Circuit Simulation Project

https://esim.fossee.in/circuit-simulation-project

Name of the participant :

SAI AUCHITYA BUSSA

Title of the circuit :

GRAY CODE COUNTER

Theory/Description :

Gray code is a kind of binary number system where only one bit will change at a time. Today gray code is widely used in the digital world. It will be helpful for error correction and signal transmission. The Gray counter is also useful in design and verification in the VLSI domain.

A Gray Code encodes integers as sequences of bits with the property that the representations of adjacent integers differ in exactly one binary position.

This design code has two inputs, clock and reset signals and one 8 bit output that will generate gray code.

First, if the rstn signal is high, then the output will be zero, and as soon as rstn goes low, on the rising edge of clk, the design will generate a eight-bit gray code and continue to generate at every rising edge of clk signal.

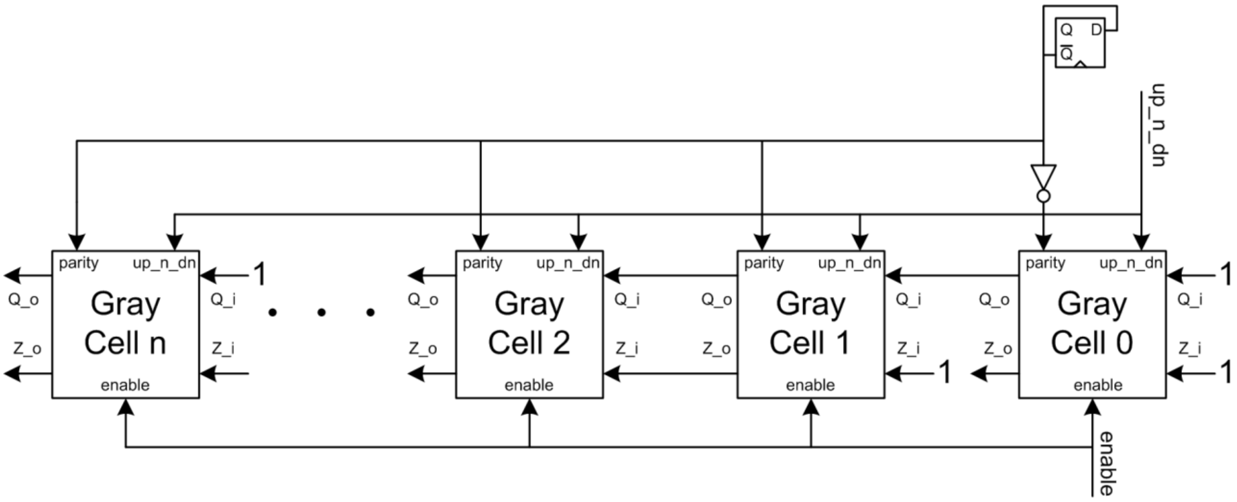
This design code can be upgraded and put binary numbers as input, and this design will work as a binary to gray code converter.

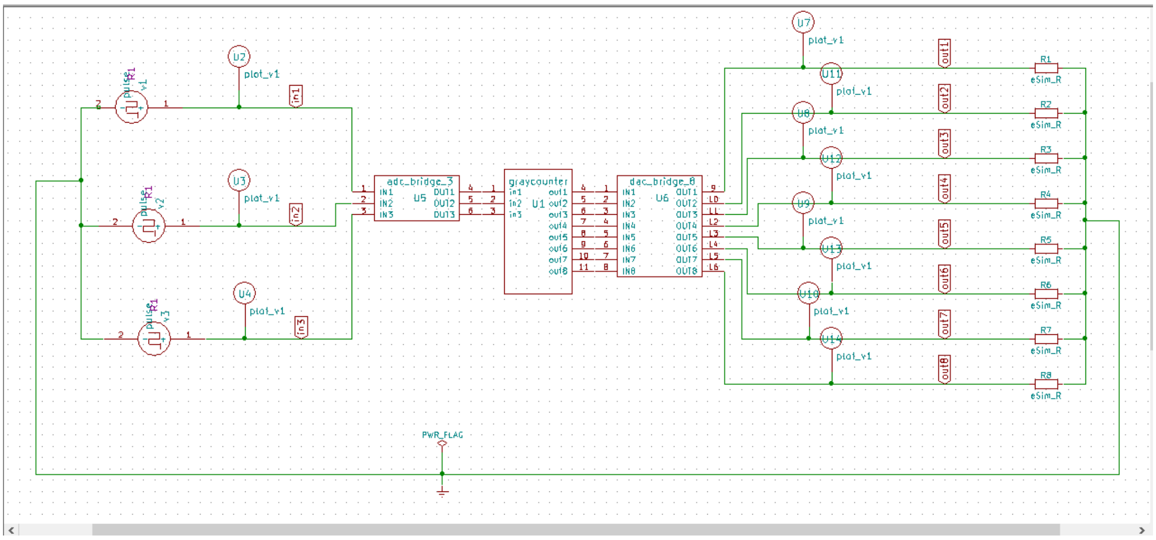
VHDL CODE:

```
library ieee;

use ieee.std_logic_1164.all;

use ieee.numeric_std.all;
```

```
entity graycounter is

        port

        (

                clk         : in std_logic;

                reset       : in std_logic;

                enable      : in std_logic;

                gray_count : out std_logic_vector(7 downto 0)

        );


end graycounter;

architecture rtl of graycounter is


        signal q  : std_logic_vector (8 downto 0);


        signal no_ones_below  : std_logic_vector (8 downto 0);


        signal q_msb : std_logic;


begin


        q_msb <= q(7) or q(8);


        process(clk, reset, enable)
        begin


                if(reset = '1') then



                        q(0) <= '1';
                        q(8 downto 1) <= (others => '0');


                elsif(rising_edge(clk) and enable='1') then



                        q(0) <= not q(0);


                        for i in 1 to 8 loop
                q(i) <= q(i) xor (q(i-1) and no_ones_below(i-1));

            end loop;

            q(8) <= q(8) xor (q_msb and no_ones_below(7));

        end if;


    end process;
    no_ones_below(0) <= '1';
    process(q, no_ones_below)
    begin
        for j in 1 to 8 loop
            no_ones_below(j) <= no_ones_below(j-1) and not q(j-1);
        end loop;
    end process;
    gray_count <= q(8 downto 1);


end rtl;
```

