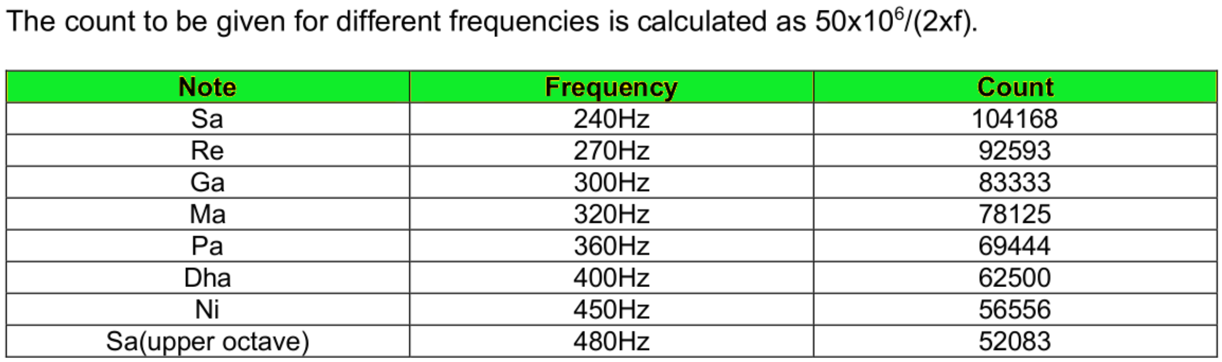
Experiment 5: Musical Notes

Prasann Viswanathan, Roll Number 190070047
EE-214, WEL, IIT Bombay
March 27, 2021

Overview of the experiment:

The purpose of the experiment was to play Musical notes’ octet on a speaker using VHDL code. We use a sequential style VHDL code to generate frequencies corresponding to musical notes on the Krypton board which in turn is connected to a speaker. The inputs are given using the switches on the board and a clock of frequency 50MHz. The output source powering the speaker is PIN_1 on Krypton board and LEDs to indicate which note is played. The frequencies for each note is created by making a simple counter using the clock divider technique. Each frequency has a different count which is calculated as 50MHz/2f. We use a testbench which tests the code by simulating the output frequency for each of the switch inputs. The frequencies can be verified in the RTL simulation by checking the output’s time periods. This verifies the functioning of the code on software. And, finally, the pin planning is done accordingly and svf is generated and parsed onto the Krypton board, the speaker is connected to the board through appropriate components and the hardware functioning is verified by checking the tone and LEDs for different switch inputs.

Approach to the experiment:



The count to be given for different frequencies is calculated as $50x10^6/(2xf)$.

Note	Frequency	Count
Sa	240Hz	104168
Re	270Hz	92593
Ga	300Hz	83333
Ma	320Hz	78125
Pa	360Hz	69444
Dha	400Hz	62500
Ni	450Hz	56556
Sa(upper octave)	480Hz	52083

Design document and VHDL code if relevant:

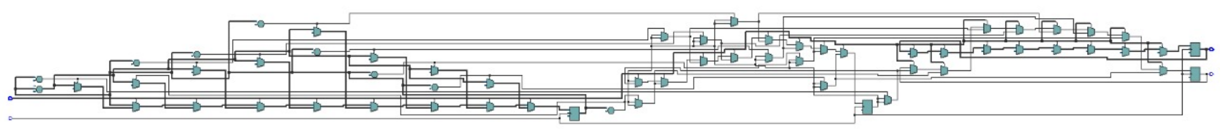
ToneGenerator_tb: Acts as Testbench.
ToneGenerator: Main logic, code template provided by TAs, some additions needed to be made.

