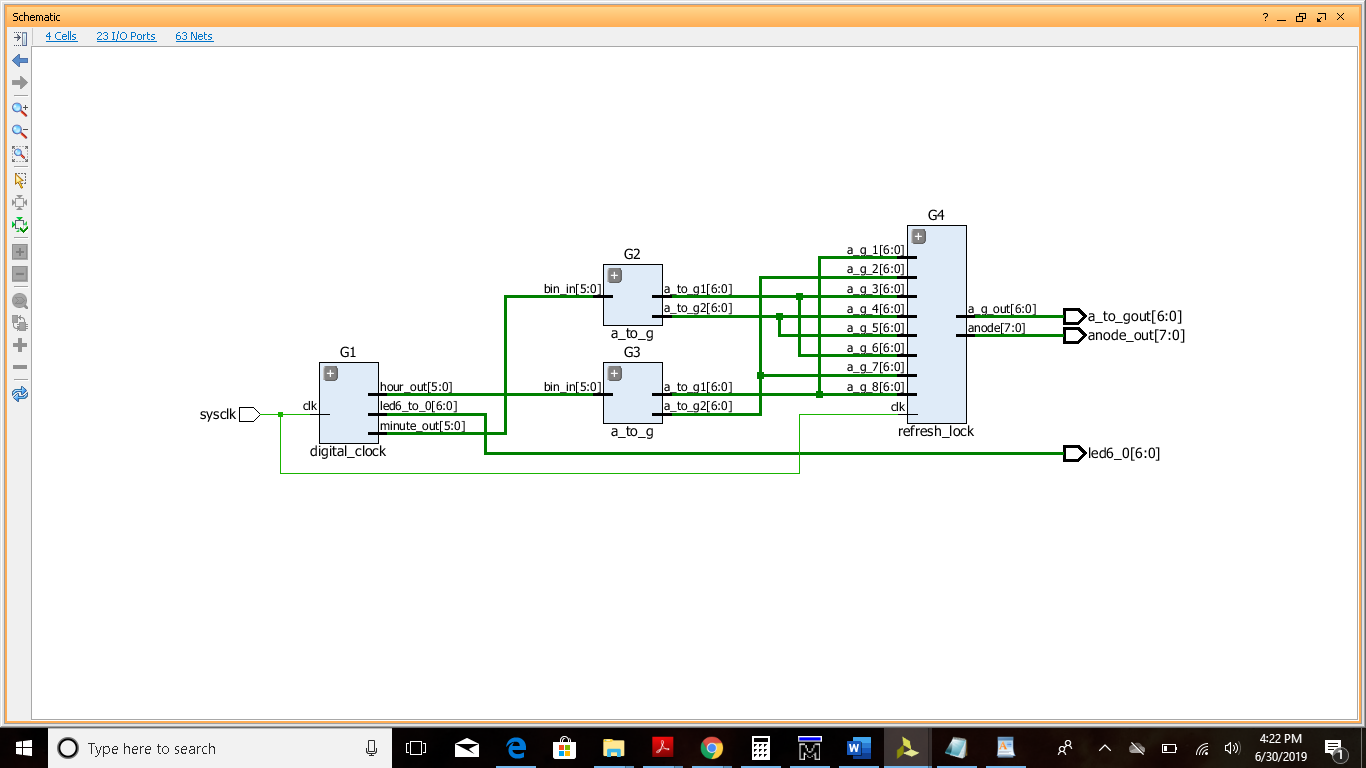
**Project-LAB4**

**The Digital clock**

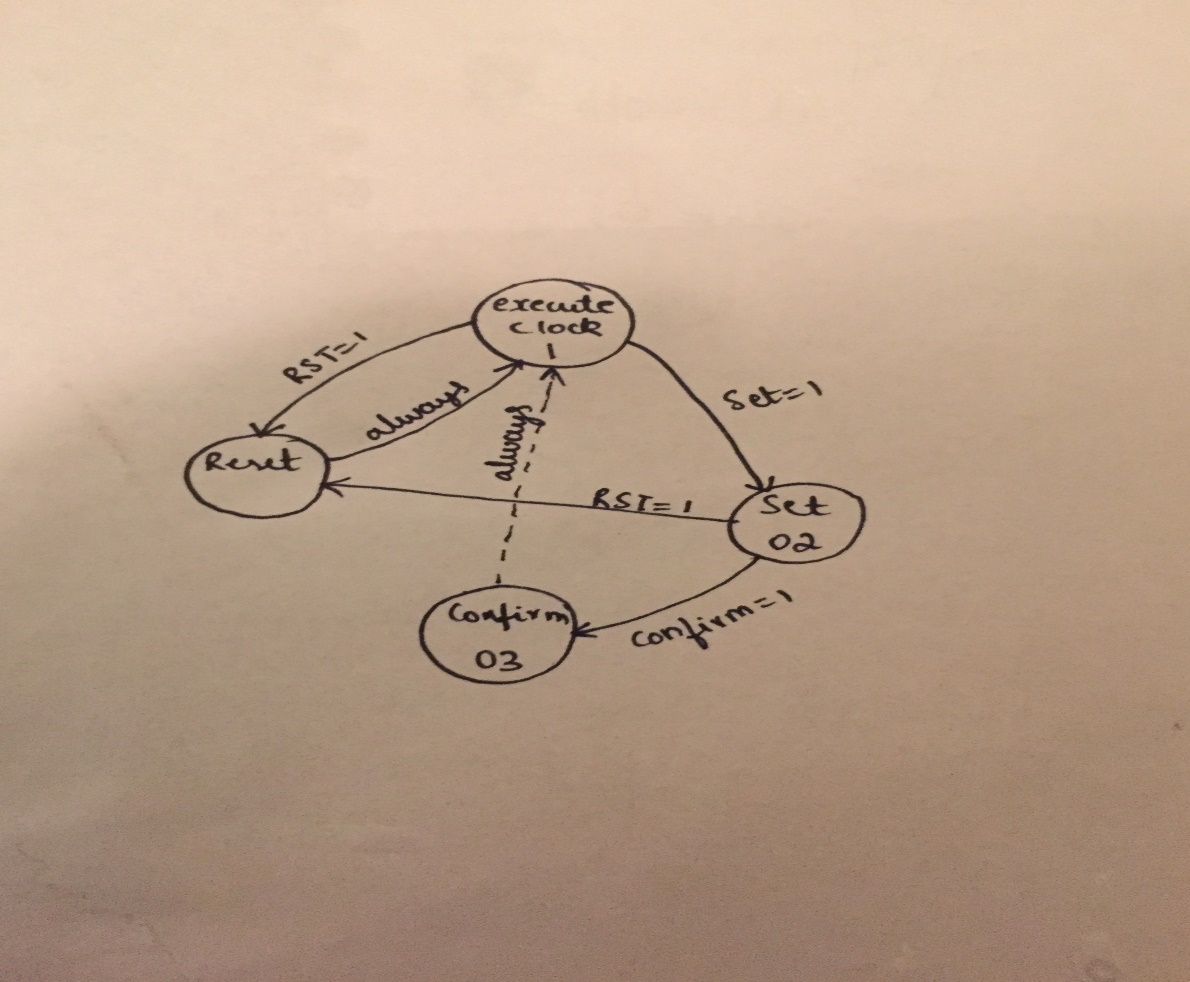
**Name: Sharath Venkatesh**

**Questions**

**Based on the above project design requirements, how will you design this digital clock system?** **Please draw below a block diagram for the subcomponents needed for this system; and explain briefly the functionality of each block/subsystem.**

****

**Additionally, what will be the state diagram describing the transition of states among the above-mentioned three states and their corresponding input/outputs?**

****

**Per your design, what is your VHDL entity code for this digital clock system? Please insert below the source code for the VHDL entity file(s) and describe the functionality of each VHDL entity file included in your project.**