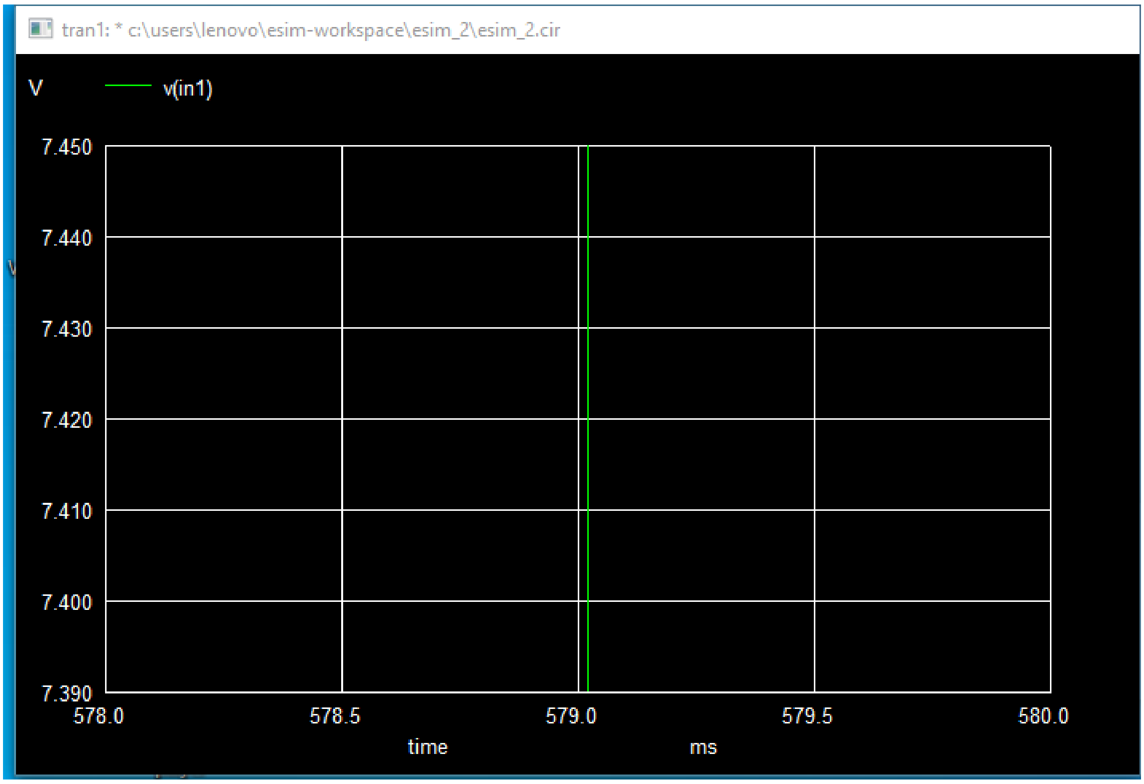
Circuit Diagram(s) :

