1) P1 and p2 run concurrently, since they are both sensitive to signal a

2) All sensitivities are evaluated at once, and only ha2 executes

4) Processes with sensitivity lists suspend when the last statement in the process is executed. Without a sensitivity list, processes suspend with wait statements

7)  
entity diagram is

Port(a,b,c : in std\_logic;

F : out std\_logic );

End diagram;

Architecture behavioral of diagram is

Signal u1o : std\_logic

Begin

U1: process (a,b)

Begin

U1o <= a or (not b);

End process u1;

U2: process(u1o, c)

Begin

F <= u1o and c;

End process u2;

End behavioral;

Architecture bahavioral2 of diagram is

begin

Process (a,b,c)

Variable tmp: std\_logic;

Begin

Tmp := ‘0’;

If(a or (not b))

Tmp <= ‘1’;

End if;

If(tmp and c)

F<= ‘1’;

Else

F <= ‘0’;

End if

End process

End behavioral2;

8) Arch

P1: Process(inputs)

Variable cnt : integer range 4 downto 0;

Begin

Cnt : = '0';

For I in 0 to 3 loop

If inputs(i) = '1' then

Cnt = cnt +1;

Else

Cnt = cnt;

End if

End loop

Count <= slv(to\_unsigned(cnt));

End process

End architecture;

10)

Architecture behav\_case of fuel\_encoder is

Begin

Process(fuel)

Begin

Case std\_logic\_vector(fuel) is

when “1100” | “1101” | “1110” | “1111” => led <= “0000”;

when “1000” | “1001” | “1010” | “1011” => led <= “1000”;

when “0100” | “0101” | “0110” | “0111” => led <= “1100”;

when “0001” | “0010” | “0011” => led <= “1110”;

when others => led <= “1111”;

end process;

end behav\_case;

13)

Architecture behavioral of decoder is

Process(w,x,y,z)

Begin

Case std\_logic\_vector(w,x,y,z) is

When “0011” => y <= “0000000001”;

When “0100” => y <= “0000000010”;

When “0101” => y <= “0000000100”;

When “0110” => y <= “0000001000”;

When “0111” => y <= “0000010000”;

When “1000” => y <= “0000100000”;

When “1001” => y <= “0001000000”;

When “1010” => y <= “0010000000”;

When “1011” => y <= “0100000000”;

When “1100” => y <= “1000000000”;

When others => y <= “XXXXXXXXXX”;

End case;

End process

End behavioral;

15)

Entity function is

Port(a,b,c : in std\_logic;

F : out std\_logic);

End function;

Architecture fcn\_ife of function is

Process(a,b,c)

Begin

If( ((not a) and b and (not c)) or (a and (not b) and (not c) ) or (a and b and (not c) ) )

F <= ‘1’;

Else

F <= ‘0’;

End if;

End process

End fcn\_ife;

Architecture fcn\_case of function is

Process(a,b,c)

Begin

case std\_logic\_vection(a,b,c) is

when “010”|”100” | “110” => f <= ‘1’;

when others => f <= ‘0’;

end case;

end process

end fcn\_case;

19)

Entity nand8 is

Port ( input : in SLV(7 downto 0);

Nand\_out : out sl);

End nand8;

Arch

Begin

Process(input)

Variable x : sl

Begin

X <= '1';

For I in 7 downto 0 loop

If input(i) = '0' then

X<= '0';

Exit;

End if;

End loop;

Nand\_out <= x;

End process;

28)

Entity homework is

Port (a,b,c : in std\_logic;

x,y : out std\_logic);

end homework;

Architecture behavioral of homework is

P1: Process (a,b,c)

Case std\_logic\_vector(a,b,c) is

When “000” => x <= ‘0’;

When “001” => x <= ‘1’;

When “010” => x <= ‘0’;

When “010” => x <= ‘0’;

When “011” => x <= ‘1’;

When “100” => x <= ‘0’;

When “101” => x <= ‘0’;

When “111” => x <= ‘0’;

When others => x <= ‘X’;

End case

End process p1;

P2: Process (a,b,c)

Case std\_logic\_vector(a,b,c) is

When “000” => y <= ‘1’;

When “001” => y <= ‘0’;

When “010” => y <= ‘1’;

When “010” => y <= ‘0’;

When “011” => y <= ‘0’;

When “100” => y <= ‘0’;

When “101” => y <= ‘0’;

When “111” => y <= ‘1’;

When others => y <= ‘X’;

End case

End process p2;

End behavioral;