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity full\_system is

port( sysclk :in std\_logic;

a\_to\_gout : out std\_logic\_vector(6 downto 0);

anode\_out : out std\_logic\_vector(7 downto 0);

led6\_0: out std\_logic\_vector(6 downto 0));

end full\_system;

architecture structural of full\_system is

-----------mux---------------------

component refresh\_lock is

Port (a\_g\_1,a\_g\_2,a\_g\_3,a\_g\_4,a\_g\_5,a\_g\_6,a\_g\_7,a\_g\_8:in std\_logic\_vector(6 downto 0);

clk:in std\_logic;

a\_g\_out:out std\_logic\_vector(6 downto 0);

anode:out std\_logic\_vector(7 downto 0));

end component refresh\_lock;

----------digital clock------------

component digital\_clock is

Port (clk : in std\_logic;

minute\_out : out std\_logic\_vector(5 downto 0):="000000";

hour\_out: out std\_logic\_vector(5 downto 0):="000000";

led6\_to\_0 : out std\_logic\_vector(6 downto 0):="0000000");

end component digital\_clock;

----------a to g-----------------

component a\_to\_g is

Port (bin\_in : in std\_logic\_vector(5 downto 0):="000000";

a\_to\_g1,a\_to\_g2: out std\_logic\_vector(6 downto 0));

end component a\_to\_g;

Signal S3,S4,S5,S6 : std\_logic\_vector(6 downto 0);

Signal S1,S2 : std\_logic\_vector(5 downto 0);

begin

G1: digital\_clock port map( clk=>sysclk,minute\_out => S1,hour\_out => S2,led6\_to\_0 =>led6\_0);

G2: a\_to\_g port map(bin\_in => S1,a\_to\_g1 => S5,a\_to\_g2 => S6);---minute

G3: a\_to\_g port map(bin\_in => S2,a\_to\_g1 => S3,a\_to\_g2 => S4);---hr

G4: refresh\_lock port map(a\_g\_1 => S3,a\_g\_2 => S4 ,a\_g\_3 => S5,a\_g\_4 => S6 ,a\_g\_5 =>S6 ,a\_g\_6 => S5 ,a\_g\_7=> S4,a\_g\_8 => S3,clk => sysclk,a\_g\_out => a\_to\_gout,anode => anode\_out);

end structural;

---------------mux------------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.std\_logic\_unsigned.all;

entity refresh\_lock is

Port (a\_g\_1,a\_g\_2,a\_g\_3,a\_g\_4,a\_g\_5,a\_g\_6,a\_g\_7,a\_g\_8:in std\_logic\_vector(6 downto 0);

clk:in std\_logic;

a\_g\_out:out std\_logic\_vector(6 downto 0);

anode:out std\_logic\_vector(7 downto 0));

end refresh\_lock;

architecture Behavioral of refresh\_lock is

signal count :std\_logic\_vector(2 downto 0):="000";

signal count1:std\_logic\_vector(2 downto 0):="000";

signal count2 :std\_logic\_vector(27 downto 0):=x"0000000";

signal int\_clk: std\_logic :='0';

begin

counter:process(int\_clk) is

begin

if(rising\_edge(int\_clk)) then

if(count = "111")then

count<="000";

a\_g\_out<=a\_g\_8;

elsif(count ="000") then

count<=count+1;

a\_g\_out<=a\_g\_1;

elsif(count ="001") then

count<=count+1;

a\_g\_out<=a\_g\_2;

elsif(count ="010") then

count<=count+1;

a\_g\_out<=a\_g\_3;

elsif(count ="011") then

count<=count+1;

a\_g\_out<=a\_g\_4;

elsif(count ="100") then

count<=count+1;

a\_g\_out<=a\_g\_5;

elsif(count ="101") then

count<=count+1;

a\_g\_out<=a\_g\_6;

elsif(count ="110") then

count<=count+1;

a\_g\_out<=a\_g\_7;

end if;

end if;

end process counter;

anode\_counter:process(int\_clk) is

begin

if(rising\_edge(int\_clk)) then

if(count1 = "111")then

count1<="000";

anode<="01111111";

elsif(count1 ="000") then

count1<=count1+1;

anode<="11111110";

elsif(count ="001") then

count1<=count1+1;

anode<="11111101";

elsif(count1 ="010") then

count1<=count1+1;

anode<="11111011";

elsif(count1 ="011") then

count1<=count1+1;

anode<="11110111";

elsif(count1 ="100") then

count1<=count1+1;

anode<="11101111";

elsif(count1 ="101") then

count1<=count1+1;

anode<="11011111";

elsif(count1 ="110") then

count1<=count1+1;

anode<="10111111";

end if;

end if;

end process anode\_counter;

slow\_clock\_process: process(clk) is

begin

if(rising\_edge(clk)) then

if count2 < x"0000000" then

count2<=count2+1;

else

int\_clk<= not int\_clk;

count2<=(others => '0');

end if;

end if;

end process slow\_clock\_process;

end Behavioral;