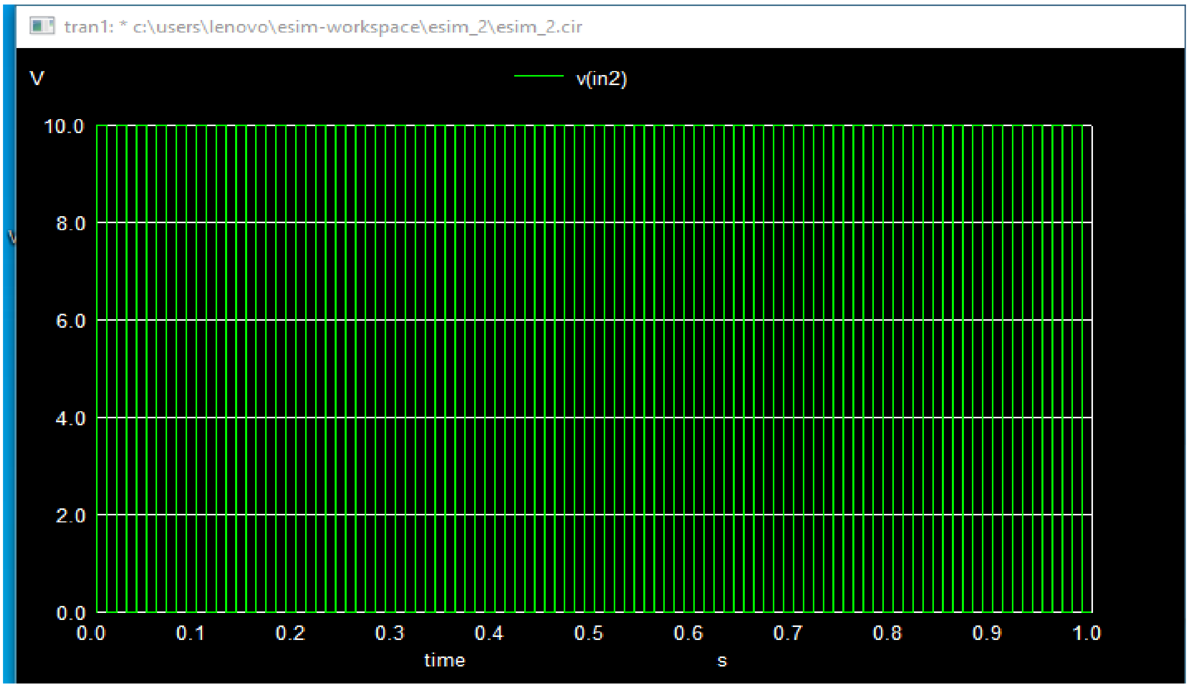


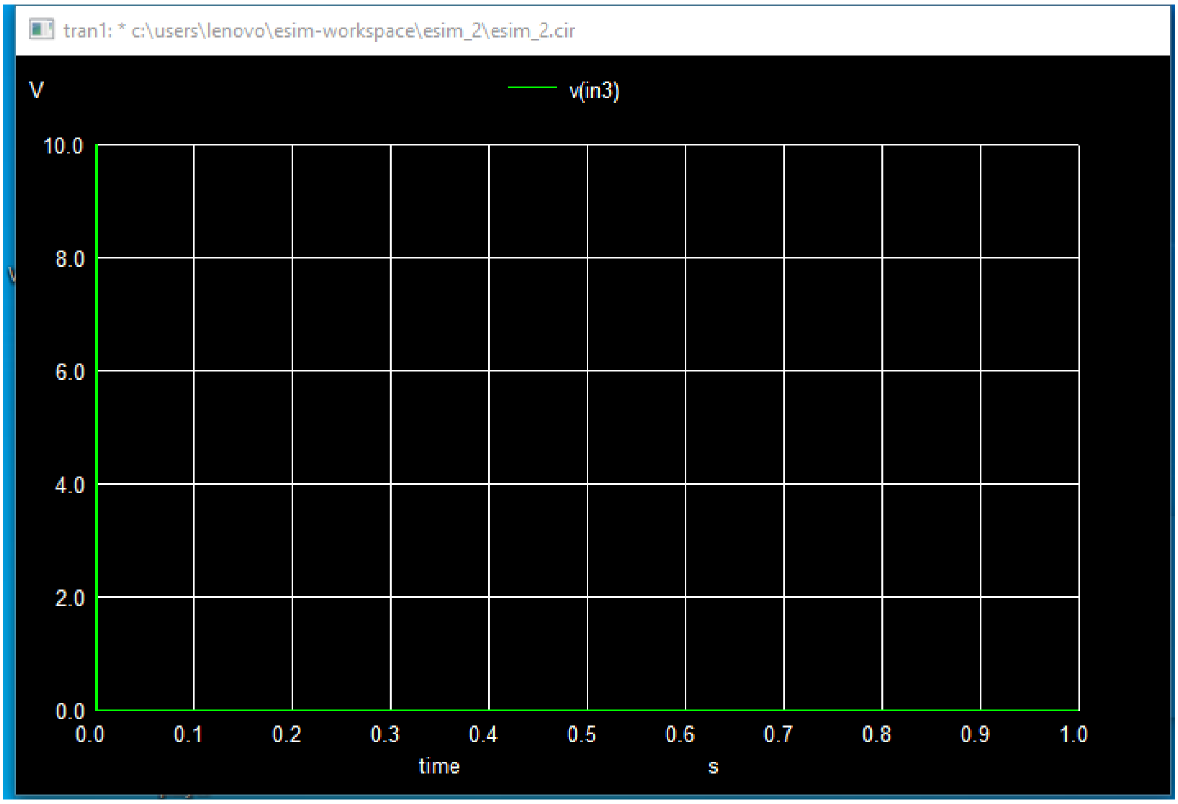


Results (Input, Output waveforms and/or Multimeter readings) :

