Blum, Sima Mitra, Jason Wright

// --------------------------------------------------------------------

**module** DE2\_TOP **(**

// Clock Input

**input** CLOCK\_27**,** // 27 MHz

**input** CLOCK\_50**,** // 50 MHz

**input** EXT\_CLOCK**,** // External Clock

// Push Button

**input** **[**3**:**0**]** KEY**,** // Pushbutton[3:0]

// DPDT Switch

**input** **[**17**:**0**]** SW**,** // Toggle Switch[17:0]

// 7-SEG Display

**output** **[**6**:**0**]** HEX0**,** // Seven Segment Digit 0

**output** **[**6**:**0**]** HEX1**,** // Seven Segment Digit 1

**output** **[**6**:**0**]** HEX2**,** // Seven Segment Digit 2

**output** **[**6**:**0**]** HEX3**,** // Seven Segment Digit 3

**output** **[**6**:**0**]** HEX4**,** // Seven Segment Digit 4

**output** **[**6**:**0**]** HEX5**,** // Seven Segment Digit 5

**output** **[**6**:**0**]** HEX6**,** // Seven Segment Digit 6

**output** **[**6**:**0**]** HEX7**,** // Seven Segment Digit 7

// LED

**output** **[**8**:**0**]** LEDG**,** // LED Green[8:0]

**output** **[**17**:**0**]** LEDR**,** // LED Red[17:0]

// UART

**output** UART\_TXD**,** // UART Transmitter

**input** UART\_RXD**,** // UART Receiver

// IRDA

**output** IRDA\_TXD**,** // IRDA Transmitter

**input** IRDA\_RXD**,** // IRDA Receiver

// SDRAM Interface

**inout** **[**15**:**0**]** DRAM\_DQ**,** // SDRAM Data bus 16 Bits

**output** **[**11**:**0**]** DRAM\_ADDR**,** // SDRAM Address bus 12 Bits

**output** DRAM\_LDQM**,** // SDRAM Low-byte Data Mask

**output** DRAM\_UDQM**,** // SDRAM High-byte Data Mask

**output** DRAM\_WE\_N**,** // SDRAM Write Enable

**output** DRAM\_CAS\_N**,** // SDRAM Column Address Strobe

**output** DRAM\_RAS\_N**,** // SDRAM Row Address Strobe

**output** DRAM\_CS\_N**,** // SDRAM Chip Select

**output** DRAM\_BA\_0**,** // SDRAM Bank Address 0

**output** DRAM\_BA\_1**,** // SDRAM Bank Address 0

**output** DRAM\_CLK**,** // SDRAM Clock

**output** DRAM\_CKE**,** // SDRAM Clock Enable

// Flash Interface

**inout** **[**7**:**0**]** FL\_DQ**,** // FLASH Data bus 8 Bits

**output** **[**21**:**0**]** FL\_ADDR**,** // FLASH Address bus 22 Bits

**output** FL\_WE\_N**,** // FLASH Write Enable

**output** FL\_RST\_N**,** // FLASH Reset

**output** FL\_OE\_N**,** // FLASH Output Enable

**output** FL\_CE\_N**,** // FLASH Chip Enable

// SRAM Interface

**inout** **[**15**:**0**]** SRAM\_DQ**,** // SRAM Data bus 16 Bits

**output** **[**17**:**0**]** SRAM\_ADDR**,** // SRAM Address bus 18 Bits

**output** SRAM\_UB\_N**,** // SRAM High-byte Data Mask

**output** SRAM\_LB\_N**,** // SRAM Low-byte Data Mask

**output** SRAM\_WE\_N**,** // SRAM Write Enable

**output** SRAM\_CE\_N**,** // SRAM Chip Enable

**output** SRAM\_OE\_N**,** // SRAM Output Enable

// ISP1362 Interface