---------------a\_to\_g----------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity a\_to\_g is

Port (bin\_in : in std\_logic\_vector(5 downto 0):="000000";

a\_to\_g1,a\_to\_g2: out std\_logic\_vector(6 downto 0));

end a\_to\_g;

architecture structural of a\_to\_g is

begin

process(bin\_in) is

begin

case bin\_in is

when "000000" =>

a\_to\_g1<="1000000";

a\_to\_g2<="1000000";--00

when "000001" =>

a\_to\_g1<="1000000";

a\_to\_g2<="1111001";--01

when "000010" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0100100";--02

when "000011" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0110000";--03

when "000100" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0011011";--04

when "000101" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0010010";--05

when "000110" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0000010";--06

when "000111" =>

a\_to\_g1<="1000000";

a\_to\_g2<="1111000";--07

when "001000" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0000000";--08

when "001001" =>

a\_to\_g1<="1000000";

a\_to\_g2<="0011000";--09

when "001010" =>

a\_to\_g1<="1111001";

a\_to\_g2<="1000000";--10

when "001011" =>

a\_to\_g1<="1111001";

a\_to\_g2<="1111001";--11

when "001100" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0100100";--12

when "001101" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0110000";--13

when "001110" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0011011";--14

when "001111" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0010010";--15

when "010000" =>

a\_to\_g1<="1111001";

a\_to\_g2<="1000000";--16

when "010001" =>

a\_to\_g1<="1111001";

a\_to\_g2<="1111000";--17

when "010010" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0000000";--18

when "010011" =>

a\_to\_g1<="1111001";

a\_to\_g2<="0011000";--19

when "010100" =>

a\_to\_g1<="0100100";

a\_to\_g2<="1000000";--20

when "010101" =>

a\_to\_g1<="0100100";

a\_to\_g2<="1111001";--21

when "010110" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0100100";--22

when "010111" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0110000";--23

when "011000" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0011011";--24

when "011001" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0010010";--25

when "011010" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0000010";--26

when "011011" =>

a\_to\_g1<="0100100";

a\_to\_g2<="1111000";--27

when "011100" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0000000";--28

when "011101" =>

a\_to\_g1<="0100100";

a\_to\_g2<="0011000";--29

when "011110" =>

a\_to\_g1<="0110000";

a\_to\_g2<="1000000";--30

when "011111" =>

a\_to\_g1<="0110000";

a\_to\_g2<="1111001";--31

when "100000" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0100100";--32

when "100001" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0110000";--33

when "100010" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0011011";--34

when "100011" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0010010";--35

when "100100" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0000010";--36

when "100101" =>

a\_to\_g1<="0110000";

a\_to\_g2<="1111000";--37

when "100110" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0000000";--38

when "100111" =>

a\_to\_g1<="0110000";

a\_to\_g2<="0011000";--39

when "101000" =>

a\_to\_g1<="0011011";

a\_to\_g2<="1000000";--40

when "101001" =>

a\_to\_g1<="0011011";

a\_to\_g2<="1111001";--41

when "101010" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0100100";--42

when "101011" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0110000";--43

when "101100" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0011011";--44

when "101101" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0010010";--45

when "101110" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0000010";--46

when "101111" =>

a\_to\_g1<="0011011";

a\_to\_g2<="1111000";--47

when "110000" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0000000";--48

when "110001" =>

a\_to\_g1<="0011011";

a\_to\_g2<="0011000";--49

when "110010" =>

a\_to\_g1<="0010010";

a\_to\_g2<="1000000";--50

when "110011" =>

a\_to\_g1<="0010010";

a\_to\_g2<="1111001";--51

when "110100" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0100100";--52

when "110101" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0110000";--53

when "110110" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0011011";--54

when "110111" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0010010";--55

when "111000" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0000010";--56

when "111001" =>

a\_to\_g1<="0010010";

a\_to\_g2<="1111000";--57

when "111010" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0000000";--58

when "111011" =>

a\_to\_g1<="0010010";

a\_to\_g2<="0011000";--59

when others =>

a\_to\_g1<="1000000";

a\_to\_g2<="1000000";

end case;

end process;

end structural;