Architecture of ToneGenerator is:

```
architecture behavioral of toneGenerator is
begin
process(clk)
variable count_sa1 : integer := 0;
variable sa1 : std_logic := '0';
begin
if(clk = '1' and clk' event) then
        if(switch(0) = '1') then
                if (count_sa1 = 104167) then--240Hz
                        count_sa1 := 1;
                        sa1 := not sa1;
                else
                        count_sa1 := count_sa1 + 1;
                end if;
                        toneOut <= sa1;
                        LED <= (0 => '1', others => '0');

        elsif(switch(1) = '1') then
                if (count_sa1 = 92593) then--270Hz
                        count_sa1 := 1;
                        sa1 := not sa1;
                else
                        count_sa1 := count_sa1 + 1;
                end if;
                        toneOut <= sa1;
                        LED <= (1 => '1', others => '0');

        elsif(switch(2) = '1') then
                if (count_sa1 = 83333) then--300Hz
                        count_sa1 := 1;
                        sa1 := not sa1;
                else
                        count_sa1 := count_sa1 + 1;
                end if;
                        toneOut <= sa1;
                        LED <= (2 => '1', others => '0');

        elsif(switch(3) = '1') then
                if (count_sa1 = 78125) then--320Hz
                        count_sa1 := 1;
                        sa1 := not sa1;
                else
                        count_sa1 := count_sa1 + 1;
                end if;
                        toneOut <= sa1;
                        LED <= (3 => '1', others => '0');

        elsif(switch(4) = '1') then
                if (count_sa1 = 69444) then--360Hz
                        count_sa1 := 1;
                        sa1 := not sa1;
                else
                        count_sa1 := count_sa1 + 1;
                end if;
                        toneOut <= sa1;
                        LED <= (4 => '1', others => '0');

        elsif(switch(5) = '1') then
                if (count_sa1 = 62500) then--400Hz
```

```
            count_sa1 := 1;
            sa1 := not sa1;
        else
            count_sa1 := count_sa1 + 1;
        end if;
            toneOut <= sa1;
            LED <= (5 => '1', others => '0');

    elsif(switch(6) = '1') then
        if (count_sa1 = 55556) then--450Hz
            count_sa1 := 1;
            sa1 := not sa1;
        else
            count_sa1 := count_sa1 + 1;
        end if;
            toneOut <= sa1;
            LED <= (6 => '1', others => '0');

    elsif(switch(7) = '1') then
        if (count_sa1 = 52083) then--480Hz
            count_sa1 := 1;
            sa1 := not sa1;
        else
            count_sa1 := count_sa1 + 1;
        end if;
            toneOut <= sa1;
            LED <= (7 => '1', others => '0');
    end if;
end if;
end process;
end behavioral;
```

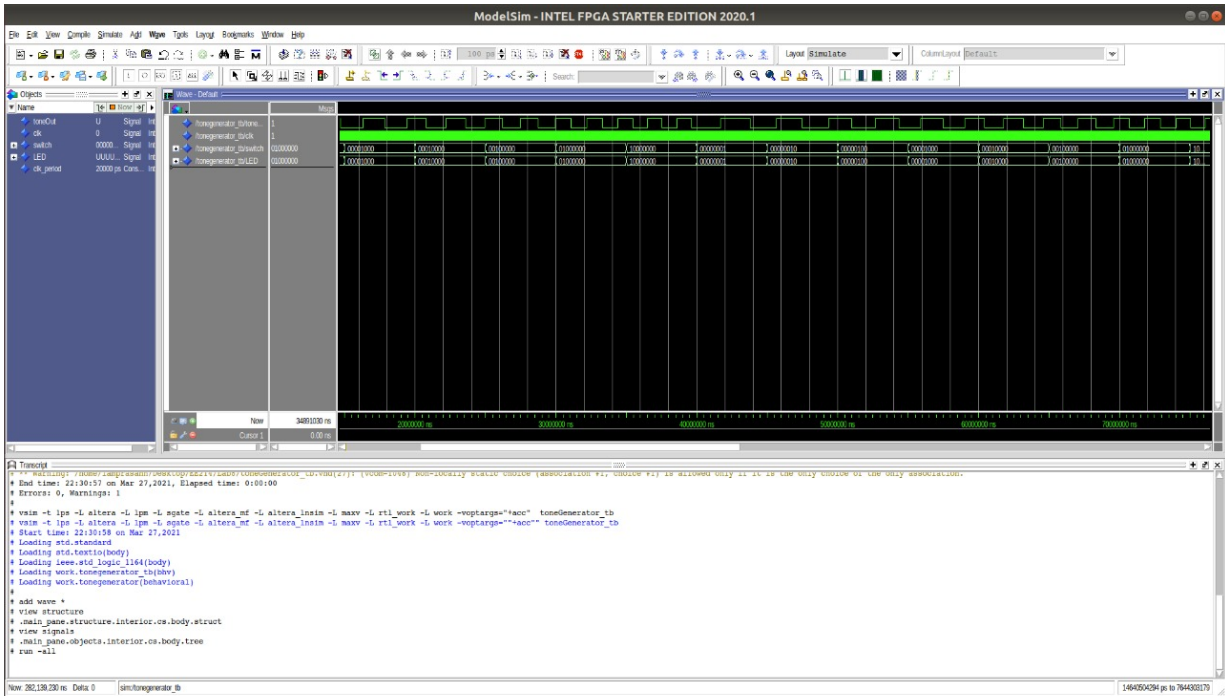
RTL View:



DUT Input/Output Format:

NA

RTL Simulation:



Name	Now		
toneOut	U	Signal	Int
clk	0	Signal	Int
switch	00000...	Signal	Int
LED	UUUU...	Signal	Int
clk_period	20000 ps	Cons...	Int

	Msgs
/tonegenerator_tb/tone...	1
/tonegenerator_tb/clk	1
/tonegenerator_tb/switch	01000000
/tonegenerator_tb/LED	01000000
Now	34891030 ns
Cursor 1	0.00 ns

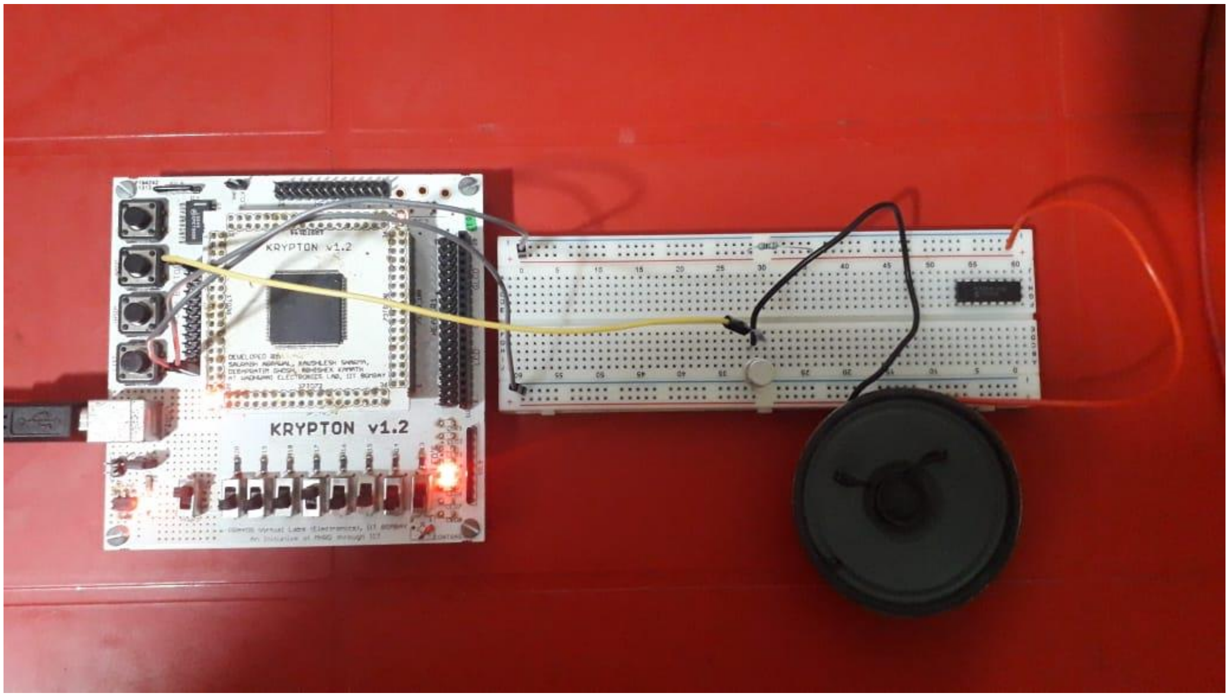
```
# ** Warning: /home/lamprasann/Desktop/EE214/Labs/toneGenerator_tb.vhd(27): (vcom-1048) Non-locally static choice (association #1, choice #1) is allowed only if it is the only choice of the only association.
# End time: 22:30:57 on Mar 27,2021, Elapsed time: 0:00:00
# Errors: 0, Warnings: 1
#
# vsim -t 1ps -L altera -L lpm -L sgate -L altera_mf -L altera_lnsim -L maxv -L rtl_work -L work -voptargs="+acc"  toneGenerator_tb
# vsim -t 1ps -L altera -L lpm -L sgate -L altera_mf -L altera_lnsim -L maxv -L rtl_work -L work -voptargs=""+acc"" toneGenerator_tb
# Start time: 22:30:58 on Mar 27,2021
# Loading std.standard
# Loading std.textio(body)
# Loading ieee.std_logic_1164(body)
# Loading work.tonegenerator_tb(bhv)
# Loading work.tonegenerator(behavioral)
#
# add wave *
# view structure
# .main_pane.structure.interior.cs.body.struct
# view signals
# .main_pane.objects.interior.cs.body.tree
# run -all
```