**inout** **[**15**:**0**]** OTG\_DATA**,** // ISP1362 Data bus 16 Bits

**output** **[**1**:**0**]** OTG\_ADDR**,** // ISP1362 Address 2 Bits

**output** OTG\_CS\_N**,** // ISP1362 Chip Select

**output** OTG\_RD\_N**,** // ISP1362 Write

**output** OTG\_WR\_N**,** // ISP1362 Read

**output** OTG\_RST\_N**,** // ISP1362 Reset

**output** OTG\_FSPEED**,** // USB Full Speed, 0 = Enable, Z = Disable

**output** OTG\_LSPEED**,** // USB Low Speed, 0 = Enable, Z = Disable

**input** OTG\_INT0**,** // ISP1362 Interrupt 0

**input** OTG\_INT1**,** // ISP1362 Interrupt 1

**input** OTG\_DREQ0**,** // ISP1362 DMA Request 0

**input** OTG\_DREQ1**,** // ISP1362 DMA Request 1

**output** OTG\_DACK0\_N**,** // ISP1362 DMA Acknowledge 0

**output** OTG\_DACK1\_N**,** // ISP1362 DMA Acknowledge 1

// LCD Module 16X2

**inout** **[**7**:**0**]** LCD\_DATA**,** // LCD Data bus 8 bits

**output** LCD\_ON**,** // LCD Power ON/OFF

**output** LCD\_BLON**,** // LCD Back Light ON/OFF

**output** LCD\_RW**,** // LCD Read/Write Select, 0 = Write, 1 = Read

**output** LCD\_EN**,** // LCD Enable

**output** LCD\_RS**,** // LCD Command/Data Select, 0 = Command, 1 = Data

// SD Card Interface

**inout** SD\_DAT**,** // SD Card Data

**inout** SD\_DAT3**,** // SD Card Data 3

**inout** SD\_CMD**,** // SD Card Command Signal

**output** SD\_CLK**,** // SD Card Clock

// I2C

**inout** I2C\_SDAT**,** // I2C Data

**output** I2C\_SCLK**,** // I2C Clock

// PS2

**input** PS2\_DAT**,** // PS2 Data

**input** PS2\_CLK**,** // PS2 Clock

// USB JTAG link

**input** TDI**,** // CPLD -> FPGA (data in)

**input** TCK**,** // CPLD -> FPGA (clk)

**input** TCS**,** // CPLD -> FPGA (CS)

**output** TDO**,** // FPGA -> CPLD (data out)

// VGA

**output** VGA\_CLK**,** // VGA Clock

**output** VGA\_HS**,** // VGA H\_SYNC

**output** VGA\_VS**,** // VGA V\_SYNC

**output** VGA\_BLANK**,** // VGA BLANK

**output** VGA\_SYNC**,** // VGA SYNC

**output** **[**9**:**0**]** VGA\_R**,** // VGA Red[9:0]

**output** **[**9**:**0**]** VGA\_G**,** // VGA Green[9:0]

**output** **[**9**:**0**]** VGA\_B**,** // VGA Blue[9:0]

// Ethernet Interface

**inout** **[**15**:**0**]** ENET\_DATA**,** // DM9000A DATA bus 16Bits

**output** ENET\_CMD**,** // DM9000A Command/Data Select, 0 = Command, 1 = Data

**output** ENET\_CS\_N**,** // DM9000A Chip Select

**output** ENET\_WR\_N**,** // DM9000A Write

**output** ENET\_RD\_N**,** // DM9000A Read

**output** ENET\_RST\_N**,** // DM9000A Reset

**input** ENET\_INT**,** // DM9000A Interrupt

**output** ENET\_CLK**,** // DM9000A Clock 25 MHz

// Audio CODEC

**inout** AUD\_ADCLRCK**,** // Audio CODEC ADC LR Clock

**input** AUD\_ADCDAT**,** // Audio CODEC ADC Data

**inout** AUD\_DACLRCK**,** // Audio CODEC DAC LR Clock

**output** AUD\_DACDAT**,** // Audio CODEC DAC Data

**inout** AUD\_BCLK**,** // Audio CODEC Bit-Stream Clock

**output** AUD\_XCK**,** // Audio CODEC Chip Clock

// TV Decoder

**input** **[**7**:**0**]** TD\_DATA**,** // TV Decoder Data bus 8 bits

**input** TD\_HS**,** // TV Decoder H\_SYNC

**input** TD\_VS**,** // TV Decoder V\_SYNC

**output** TD\_RESET**,** // TV Decoder Reset

// GPIO

**inout** **[**35**:**0**]** GPIO\_0**,** // GPIO Connection 0

**inout** **[**35**:**0**]** GPIO\_1 // GPIO Connection 1

**);**

//Turn off all displays.

