Deber de Digitales 2 Primer Parcial

Ejercicio 1:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port(Resetn, Clock, X, Y: in std\_logic;

OK: out std\_logic);

end deber;

architecture comportamiento of deber is

type estado is (e0, e1, e2, e3, e4, e5, e6, e7);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn='0' then y <= e0;

elsif clock'event and clok = '1' then

case y

when e0 =>

if X = '1' and Y = '1' then y <= e1;

else y <= e0;

end if;

when e1 =>

if X = '0' and Y = '0' then y <= e3;

elsif X = '0' and Y = '1' then y <= e5;

elsif X = '1' and Y = '0' then y <= e5;

else y <= e3;

end if;

when e2 =>

if X = '0' and Y = '0' then y <= e6;

elsif X = '0' and Y = '1' then y <= e4;

elsif X = '1' and Y = '0' then y <= e4;

else y <= e6;

end if;

when e3 =>

if X = '0' and Y = '0' then y <= e2;

elsif X = '0' and Y = '1' then y <= e7;

elsif X = '1' and Y = '0' then y <= e7;

else y <= e2;

end if;

when e4 =>

y <= e0;

when e5 =>

y <= e7;

when e6 =>

y <= e0;

end case;

end if;

end process;

OK <= not(X\*not(Y)+ not(X)\*Y) when y = e6 else '0';

end comportamiento;

Ejercicio 2:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, DG1, DG2: in std\_logic;

S: out std\_logic);

end deber;

architecture comportamiento of deber is

type estado is (e0, e1, e2, e3, e4);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= e0;

elsif clock'event and clock = '1' then

case y

when e0 =>

if DG1 = '0' and DG2 = '0' then y <= e0;

elsif DG1 = '0' and DG2 = '1' then y <= e2;

elsif DG1 = '1' and DG2 = '0' then y <= e1;

else y <= e0;

end if;

when e1 =>

if DG1 = '0' and DG2 = '0' then y <= e1;

elsif DG1 = '0' and DG2 = '1' then y <= e3;

elsif DG1 = '1' and DG2 = '0' then y <= e4;

else y <= e1;

end if;

when e2 =>

if DG1 = '0' and DG2 = '0' then y <= e2;

elsif DG1='0' and DG2 = '1' then y <= e1;

elsif DG1='1' and DG2 = '0' then y <= e3;

else y <= e2;

end if;

when e3 =>

if DG1 = '0' and DG2 = '0' then y <= e4;

else y <= e0;

end if;

when e4 =>

y <= e2;

end case;

end if;

end process;

S <= '1' when y = e3 or y = e4 else '0';

end comportamiento;

Ejercicio 3:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port(Resetn, Clock, A, B: in std\_logic;

HP: out std\_logic);

end deber;

architecture comportamiento of deber is

type estado is (e0, e1, e2, e3, e4, e5)

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = 0 then y<= e0;

elsif clock'event and clock = '1' then

case y

when e0 =>

if X = '0' and Y = '0' then y <= e0;

elsif X = '0' and Y = '1' then y <= e2;

elsif X = '1' and Y = '0' then y <= e1;

else y <= e0;

end if;

when e1 =>

if X = '0' and Y = '0' then y <= e2;

else y <= e3;

end if;

when e2 =>

if X = '1' and Y = '1' then y <= e1;

else y <= e3;

end if;

when e3 =>

y <= e0;

end if;

when e4 =>

if X = '0' and Y = '0' then y <= e0;

elsif X = '0' and Y = '1' then y <= e1;

elsif X = '1' and Y = '0' then y <= e2;

else y <= e3;

end if;

when e5 =>

y <= e0;

end if;

end case;

end if;

end process;

HP <= X or Y when y = e3 or y = e4 else '0';

end comportamiento;

Ejercicio 4:

std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, X1, X2: in std\_logic;

Q1, Q2: out std\_logic);

end deber;

architecture comportamiento of leccion is

type estado is (A, B, C);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= A;

elsif clock'event and clock = '1' then

case y

when A =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= B;

else y <= D;

end if;

when B =>

y <= A;

when C =>

if X = '0' and Y = '0' then y <= B0;

elsif X = '0' and Y = '1' then y <= D;

else y <= A;

end if;

end case;

end if

end process;

process (y, X1, X2)

begin

case y

when A =>

Q1 <= '1';

Q2 <= '1';

when B =>

Q1 <= X1 or X2;

Q2 <= '0';

when C =>

Q1 <= X1;

Q2 <= '1';

end case;

end process;

end comportamiento;

Ejercicio 5:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, In1, In2: in std\_logic;