---------------clock----------------------

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity digital\_clock is

Port (clk : in std\_logic;

minute\_out : out std\_logic\_vector(5 downto 0):="000000";

hour\_out: out std\_logic\_vector(5 downto 0):="000000";

led6\_to\_0 : out std\_logic\_vector(6 downto 0):="0000000");

end digital\_clock;

architecture structural of digital\_clock is

signal m1: std\_logic\_vector(5 downto 0):="000000";

signal h1 : std\_logic\_vector(5 downto 0):="000000";

signal count1 :std\_logic\_vector(27 downto 0):=x"000012B";

signal count2 :std\_logic\_vector(27 downto 0):=x"0004650";

signal count3 :std\_logic\_vector(27 downto 0):=x"0000004";

signal m\_a\_1: std\_logic\_vector(5 downto 0):="000000";----alarm minute

signal h\_a\_1 : std\_logic\_vector(5 downto 0):="000000";-----alarm hour

signal seconds :std\_logic\_vector(6 downto 0):="0000000";

signal int\_clk1: std\_logic :='0';

signal int\_clk2: std\_logic :='0';

signal int\_clk3: std\_logic :='0';

signal flag: std\_logic:='1';

begin

-----state transition------------

--------seconds-----------------

sec\_process:process(int\_clk3) is

begin

if(flag ='1') then

if (rising\_edge(int\_clk3)) then

if( seconds < "0111011")then

seconds<=seconds+1;

else

seconds<="0000000";

end if;

led6\_to\_0<= seconds;

end if;

end if;

end process sec\_process;

--------minute------------------

M1\_process:process(int\_clk1,m1) is

begin

if(flag ='1') then

if (rising\_edge(int\_clk1)) then

if(m1 < "111011") then

m1<=m1+1;

else

m1<="000000";

end if;

minute\_out<=m1;

end if;

else

minute\_out<=m1;

end if;

end process M1\_process;

--------hour------------------

M2\_process:process(int\_clk2) is

begin

if(flag='1')then

if( rising\_edge(int\_clk2))then

if(h1 < "010111") then

h1<=h1+1;

else

h1<="000000";

end if;

hour\_out<=h1;

end if;

else

hour\_out<=h1;

end if;

end process M2\_process;

----------- clocks-----------------

------------seconds clock---------

slow\_clock\_1sec: process(clk) is

begin

if(rising\_edge(clk)) then

if count3 < x"0000004" then

count3<=count3+1;

else

int\_clk3<= not int\_clk3;

count3<=(others => '0');

end if;

end if;

end process slow\_clock\_1sec;

------------60sec clock-----------

slow\_clock\_60sec: process(clk) is

begin

if(rising\_edge(clk)) then

if count1 < x"000012B" then

count1<=count1+1;

else

int\_clk1<= not int\_clk1;

count1<=(others => '0');

end if;

end if;

end process slow\_clock\_60sec;

------------1hr clock-----------

slow\_clock\_1hr: process(clk) is

begin

if(rising\_edge(clk)) then

if count2 < x"0004650" then

count2<=count2+1;

else

int\_clk2<= not int\_clk2;

count2<=(others => '0');

end if;

end if;

end process slow\_clock\_1hr;

end structural;