Now: 282,139,230 ns Delta: 0   sim:/tonegenerator_tb   14640504294 ps to 7644303179

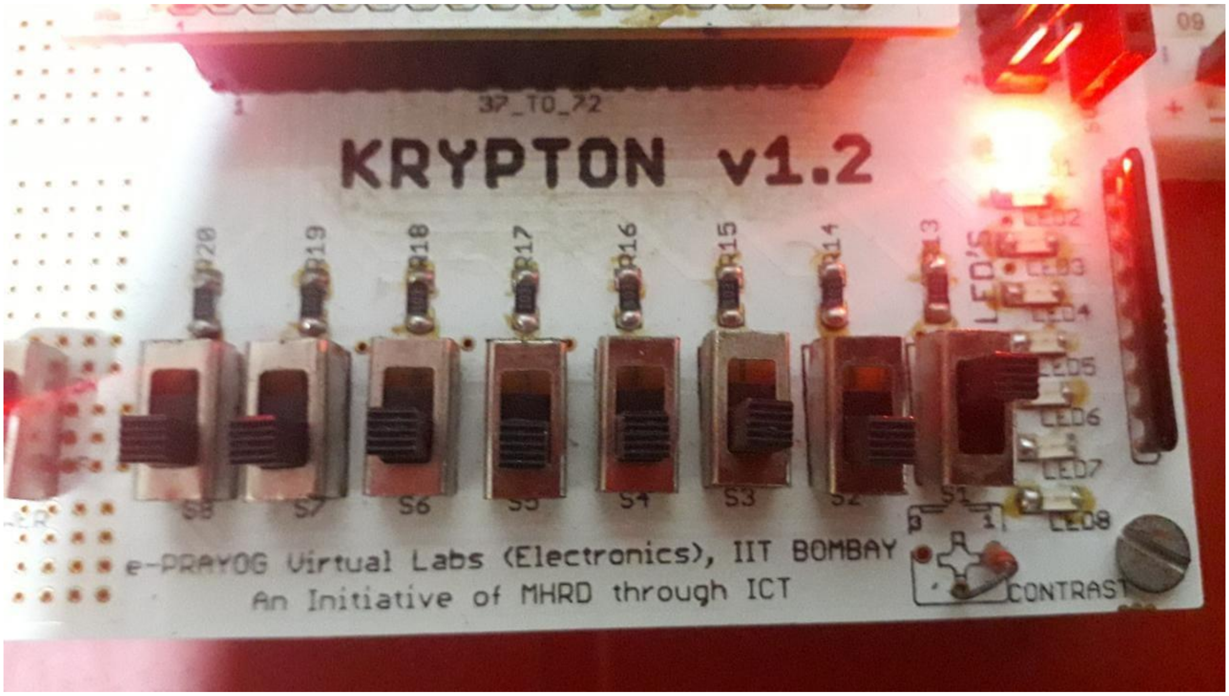
Gate-level Simulation:

NA

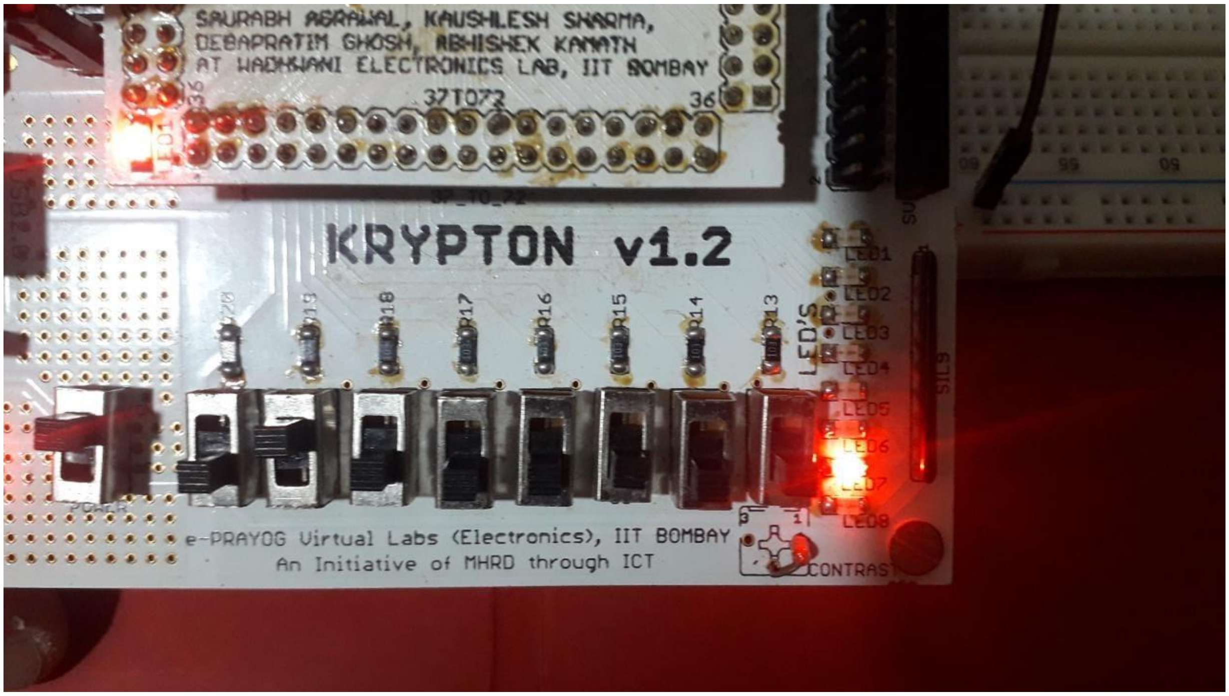
Krypton board*:



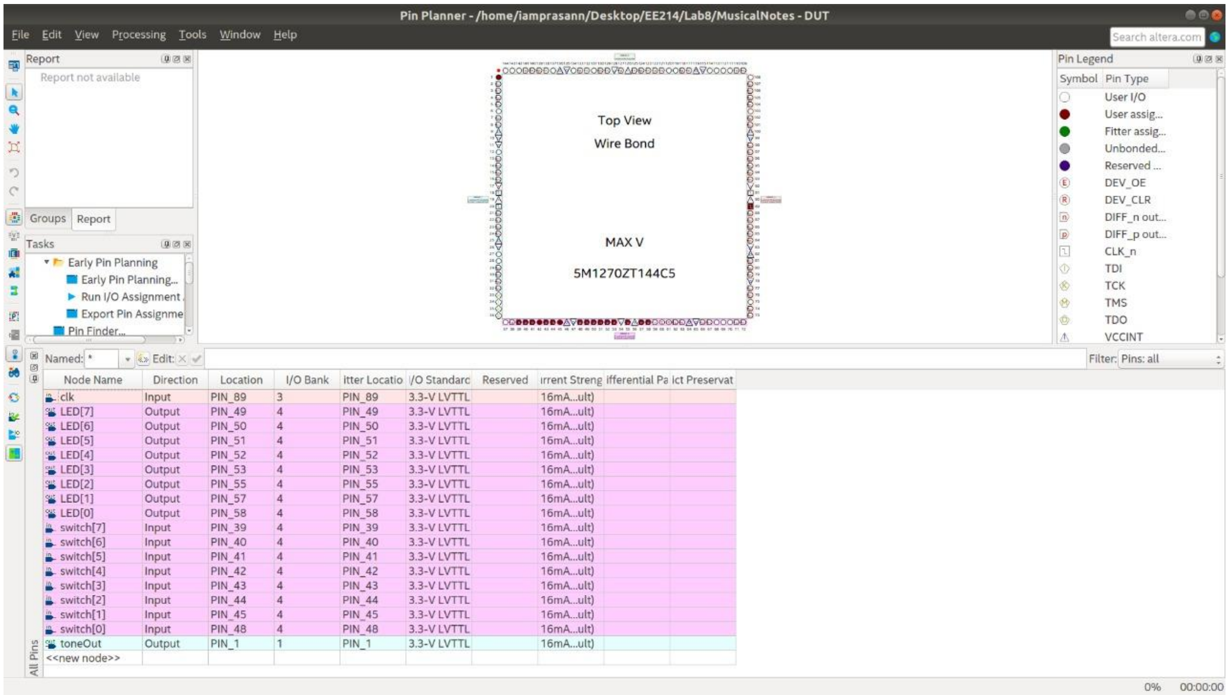
Krypton board connected to speaker with proper connections to play music



