Experiment 5 : Tone Generation

Badal Varshney, Roll Number 19D070015

EE-214, WEL, IIT Bombay

March 25, 2021

## Overview of the experiment:

|  |
| --- |
| In this experiment, we will generate the seven major notes in the Indian classical music named sa, re, ga, ma, pa, dha, ni and Sa (upper octave). we will use the 8 slide switches on the krypton Board to play these notes. |

## Approach to the experiment:

|  |
| --- |
| Approach was to make separate counts for each tone and as per counts we get from clock divider formula we play nth tones on speaker in sync with switches on krypton board. |

## Design document and VHDL code if relevant:

|  |
| --- |
| **Code of tone generator**-  library ieee;  use ieee.std\_logic\_1164.all;  library ieee;  use ieee.std\_logic\_1164.all;  library work;  use work.all;  entity toneGenerator is  port (toneOut : out std\_logic; --this pin will give your notes output  clk : in std\_logic;  LED : out std\_logic\_vector(7 downto 0);  switch : in std\_logic\_vector(7 downto 0)  );  end entity toneGenerator;    architecture bev of toneGenerator is  begin  process(clk)  variable count\_sa1, count\_re, count\_ga, count\_ma,count\_pa, count\_dha, count\_ni, count\_sa2: integer range 0 to 1E8 := 1;  variable sa1, re, ga, ma, pa, dha, ni, sa2: std\_logic := '1';  begin  if (clk = '1' and clk' event) then  if (switch(0) = '1') then  if (count\_sa1 = 104168) 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\_re = 92594) then --270Hz  count\_re := 1;  re := not re;  else  count\_re := count\_re + 1;  end if;  toneOut <= re;  LED <= (1 => '1', others => '0');  elsif (switch(2) = '1') then  if (count\_ga = 83334) then--300Hz  count\_ga := 1;  ga := not ga;  else  count\_ga := count\_ga + 1;  end if;  toneOut <= ga;  LED <= (2 => '1', others => '0');  elsif (switch(3) = '1') then  if (count\_ma = 78126) then--320Hz  count\_ma := 1;  ma := not ma;  else  count\_ma := count\_ma + 1;  end if;  toneOut <= ma;  LED <= (3 => '1', others => '0');  elsif (switch(4) = '1') then  if (count\_pa = 69445) then--360Hz  count\_pa := 1;  pa := not pa;  else  count\_pa := count\_pa + 1;  end if;  toneOut <= pa;  LED <= (4 => '1', others => '0');  elsif (switch(5) = '1') then  if (count\_dha = 62501) then--400Hz  count\_dha := 1;  dha := not dha;  else  count\_dha := count\_dha + 1;  end if;  toneOut <= dha;  LED <= (5 => '1', others => '0');  elsif (switch(6) = '1') then  if (count\_ni = 55557) then--450Hz  count\_ni := 1;  ni := not ni;  else  count\_ni := count\_ni + 1;  end if;  toneOut <= ni;  LED <= (6 => '1', others => '0');  elsif (switch(7) = '1') then  if (count\_sa2 = 52084) then--480Hz  count\_sa2 := 1;  sa2 := not sa2;  else  count\_sa2 := count\_sa2 + 1;  end if;  toneOut <= sa2;  LED <= (7 => '1', others => '0');  end if;  end if;  end process;    end bev;  **Code of Testbench-**  LIBRARY IEEE;  USE IEEE.STD\_LOGIC\_1164.ALL;    entity toneGenerator\_tb is  end entity toneGenerator\_tb;    architecture bhv of toneGenerator\_tb is  component toneGenerator is  port (toneOut : out std\_logic;  clk : in std\_logic;  LED : out std\_logic\_vector(7 downto 0);  switch : in std\_logic\_vector(7 downto 0));  end component;    signal toneOut : std\_logic := '0';  signal clk: std\_logic := '0';  signal switch, LED : std\_logic\_vector(7 downto 0) := (others => '0');  constant clk\_period : time := 20 ns;  begin  dut\_instance: toneGenerator port map(toneOut, clk, LED, switch);  clk <= not clk after clk\_period/2 ;  stim\_proc: process  begin  for i in 0 to 7 loop --This loop will run by making only 1 switch on at a time  switch <= (i => '1', others => '0');  wait for 5 ms;  end loop;  end process;  end bhv;  -- Run the testbench for 40 ms |

## RTL View:

|  |
| --- |
| Attach screen-shot of the RTL view generated by Quartus. |

## DUT Input/Output Format:

|  |
| --- |
| Mention the format (LSB/MSB of input and output) and few test cases from trace-file. |

## RTL Simulation:

|  |
| --- |
| Attach the clearly visible screen-shot of RTL simulation waveforms. |

## Gate-level Simulation:

|  |
| --- |
| Attach the clearly visible screen-shot of Gate-level Simulation. |

## Krypton board\*:

|  |
| --- |
| Map the logic circuit to the Krypton board and attach the images of the pin assignment and output observed on the board (switches/LEDs).  **Pin Planner-**    THE Video Link contain video of speaker working-  <https://drive.google.com/file/d/1akHN-pStP0l90ynR6yold0F2ZlTazn-c/view?usp=sharing>  LED4 and switch4 on(ma tone)- |

## Observations\*:

|  |
| --- |
| The results were expected and the Krypton Board worked well and all the test cases were passed. |

## References:

|  |
| --- |
| You may include the references if any. |

\* To be submitted after the tutorial on ”Using Krypton.