**assign** HEX0 **=** 7'h7F**;**

**assign** HEX1 **=** 7'h7F**;**

**assign** HEX2 **=** 7'h7F**;**

**assign** HEX3 **=** 7'h7F**;**

**assign** HEX4 **=** 7'h7F**;**

**assign** HEX5 **=** 7'h7F**;**

**assign** HEX6 **=** 7'h7F**;**

**assign** HEX7 **=** 7'h7F**;**

//assign LEDR = 18'h0;

//assign LEDG = 9'h0;

//Set all GPIO to tri-state.

//assign GPIO\_0 = 36'hzzzzzzzzz;

**assign** GPIO\_1 **=** 36'hzzzzzzzzz**;** //We Used this for debugging

//Disable audio codec.

//assign AUD\_DACDAT = 1'b0;

//assign AUD\_XCK = 1'b0;

//Disable DRAM.

**assign** DRAM\_ADDR **=** 12'h0**;**

**assign** DRAM\_BA\_0 **=** 1'b0**;**

**assign** DRAM\_BA\_1 **=** 1'b0**;**

**assign** DRAM\_CAS\_N **=** 1'b1**;**

**assign** DRAM\_CKE **=** 1'b0**;**

**assign** DRAM\_CLK **=** 1'b0**;**

**assign** DRAM\_CS\_N **=** 1'b1**;**

**assign** DRAM\_DQ **=** 16'hzzzz**;**

**assign** DRAM\_LDQM **=** 1'b0**;**

**assign** DRAM\_RAS\_N **=** 1'b1**;**

**assign** DRAM\_UDQM **=** 1'b0**;**

**assign** DRAM\_WE\_N **=** 1'b1**;**

//Disable Ethernet.

**assign** ENET\_CLK **=** 1'b0**;**

**assign** ENET\_CS\_N **=** 1'b1**;**

**assign** ENET\_CMD **=** 1'b0**;**

**assign** ENET\_DATA **=** 16'hzzzz**;**

**assign** ENET\_RD\_N **=** 1'b1**;**

**assign** ENET\_RST\_N **=** 1'b1**;**

**assign** ENET\_WR\_N **=** 1'b1**;**

//Disable flash.

**assign** FL\_ADDR **=** 22'h0**;**

**assign** FL\_CE\_N **=** 1'b1**;**

**assign** FL\_DQ **=** 8'hzz**;**

**assign** FL\_OE\_N **=** 1'b1**;**

**assign** FL\_RST\_N **=** 1'b1**;**

**assign** FL\_WE\_N **=** 1'b1**;**

//Disable LCD.

**assign** LCD\_BLON **=** 1'b0**;**

**assign** LCD\_DATA **=** 8'hzz**;**

**assign** LCD\_EN **=** 1'b0**;**

**assign** LCD\_ON **=** 1'b0**;**

**assign** LCD\_RS **=** 1'b0**;**

**assign** LCD\_RW **=** 1'b0**;**

//Disable OTG.

**assign** OTG\_ADDR **=** 2'h0**;**

**assign** OTG\_CS\_N **=** 1'b1**;**

**assign** OTG\_DACK0\_N **=** 1'b1**;**

**assign** OTG\_DACK1\_N **=** 1'b1**;**

**assign** OTG\_FSPEED **=** 1'b1**;**

**assign** OTG\_DATA **=** 16'hzzzz**;**

**assign** OTG\_LSPEED **=** 1'b1**;**

**assign** OTG\_RD\_N **=** 1'b1**;**

**assign** OTG\_RST\_N **=** 1'b1**;**

**assign** OTG\_WR\_N **=** 1'b1**;**

//Disable SDRAM.

