Group 19
Frank Luong 260481340
Jeffrey Leung 260402139

G19_Square_Wave

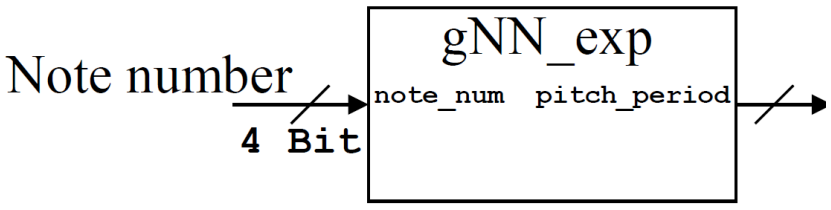
Our team designed a square wave of a music box that is capable of providing musical notes. A frequency divider is used to generate a waveform. It is generated with a counter that counts from the *period* down to 0.The *period* is the pitch_period shifted by an amount octave. As the counter reaches the value 0, the counter resets.

First of all, the selector method is used to implement different notes. The 4-bit binary value represents the note number between 0 to 11. Our team used the 4-bit note number to create the period of the pitch, which is a 20-bit value. (See Figure 1) For example, when we input "0000", the output will be "10111010101000100011". The circuit will output all ones for the 4 invalid inputs, which are 12 through 15.

Table 1

Variable Name	Type	Bit
note_number	in std logic_vector(3 downto 0)	4
pitch_period	out std logic_vector(19 downto 0)	20

Figure 1: The Exponentiator Circuit

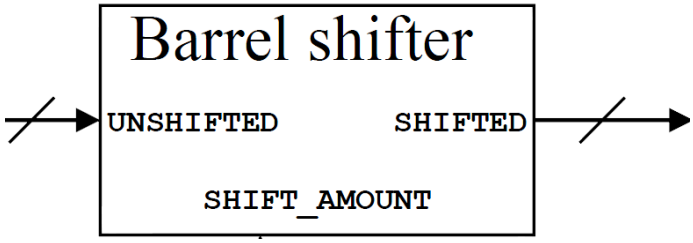


Also, a barrel shifter is needed to shift the data word in one clock cycle, and is provided from the laboratory description.(See Figure 2)

```
PERIOD <= to_stdlogicvector(to_bitvector(pitch_period) SRL
                            to_integer(unsigned(octave)));
```

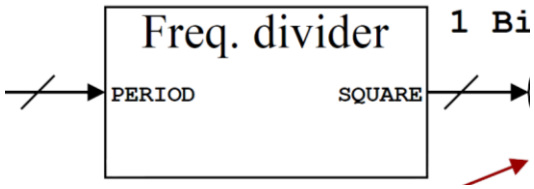
Group 19
Frank Luong 260481340
Jeffrey Leung 260402139

Figure 2: Barrel Shifter



The barrel shifter will output a 20-bit value to the input of the frequency divider. (See Figure 3) The frequency divider divides the clock cycle by the *period*. Note that the frequency divider counter is implemented in a single process block.

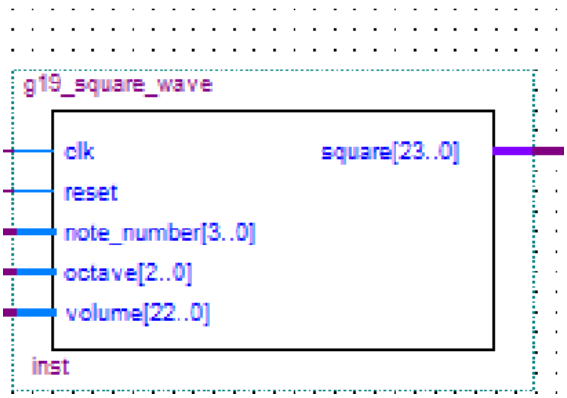
Figure 3: Frequency Divider



Finally, depending on the value of the output of the frequency divider, the multiplication is implemented by using a multiplexer by setting the output to the positive volume or negative volume. The multiplexer's input is a 1-bit value, which is a select line and chooses the corresponding volume.

The overall VHDL codes, including the exponentiator, barrel shifter, frequency divider and the multiplexer is compiled and a schematic block of the square wave is created. (See Figure 4)