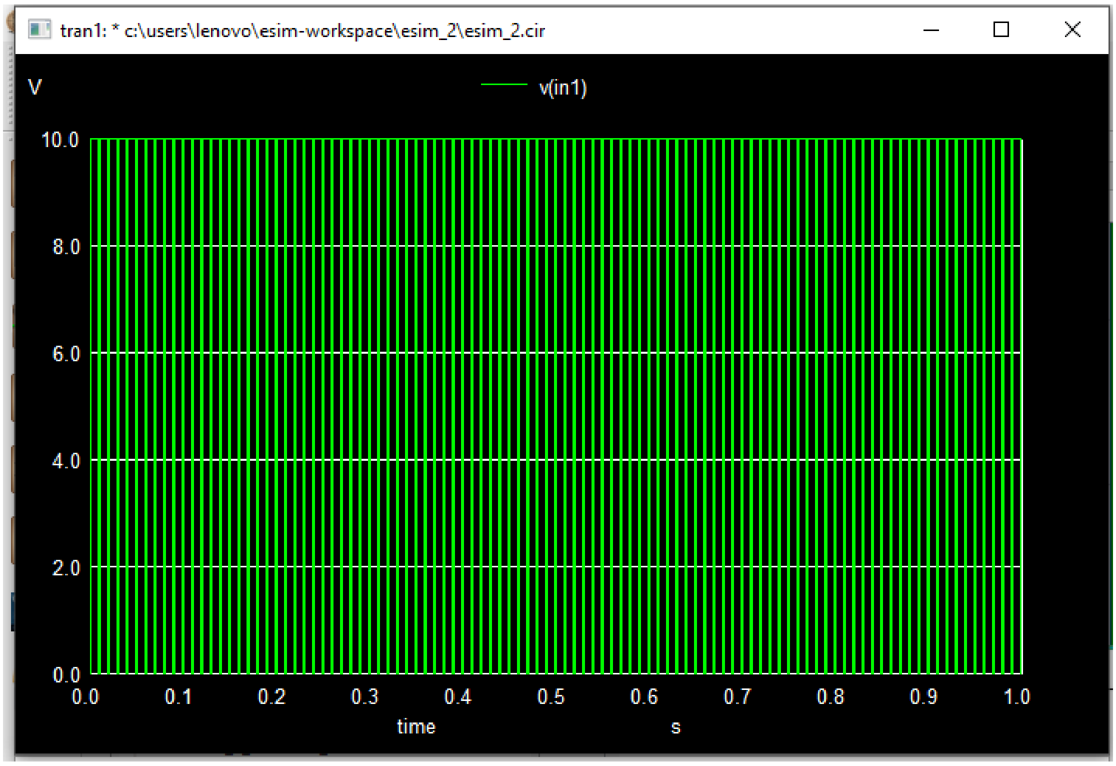
INPUTS:


tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir

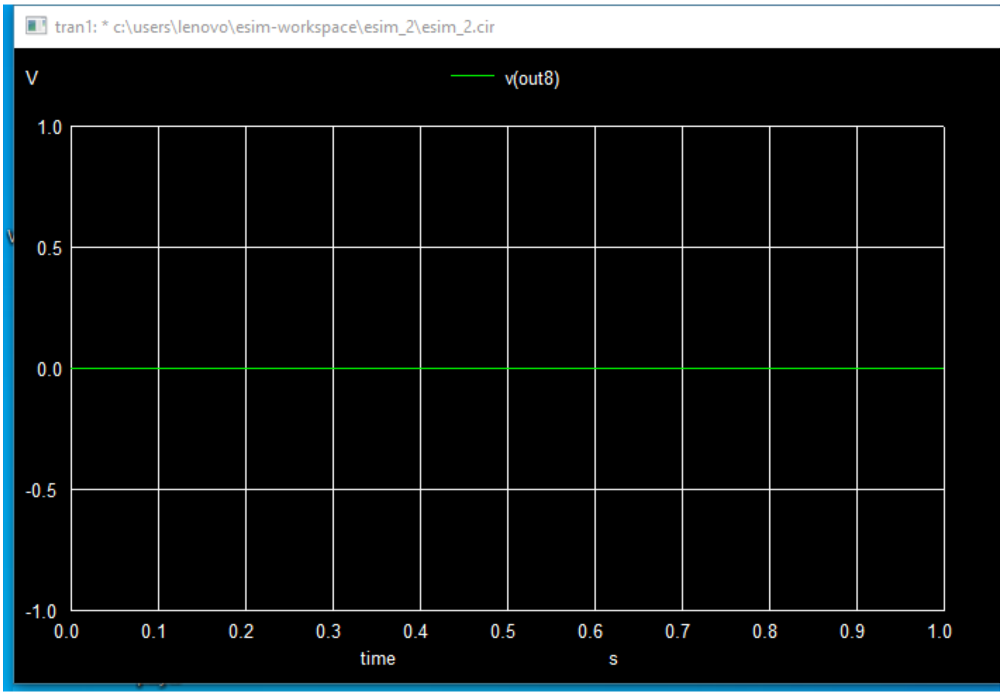

tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(in2)
10.0
8.0
6.0
4.0
2.0
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s

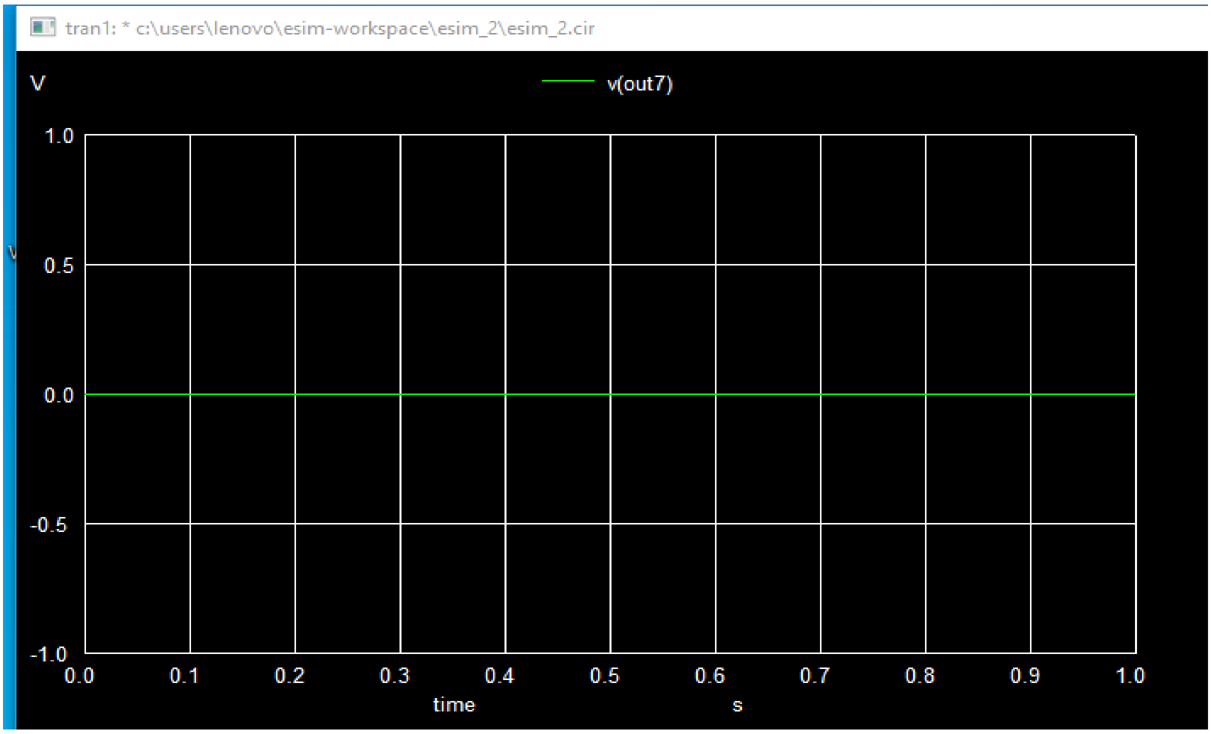

tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(in3)
10.0
8.0
6.0
4.0
2.0
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s