**Based on your design, what will be a good test bench for simulating the operation of your system? Please describe it and insert below the source code from your VHDL testbench file.**

library IEEE;

use IEEE.Std\_logic\_1164.all;

use IEEE.Numeric\_Std.all;

entity full\_system\_tb is

end;

architecture bench of full\_system\_tb is

component full\_system

port( sysclk :in std\_logic;

a\_to\_gout : out std\_logic\_vector(6 downto 0);

anode\_out : out std\_logic\_vector(7 downto 0);

led6\_0: out std\_logic\_vector(6 downto 0));

end component;

signal sysclk: std\_logic;

signal a\_to\_gout: std\_logic\_vector(6 downto 0);

signal anode\_out: std\_logic\_vector(7 downto 0);

signal led6\_0: std\_logic\_vector(6 downto 0);

constant clock\_period: time := 100 ms;

signal stop\_the\_clock: boolean;

begin

uut: full\_system port map ( sysclk => sysclk,

a\_to\_gout => a\_to\_gout,

anode\_out => anode\_out,

led6\_0 => led6\_0 );

stimulus: process

begin

-- Put initialisation code here

-- Put test bench stimulus code here

stop\_the\_clock <= false;

wait;

end process;

clocking: process

begin

while not stop\_the\_clock loop

sysclk <= '0', '1' after clock\_period / 2;

wait for clock\_period;

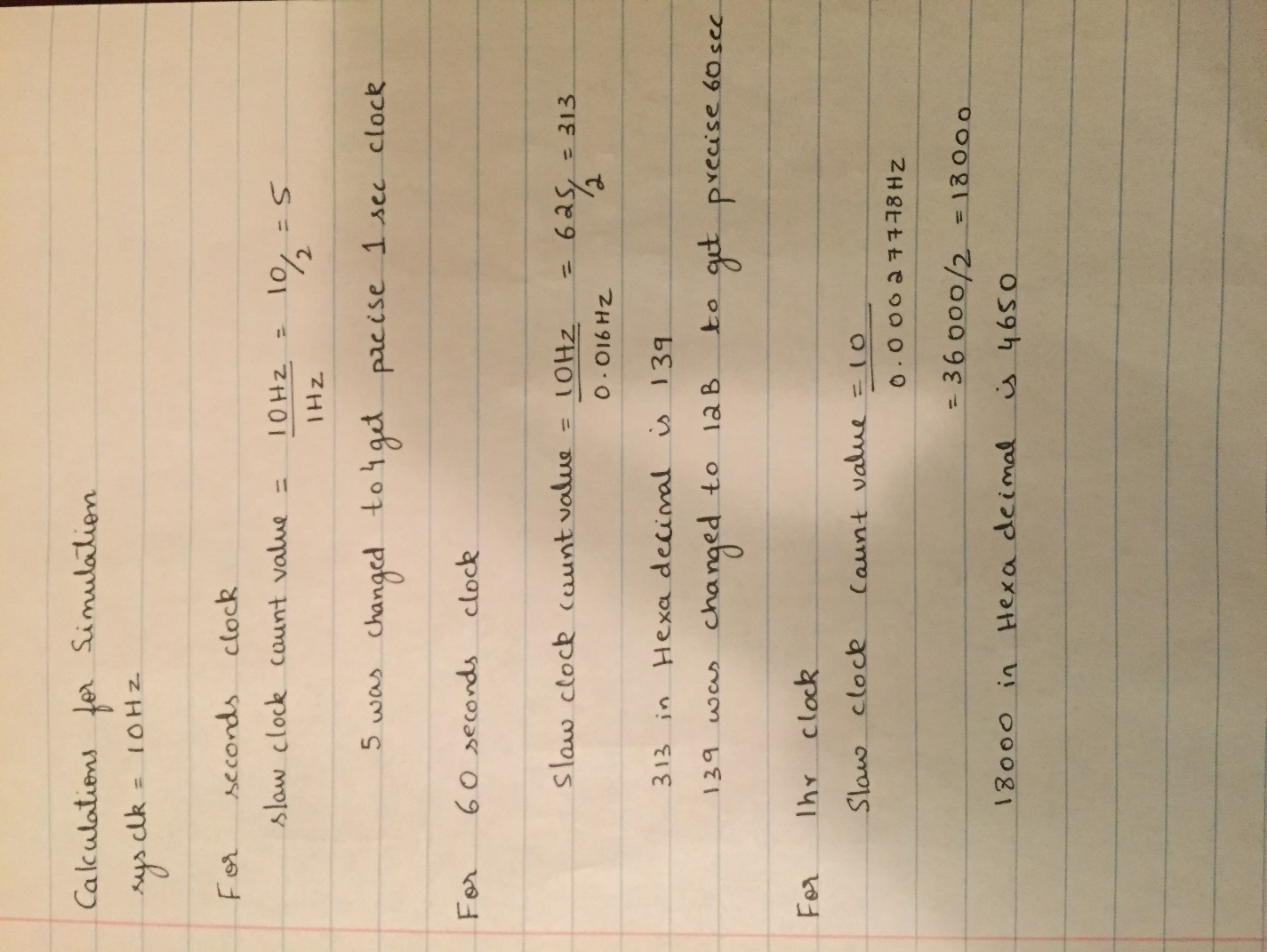
end loop;