Switch 1 corresponds to LED1


SAURABH AGRAWAL, KAUSHLESH SHARMA,
DEBAPRATIM GHOSH, ABHISHEK KAMATH
AT WADHWANI ELECTRONICS LAB, IIT BOMBAY
KRYPTON v1.2
LED'S
LED1
LED2
LED3
LED4
LED5
LED6
LED7
LED8
POWER
e-PRAYOG Virtual Labs (Electronics), IIT BOMBAY
An Initiative of MHRD through ICT
CONTRAST

Switch 7 corresponds to LED 7.



Node Name	Direction	Location	I/O Bank	itter Locatio	/O Standarc	Reserved	irrent Streng	ifferential Pa	ict Preservat
clk	Input	PIN_89	3	PIN_89	3.3-V LVTTL		16mA...ult)		
LED[7]	Output	PIN_49	4	PIN_49	3.3-V LVTTL		16mA...ult)		
LED[6]	Output	PIN_50	4	PIN_50	3.3-V LVTTL		16mA...ult)		
LED[5]	Output	PIN_51	4	PIN_51	3.3-V LVTTL		16mA...ult)		
LED[4]	Output	PIN_52	4	PIN_52	3.3-V LVTTL		16mA...ult)		
LED[3]	Output	PIN_53	4	PIN_53	3.3-V LVTTL		16mA...ult)		
LED[2]	Output	PIN_55	4	PIN_55	3.3-V LVTTL		16mA...ult)		
LED[1]	Output	PIN_57	4	PIN_57	3.3-V LVTTL		16mA...ult)		
LED[0]	Output	PIN_58	4	PIN_58	3.3-V LVTTL		16mA...ult)		
switch[7]	Input	PIN_39	4	PIN_39	3.3-V LVTTL		16mA...ult)		
switch[6]	Input	PIN_40	4	PIN_40	3.3-V LVTTL		16mA...ult)		
switch[5]	Input	PIN_41	4	PIN_41	3.3-V LVTTL		16mA...ult)		
switch[4]	Input	PIN_42	4	PIN_42	3.3-V LVTTL		16mA...ult)		
switch[3]	Input	PIN_43	4	PIN_43	3.3-V LVTTL		16mA...ult)		
switch[2]	Input	PIN_44	4	PIN_44	3.3-V LVTTL		16mA...ult)		
switch[1]	Input	PIN_45	4	PIN_45	3.3-V LVTTL		16mA...ult)		
switch[0]	Input	PIN_48	4	PIN_48	3.3-V LVTTL		16mA...ult)		
toneOut	Output	PIN_1	1	PIN_1	3.3-V LVTTL		16mA...ult)		
<<new node>>									

Pin planning shown

Observations*:

Sound generated matches that of Indian Ra Ga tones
https://drive.google.com/file/d/1iKI46898pqyURBS7piD7kgGFXEIoL_hT/view?usp=sharing
Can view the video of the tones and LEDs at this link. (Better than Moodle Placeholder)

References:

TA provided codes.