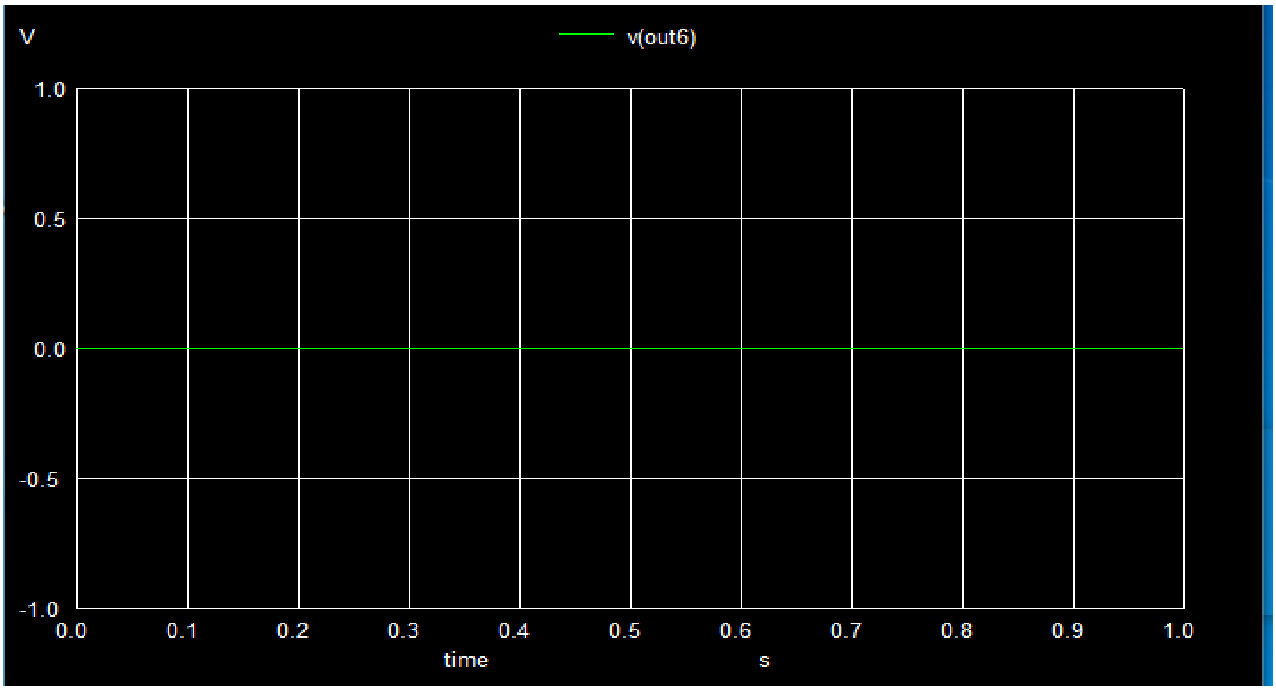
PULSE:

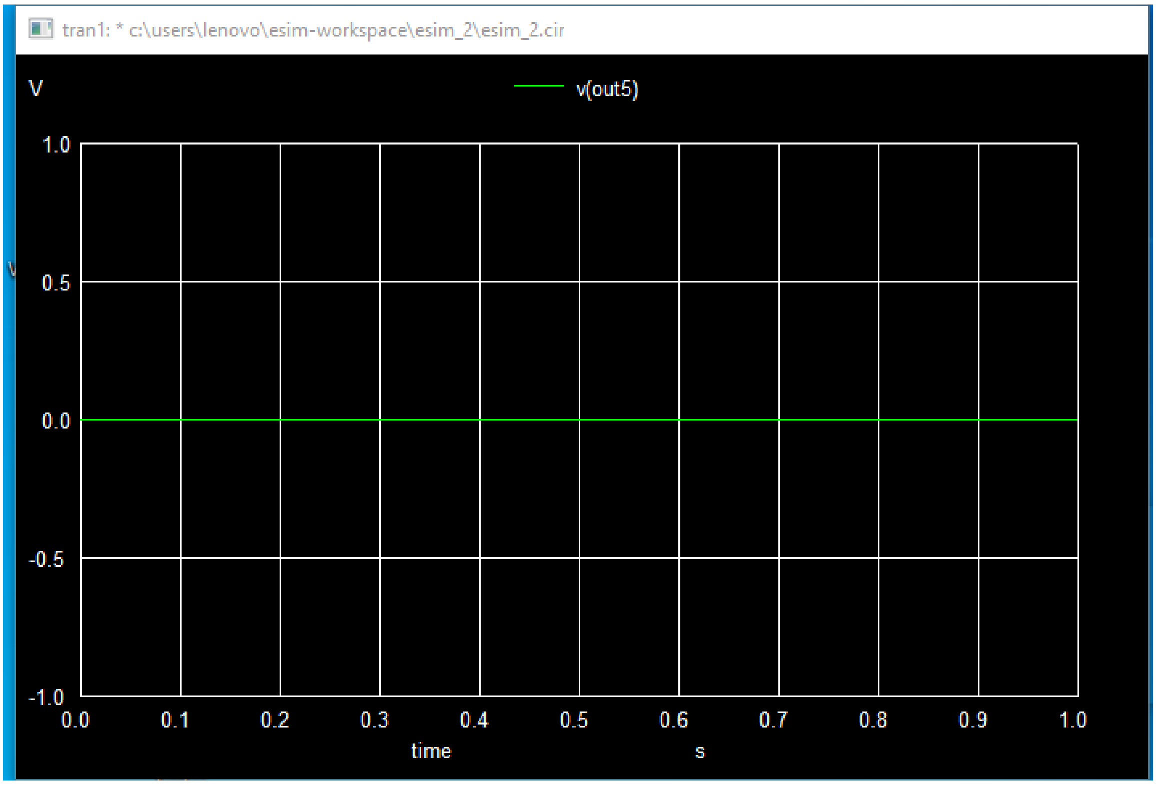

tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(in1)
10.0
8.0
6.0
4.0
2.0
0.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s