**assign** SD\_DAT **=** 1'bz**;**

**assign** SD\_CLK **=** 1'b0**;**

//Disable SRAM.

**assign** SRAM\_ADDR **=** 18'h0**;**

**assign** SRAM\_CE\_N **=** 1'b1**;**

**assign** SRAM\_DQ **=** 16'hzzzz**;**

**assign** SRAM\_LB\_N **=** 1'b1**;**

**assign** SRAM\_OE\_N **=** 1'b1**;**

**assign** SRAM\_UB\_N **=** 1'b1**;**

**assign** SRAM\_WE\_N **=** 1'b1**;**

//Disable VGA.

//assign VGA\_CLK = 1'b0;

**assign** VGA\_BLANK **=** 1'b0**;**

**assign** VGA\_SYNC **=** 1'b0**;**

**assign** VGA\_HS **=** 1'b0**;**

**assign** VGA\_VS **=** 1'b0**;**

**assign** VGA\_R **=** 10'h0**;**

**assign** VGA\_G **=** 10'h0**;**

**assign** VGA\_B **=** 10'h0**;**

//Disable all other peripherals.

//assign I2C\_SCLK = 1'b0;

**assign** IRDA\_TXD **=** 1'b0**;**

//assign TD\_RESET = 1'b0;

**assign** TDO **=** 1'b0**;**

**assign** UART\_TXD **=** 1'b0**;**

**wire** VGA\_CTRL\_CLK**;**

**wire** AUD\_CTRL\_CLK**;**

**wire** DLY\_RST**;**

**assign** TD\_RESET **=** 1'b1**;** // Allow 27 MHz

**assign** AUD\_ADCLRCK **=** AUD\_DACLRCK**;**

**assign** AUD\_XCK **=** AUD\_CTRL\_CLK**;**

**assign** GPIO\_0**[**0**]** **=** AUD\_DACLRCK**;** //For Debugging the Clock

Reset\_Delay r0 **(** **.**iCLK**(**CLOCK\_50**),.**oRESET**(**DLY\_RST**)** **);**

VGA\_Audio\_PLL p1 **(** **.**areset**(~**DLY\_RST**),.**inclk0**(**CLOCK\_27**),.**c0**(**VGA\_CTRL\_CLK**),.**c1**(**AUD\_CTRL\_CLK**),.**c2**(**VGA\_CLK**)** **);**