S: out std\_logic);

end deber;

architecture comportamiento of leccion is

type estado is (S1, S2, S3);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= A;

elsif clock'event and clock = '1' then

case y

when S1 =>

if In1 = '0' and In2 = '0' then y <= S1;

elsif In1 = '0' and In2 = '1' then y <= S3;

elsif In1 = '1' and In2 = '0' then y <= S2;

end if;

when S2 =>

if In1 = '0' and In2 = '0' then y <= S2;

elsif In1 = '0' and In2 = '1' then y <= S1;

elsif In1 = '1' and In2 = '0' then y <= S3;

end if;

when S3 =>

if In1 = '0' and In2 = '0' then y <= S3;

elsif In1 = '0' and In2 = '1' then y <= S2;

elsif In1 = '1' and In2 = '0' then y <= S1;

end if;

end case;

end if

end process;

process (y, S)

begin

case y

when S1 =>

S <= '0';

when S2 =>

S <= (In1 or In2);

when S3 =>

S <= In1;

end case;

end process;

end comportamiento;

Ejercicio 6:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, X1, X2: in std\_logic;

Q1, Q2: out std\_logic);

end deber;

architecture comportamiento of leccion is

type estado is (A, B, C);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= A;

elsif clock'event and clock = '1' then

case y

when A =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= C;

else y <= B;

end if;

when B =>

if X = '0' and Y = '0' then y <= B;

elsif X = '0' and Y = '1' then y <= A;

else y <= C;

end if;

when C =>

if X = '0' and Y = '0' then y <= C;

elsif X = '0' and Y = '1' then y <= B;

else y <= A;

end if;

end case;

end if

end process;

process (y, Q1, Q2)

begin

case y

when A =>

Q1 <= '1';

Q2 <= '1';

when B =>

Q1 <= (X1 or X2);

Q2 <= '0';

when C =>

Q1 <= (X1 or X2);

Q2 <= '1';

end case;

end process;

end comportamiento;

Ejercicio 7:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, X1, X2: in std\_logic;

S1, S2: out std\_logic);

end deber;

architecture comportamiento of leccion is

type estado is (A, B, C, D);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= A;

elsif clock'event and clock = '1' then

case y

when A =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= D;

elsif X = '1' and Y = '0' then y <= B;

else y <= D;

end if;

when B =>

if X = '0' and Y = '0' then y <= A;

else y <= C;

end if;

when C =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= D;

elsif X = '1' and Y = '0' then y <= C;

else y <= D;

end if;

when D =>

if X = '0' and Y = '0' then y <= A;

else y <= C;

end if;

end case;

end if

end process;

process (y, S1, S2)

begin

case y

when A =>

S1 <= '1';

S2 <= '1';

when B =>

S1 <= '1';

S2 <= '0';

when C =>

S1 <= '0';

S2 <= '1';

when D =>

S1 <= '0';

S2 <= '1';

end case;

end process;

end comportamiento;

Ejercicio 8:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port(Resetn, Clock, K: in std\_logic;

R, S: out std\_logic);

end deber;

architecture comportamiento of deber is

type estado is (e0, e1, e2, e3, e4)

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = 0 then y<= e0;

elsif clock'event and clock = '1' then

case y

when e0 =>

if K = '0' then y <= e1;

else y <= e2;

end if;

when e1 =>

if X = '0' and Y = '0' then y <= e4;

end if;

when e2 =>

if K = '1' then y <= e1;

end if;

when e3 =>

if K = '0' then y <= e4;

else y <= e5;

end if;

end if;

when e4 =>

if K = '0' then y <= e0;

else y <= e5;

end if;

end case;

end if;

end process;

process (y, R, S)

begin

case y

when e0 =>

R <= '0';

S <= '0' when k = '0' else '1';

when e1 =>

R <= '1' when k = '0' else '0';

S <= '0' when k = '0' else '1';

when e2 =>

R <= '0' when k = '1';

S <= '0' when k = '1';

when e3 =>

R <= '0' when k = '0' else '1';

S <= '1';

when e4 =>

R <= '0';

S <= '1';

end case;

end process;

end comportamiento;

Ejercicio 9:

library ieee;

use ieee.std\_logic\_1164.all;

entity deber is

port (Resetn, Clock, X1, X2: in std\_logic;

S1, S2: out std\_logic);

end deber;

architecture comportamiento of leccion is

type estado is (A, B, C, D);

signal y: estado;

begin

process (resetn, clock)

begin

if resetn = '0' then y <= A;

elsif clock'event and clock = '1' then

case y

when A =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= D;

elsif X = '1' and Y = '0' then y <= B;

else y <= D;

end if;

when B =>

if X = '0' and Y = '0' then y <= A;

else y <= C;

end if;

when C =>

if X = '0' and Y = '0' then y <= A;

elsif X = '0' and Y = '1' then y <= D;

else y <= C;

end if;

when D =>

if X = '0' and Y = '0' then y <= A;

else y <= C;

end if;

end case;

end if

end process;

process (y, S1, S2)

begin

case y

when A =>

S1 <= X2;

S2 <= '1';

when B =>

S1 <= '0';

S2 <= '1';

when C =>

S1 <= X2;

S2 <= '0';

when D =>

S1 <= '0';

S2 <= X1;

end case;

end process;

end comportamiento;g