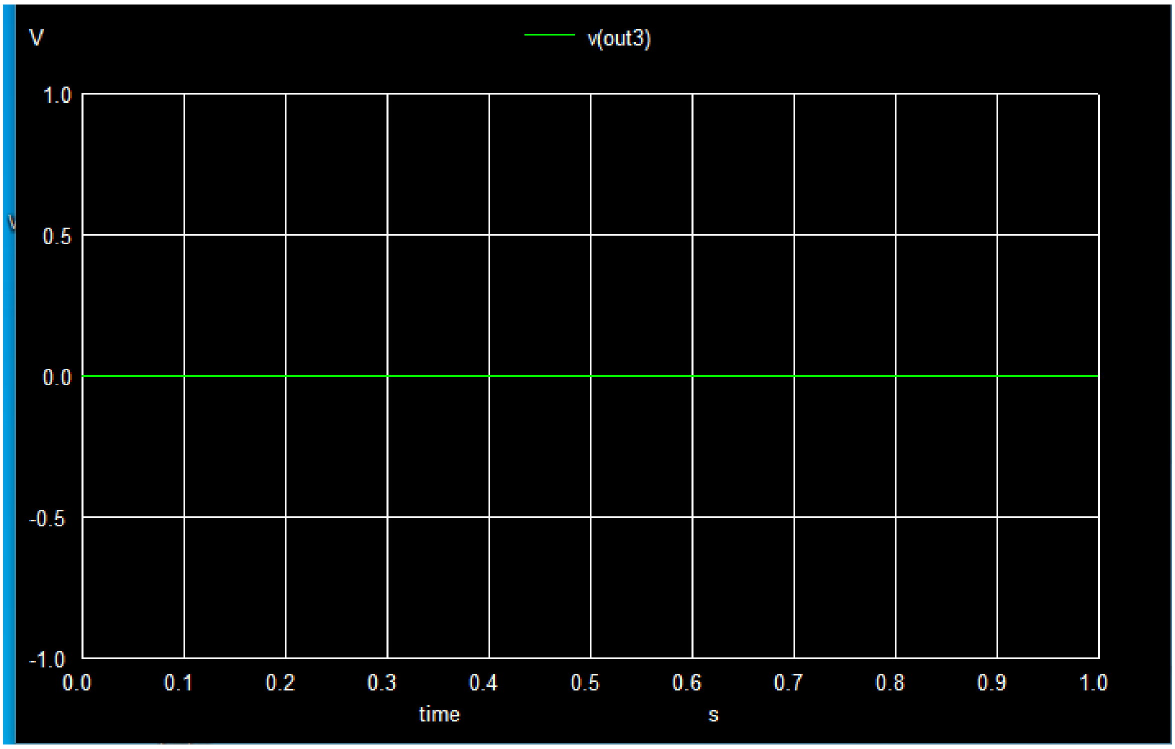
OUTPUTS:

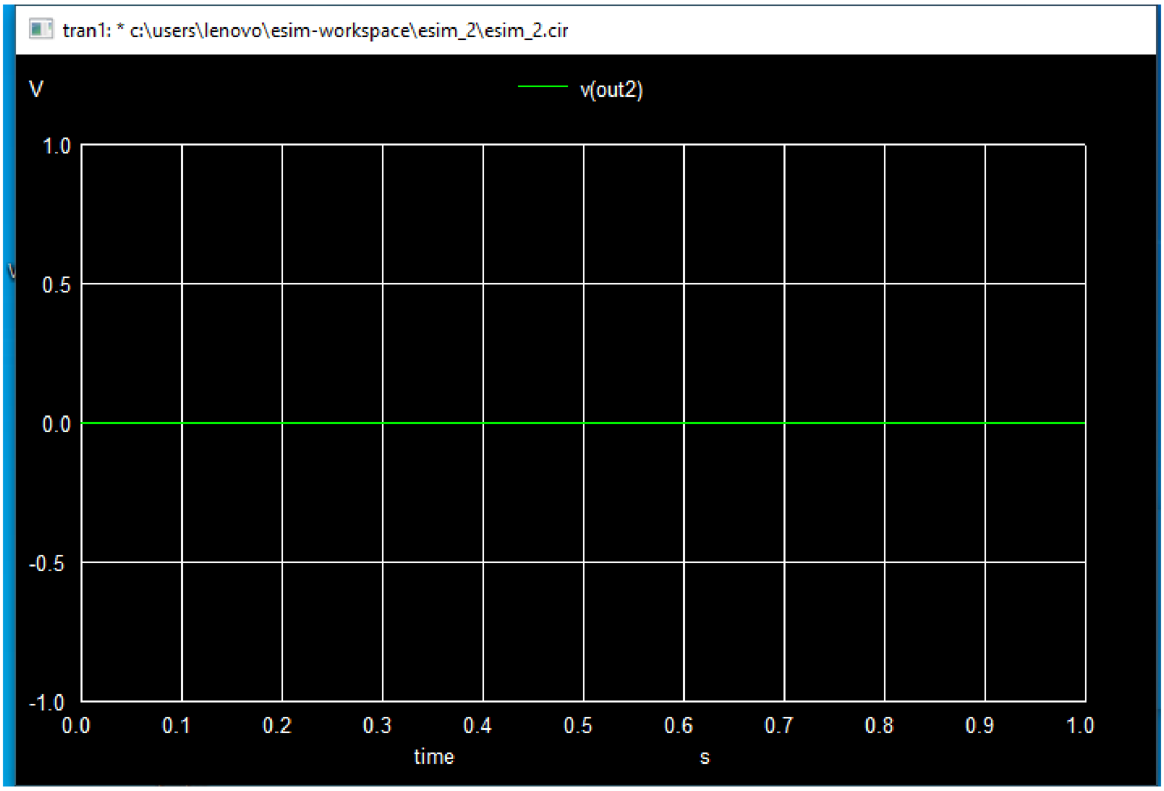

tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(out8)
1.0
0.5
0.0
-0.5
-1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s


tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(out7)
1.0
0.5
0.0
-0.5
-1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s

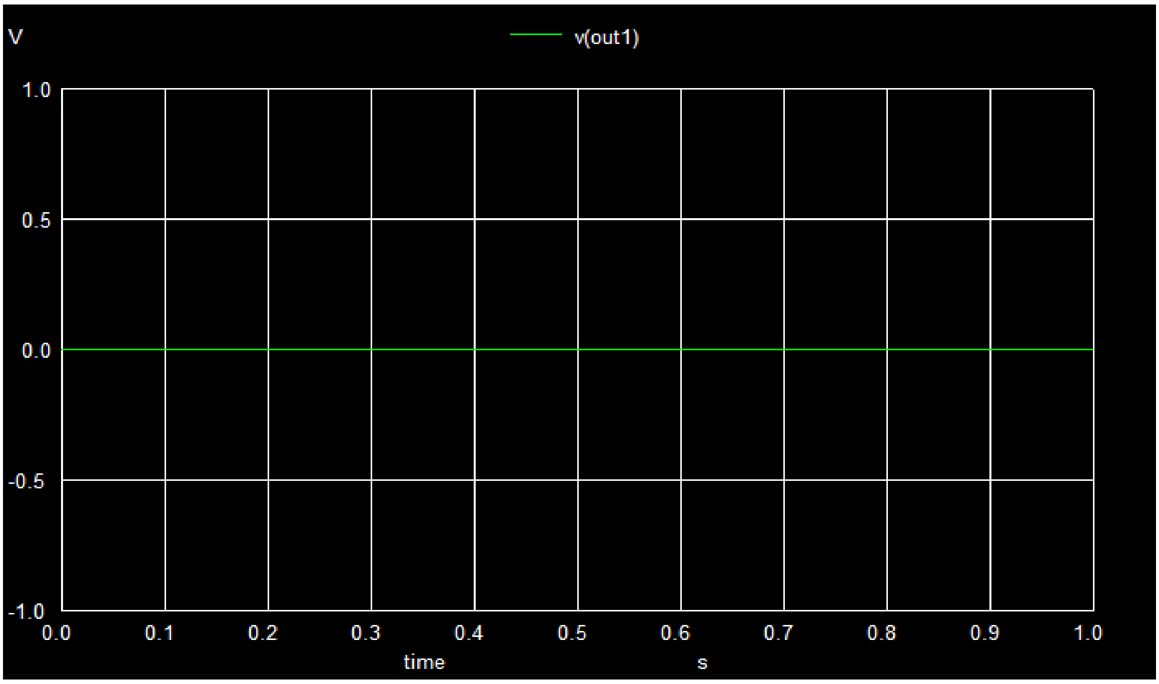

V
v(out6)
1.0
0.5
0.0
-0.5
-1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s


tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir
V
v(out5)
1.0
0.5
0.0
-0.5
-1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s


V
v(out3)
1.0
0.5
0.0
-0.5
-1.0
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
time
s



tran1: * c:\users\lenovo\esim-workspace\esim_2\esim_2.cir



Source/Reference(s) :

http://www.asic-world.com/examples/vhdl/gray.html