Figure 4: Block diagram of the Square Wave

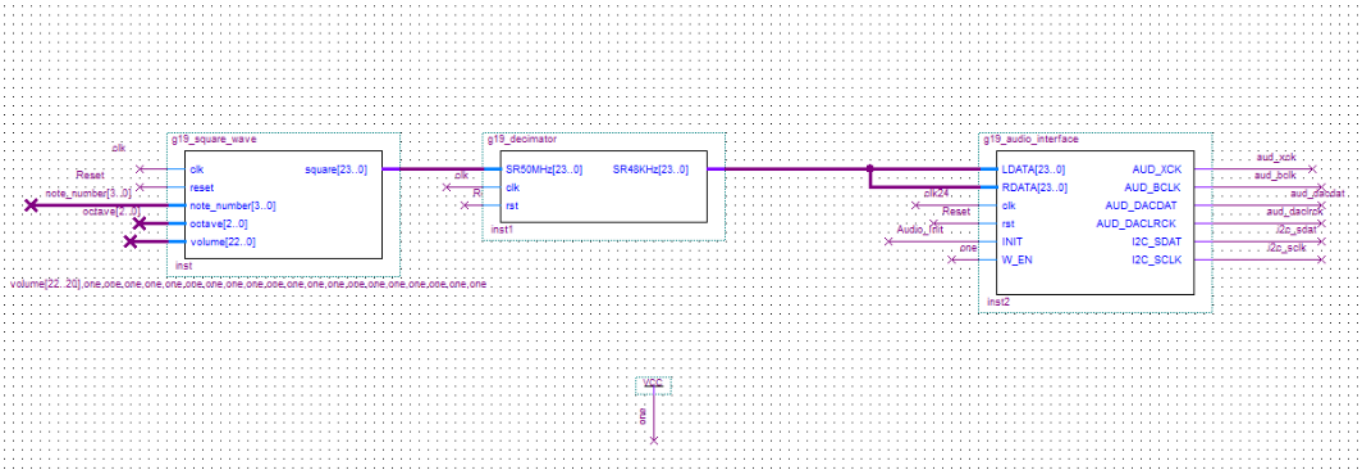


Group 19
Frank Luong 260481340
Jeffrey Leung 260402139

Table 2: List of inputs/outputs

Inputs		Outputs	
Variables	Specifications	Variables	Specifications
clk	A clock for the circuit (50 MHz)	square	24-bit output (signed)
reset	Asynchronous reset		
note_number	4-bit input		
octave	3-bit input		
volume	23-bit input (unsigned)		

Figure 4: Block Diagram of the whole circuit

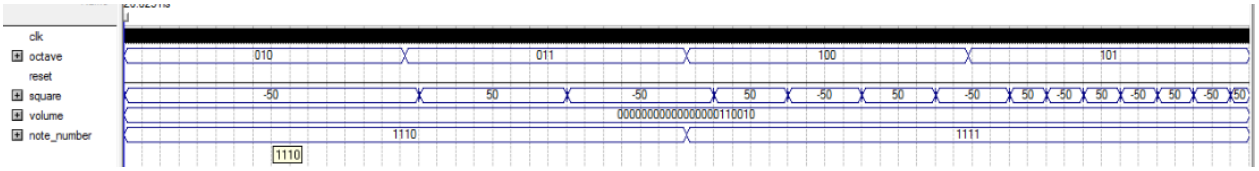


Also, the two audio modules, called it g19_audio_interface and g19_decimator, are provided by Prof. Clark. The audio interface is a module that implements the serial communication to an Audio Codec chip located on the Altera board. On the other hand, concerning the decimator, another codec chip helps to convert the sampling rate of 50 MHz to 48 KHz from the square wave generator.

The note number, which is a three bit input, can be change via switches on the testing board. To check if the square wave works correctly, a simulation is needed to test the generator. In the simulation, we can clearly see that as the octave increases, the square wave has a period that is longer than its previous state. If the octave number is increasing by 1, it will produce a doubling in the frequency. Also, we can see that the square wave its amplitudes alternate between the positive and negative volume (i.e. 50 and -50). Therefore we can safely assume that our circuit is working properly.

Group 19
Frank Luong 260481340
Jeffrey Leung 260402139

Figure 5: Simulation of the Square Wave



The square wave generator is then connected to the DE1 Altera Board. By using g19_audio_interface and g19_decimator, the interfacing and initialization of the codec chip were handled. The Reset line, Audio Init line, octave, note_number and the three MSB of the volume (Note that the other bits are set to a constant '1' level) were all connected to either the pushbuttons or switches on the DE1 Board. Also, the clock lines of the two modules are not the same since the decimator and the square_wave are connected to the 50MHz clock and the audio interface is connected to a 24MHz clock. Finally, by toggling the switched of the note_number, different musical notes are produced and if we change octave, it will increase the pitch.

In the Flow Summary, it indicates that we used 1016 transistors for the simulation of the square wave (See Fig. 6). The largest propagation delay of the circuit is 9.839 ns. (See figure 7).

Figure 6: Flow Summary

Flow Status	Successful - Thu Nov 14 12:56:55 2013
Quartus II 64-Bit Version	9.1 Build 350 03/24/2010 SP 2 SJ Full Version
Revision Name	g19_lab4
Top-level Entity Name	g19_square_wave_generator1
Family	Cyclone II
Device	EP2C20F484C7
Timing Models	Final
Met timing requirements	Yes
Total logic elements	1,016 / 18,752 (5 %)
Total combinational functions	620 / 18,752 (3 %)
Dedicated logic registers	837 / 18,752 (4 %)
Total registers	837
Total pins	20 / 315 (6 %)
Total virtual pins	0
Total memory bits	0 / 239,616 (0 %)
Embedded Multiplier 9-bit elements	0 / 52 (0 %)
Total PLLs	0 / 4 (0 %)

Figure 7: Timing Analyzer

Timing Analyzer Summary

	Type	Slack	Required Time	Actual Time	From	To	From Clock	To Clock	Failed Paths
1	Worst-case tsu	N/A	None	7.616 ns	Reset	g19_decimator:inst1|c2d2[0]	--	clk	0
2	Worst-case tco	N/A	None	9.839 ns	g19_audio_interface:inst2|AUD_DACDAT	aud_dacdat	clk24	--	0
3	Worst-case th	N/A	None	0.573 ns	volume[22]	g19_decimator:inst1|x[22]	--	clk	0
4	Clock Setup: 'clk'	N/A	None	135.23 MHz (period = 7.395 ns)	g19_decimator:inst1|c0[2]	g19_decimator:inst1|c1[56]	clk	clk	0
5	Clock Setup: 'clk24'	N/A	None	208.12 MHz (period = 4.805 ns)	g19_audio_interface:inst2|clk_count[0]	g19_audio_interface:inst2|clk_count[1]	clk24	clk24	0
6	Total number of failed paths								0