wait;

end process;

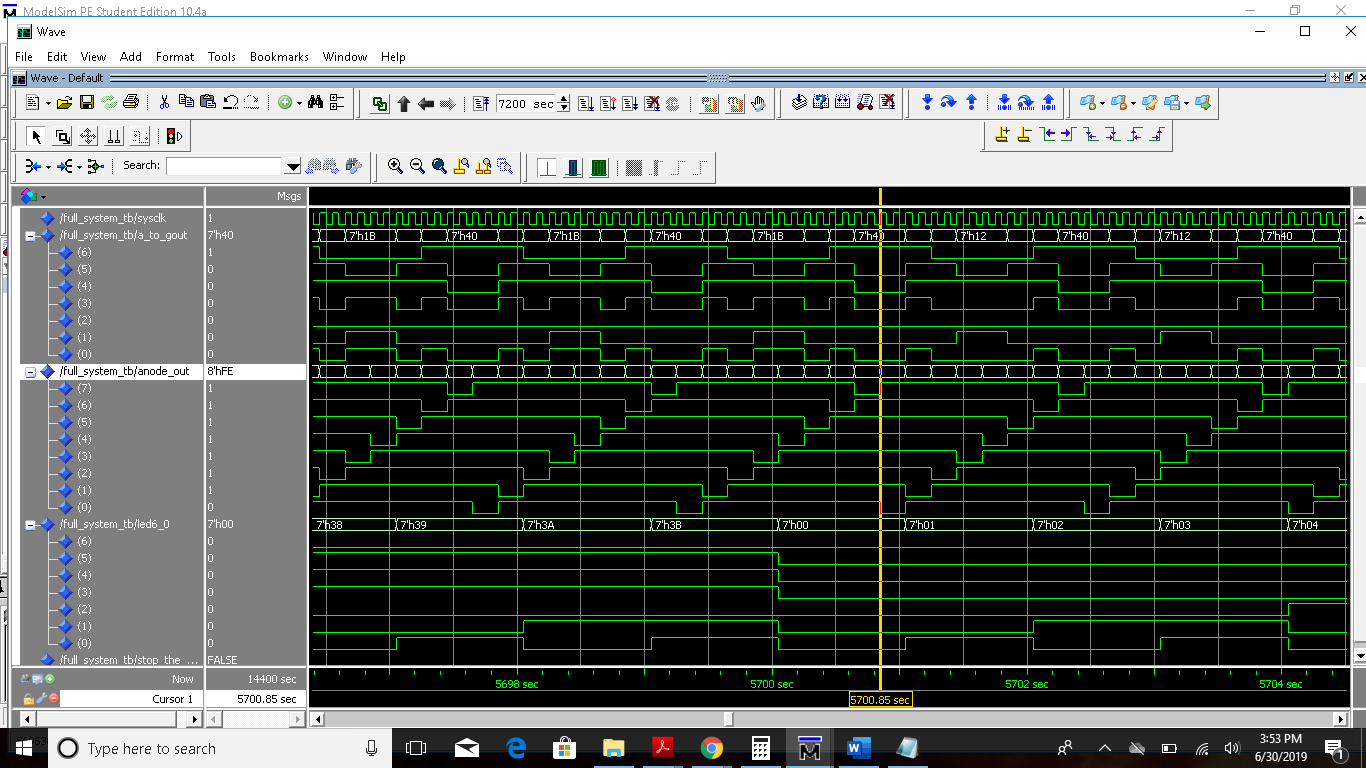
end;

