Problem 1.

LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_unsigned.ALL;

-----------------------------------------------------------------------------

-- ENTITY: Problem1 - The dimes and nickels counter for vending machine

-- that dispenses 15 cents candy.

-- example 8.6. Page 537 in the textbook Brown and Vranesic.

-----------------------------------------------------------------------------

ENTITY Problem1 IS

PORT (

Resetn : IN std\_logic;

Clock : IN std\_logic;

DN : IN std\_logic\_vector(1 downto 0);

Z : OUT std\_logic

);

END Problem1;

-----------------------------------------------------------------------------

-- ARCHITECTURE: behav

-- This architecture is implemented with behavioural VHDL

-----------------------------------------------------------------------------

ARCHITECTURE behav OF Problem1 IS

-- This area is used to define types and any internal signals

type State\_type IS (S1,S2,S3,S4,S5);

signal y\_present, y\_next : State\_type;

begin

-- DN is the input signal that causes the states to change

process (DN, y\_present) is

BEGIN

CASE y\_present IS

WHEN S1 => --State1

if (DN = "00") then

y\_next <= S1;

elsif (DN = "01") then

y\_next <= S3;

elsif (DN = "10") then

y\_next <= S2;

else

y\_next <= S1;

end if;

WHEN S2 => --State2

if (DN = "00") then

y\_next <= S2;

elsif (DN = "01") then

y\_next <= S4;

elsif (DN = "10") then

y\_next <= S5;

else

y\_next <= S2;

end if;

WHEN S3 => --State3

if (DN = "00") then

y\_next <= S3;

elsif (DN = "01") then

y\_next <= S2;

elsif (DN = "10") then

y\_next <= S4;

else

y\_next <= S3;

end if;

WHEN S4 => --State4

y\_next <= S1; -- since the clock will happen quicker than a new coin

-- can be inserted this state should just return to S1

WHEN S5 => --State5

y\_next <= S3; -- since the clock will happen quicker than a new coin

-- can be inserted this state should just return to S3

-- to ask for another coin....( case of 20 cents )

END CASE;

end process;

process (Clock, Resetn)

begin

if (Resetn = '0') then

y\_present <= S1;

elsif (Clock' EVENT AND Clock = '1') then

y\_present <= y\_next;

end if;

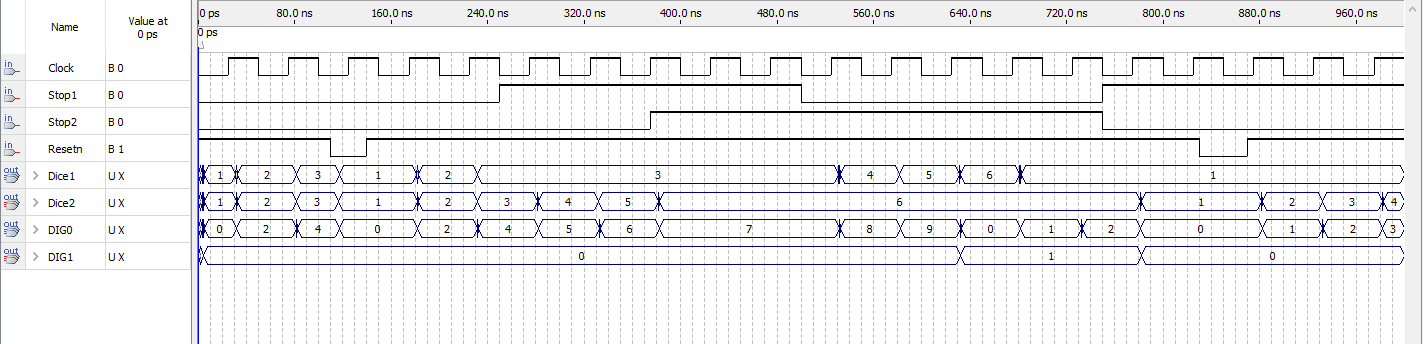
end process;

Z <= '1' when (y\_present = S5 or y\_present = S4) else '0';

END behav;

Problem 3.

In the cases outside the bounds of the dice the two dies were driven to 1.



LIBRARY ieee;

USE ieee.std\_logic\_1164.ALL;

USE ieee.std\_logic\_unsigned.ALL;

------------------------------------------------------------------------------------

-- ENTITY: ThunderBird - VHDL code that mimics teh classic ThunderBird turn singnal.

------------------------------------------------------------------------------------

ENTITY ThunderBirdVHD IS

PORT (

Resetn : IN std\_logic;

Clock : IN std\_logic;

Lft : IN std\_logic;

LeftTurnVHD: OUT std\_logic\_vector(2 downto 0)

);

END ThunderBirdVHD;

-----------------------------------------------------------------------------

-- ARCHITECTURE: behav

-- This architecture is implemented with behavioural VHDL

-----------------------------------------------------------------------------

ARCHITECTURE behav OF ThunderBirdVHD IS

-- This area is used to define types and any internal signals

type State\_type IS (NoSignal,Light1,Light2,Light3);

signal y\_present, y\_next : State\_type;

begin

process (Lft, y\_present) is

BEGIN

CASE y\_present IS

WHEN NoSignal => --State1

if (Lft = '0') then

y\_next <= NoSignal;

else

y\_next <= Light1;

LeftTurnVHD <= "001";

end if;

WHEN Light1 => --State2

y\_next <= Light2;

LeftTurnVHD<= "011";

WHEN Light1 => --State2

y\_next <= Light3;

LeftTurnVHD<= "111";

WHEN Light1 => --State2

y\_next <= NoSignal;

LeftTurnVHD<= "000";

END CASE;

end process;

process (Clock, Resetn)

begin

if (Resetn = '0') then

y\_present <= NoSignal;

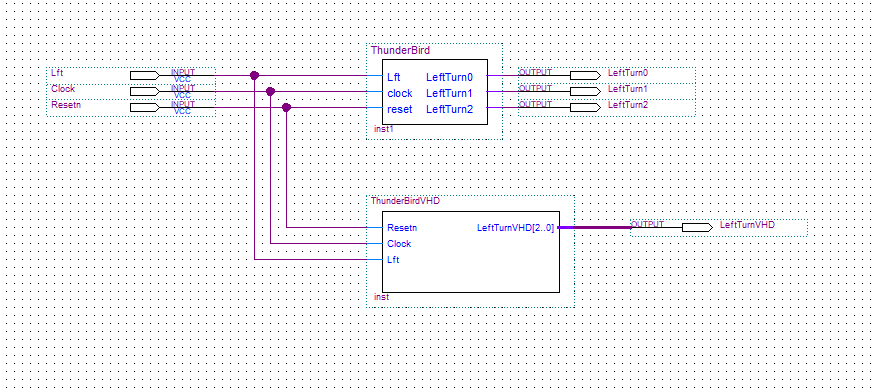
elsif (Clock' EVENT AND Clock = '1') then

y\_present <= y\_next;

end if;

end process;

END behav;



7. 