//AUD\_CTRL\_CLK = 18 MHz

I2C\_AV\_Config u3 **(** // Host Side

**.**iCLK**(**CLOCK\_50**),**

**.**iRST\_N**(**KEY**[**0**]),**

// I2C Side

**.**I2C\_SCLK**(**I2C\_SCLK**),**

**.**I2C\_SDAT**(**I2C\_SDAT**)** **);**

AUDIO\_DAC\_ADC u4 **(** // Audio Side

**.**oAUD\_BCK**(**AUD\_BCLK**),**

**.**oAUD\_DATA**(**AUD\_DACDAT**),**

**.**oAUD\_LRCK**(**AUD\_DACLRCK**),**

**.**oAUD\_inL**(**audio\_inL**),** // audio data from ADC

**.**oAUD\_inR**(**audio\_inR**),** // audio data from ADC

**.**iAUD\_ADCDAT**(**AUD\_ADCDAT**),**

**.**iAUD\_extL**(**audio\_outL**),** // audio data to DAC

**.**iAUD\_extR**(**audio\_outR**),** // audio data to DAC

// Control Signals

**.**iCLK\_18\_4**(**AUD\_CTRL\_CLK**),**

**.**iRST\_N**(**DLY\_RST**)**

**);**

/// reset ///////////////////////////////////////////////////////

//state machine start up

**wire** reset**;**

// reset control

**assign** reset **=** **~**KEY**[**0**];**

**assign** restart **=** **~**KEY**[**1**];**

/// audio stuff /////////////////////////////////////////////////

// output to audio DAC

**wire** **signed** **[**15**:**0**]** audio\_outL**,** audio\_outR **;**

**wire** **signed[**15**:**0**]** nodes\_out**;**

// make some output

**assign** audio\_outR **=** nodes\_out **;**

**assign** audio\_outL **=** nodes\_out **;**

**assign** GPIO\_0**[**18**:**1**]** **=** nodes\_out**;**

//green LEDs for button debugging

**assign** LEDG**[**0**]** **=** **~**KEY**[**0**];**

**assign** LEDG**[**1**]** **=** **~**KEY**[**0**];**

**assign** LEDG**[**2**]** **=** **~**KEY**[**1**];**

**assign** LEDG**[**3**]** **=** **~**KEY**[**1**];**

**assign** LEDG**[**4**]** **=** **~**KEY**[**2**];**

**assign** LEDG**[**5**]** **=** **~**KEY**[**2**];**

**assign** LEDG**[**6**]** **=** **~**KEY**[**3**];**

**assign** LEDG**[**7**]** **=** **~**KEY**[**3**];**

**assign** LEDR**[**15**:**0**]** **=** nodes\_out**;**

//Creates all the drum nodes

nodes drum **(.**restart**(**restart**),** **.**clk**(**AUD\_DACLRCK**),** **.**audio\_out**(**nodes\_out**),** **.**sw**(**SW**[**17**:**0**]));**

**endmodule**

//Defines the inputs and outputs for a node

**module** node**(**left**,** right**,** up**,** down**,** clk**,** reset**,** resetval**,** value**,** sw**);**

**output** **reg** **signed[**17**:**0**]** value**;**

**input** **signed[**17**:**0**]** left**;**

**input** **signed[**17**:**0**]** right**;**

**input** **signed[**17**:**0**]** up**;**

**input** **signed[**17**:**0**]** down**;**

**input** **[**17**:**0**]** sw**;**

**input** clk**;**

**input** reset**;** //When this is high, initialize to resetval

**input** **wire** **signed[**17**:**0**]** resetval**;** //the initial Gaussian shape

**reg** **signed[**17**:**0**]** prev**;**

**reg** **signed[**17**:**0**]** prev2**;**

**reg** **signed[**63**:**0**]** newval**;**

**reg** **signed[**31**:**0**]** neighbors**;**

**reg** **signed[**31**:**0**]** prevgain**;**

**reg** **signed[**31**:**0**]** rhocoeff**;**

**reg** **signed[**17**:**0**]** rhocoeff2**;**

//Everything gets updated on the positive clock edge

**always** **@** **(posedge** clk**)**

**begin**

**case(**reset**)**

1'b1 **:**

**begin**

//If reset is activated, set everything to the reset state

value **<=** resetval**;**

prev2 **<=** resetval**;**

prev **<=** resetval**;**

**end**

1'b0**:**

**begin**

//Otherwise, connect the neighbors and apply the coefficients

neighbors **=** left**+**right**+**up**+**down**;**

prevgain **=** prev**+**prev**+**prev**+**prev**;**

rhocoeff **=** **(**neighbors**-**prevgain**)>>>**sw**[**3**:**0**];**

rhocoeff2 **=** **{**rhocoeff**[**31**:**30**],** rhocoeff**[**15**:**0**]};**

newval **=** **(**rhocoeff2 **+** **(**prev **+** prev**)** **-** prev2 **+** **(**prev2**>>>**sw**[**17**:**4**]))** **-** **((**rhocoeff2 **+** **(**prev **+** prev**)** **-** prev2 **+** **(**prev2**>>>**sw**[**17**:**4**]))>>>**sw**[**17**:**4**]);**

value **=** **{**newval**[**63**:**62**],** newval**[**15**:**0**]};**

prev2 **=** prev**;**

prev **=** value**;**

**end**

**endcase**

**end**

**endmodule**