**CALCULATIONS:**

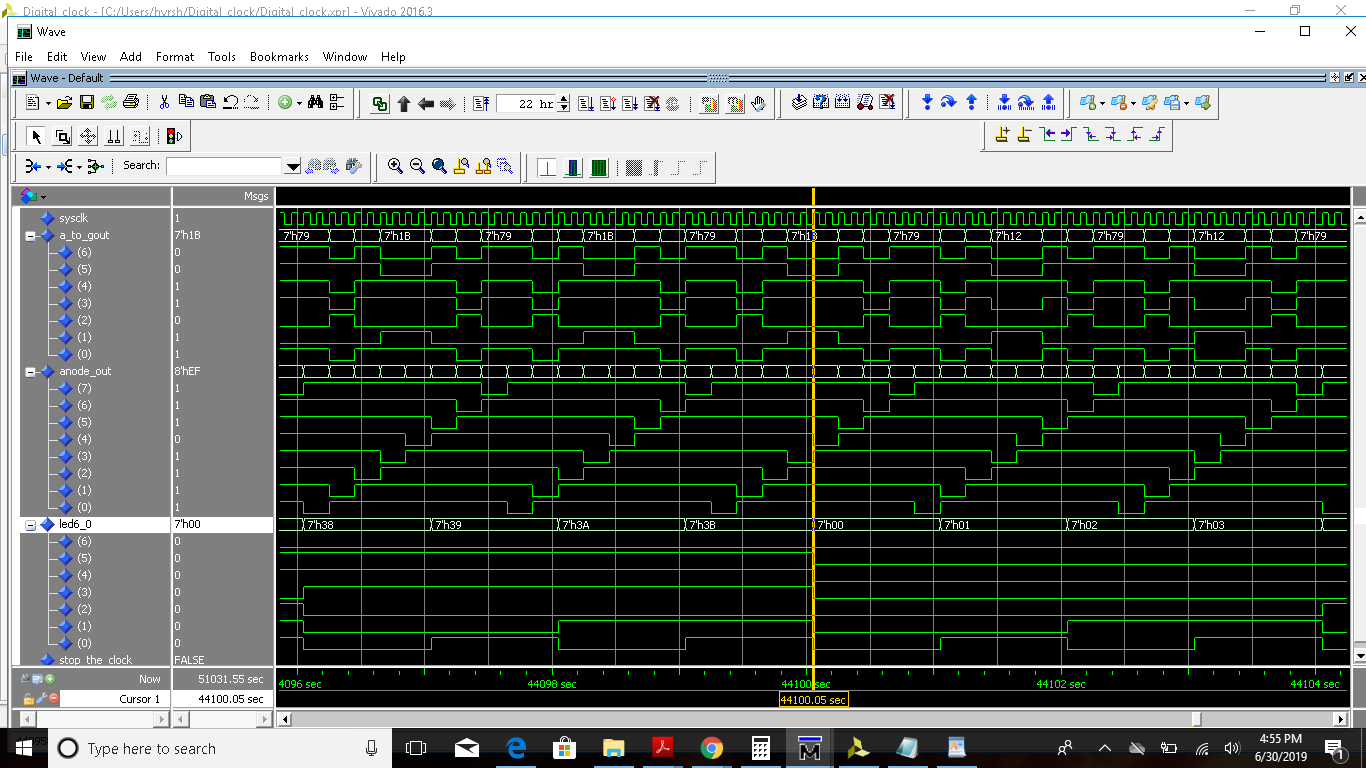


**What is the result waveform from your simulation? Based on this simulation, is your system implementation functioning as designed?**

**Output for 1:35 = 5700 sec.**

****

**Output at 12:15 = 44100 sec**

****

After the QuestaSim simulation, please implement your digital clock on the Nexys 4 DDR board, and use the setup as given earlier. After implementing your digital design on the Nexys 4 board, please demonstrate it to the instructor; and create a short video demonstrating your test results. Please upload the short video as an attachment to this report in Titanium.