TAE 1 - MDC

Date: __/__/__ Page No. : ____

Name: Jash Vikas Attari
Branch: Information Technology
Section: A
Roll no: 65

Q.1. Write the VHDL code in behavioural, structural and dataflow style for the following digital systems.

1) 3:8 Decoder

a) Behavioural style

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder3to8 is
    port (
        X0, X1, X2, Enable : in STD_LOGIC;
        Y : out STD_LOGIC_VECTOR(7 downto 0)
    );
end Decoder3to8;

architecture Behavioural of Decoder3to8 is
begin
    process(X0, X1, X2, Enable)
    begin
        if Enable = '1' then
```

Date: __/__/__ Page No. :____

case (X2 & X1 & X0) is
when "000" => Y <= "00000001";
when "001" => Y <= "00000010";
when "010" => Y <= "00000100";
when "011" => Y <= "00001000";
when "100" => Y <= "00010000";
when "101" => Y <= "00100000";
when "110" => Y <= "01000000";
when "111" => Y <= "10000000";
when others => Y <= "00000000";
end case;
else
Y <= "00000000";
end if;
end process;

end Behavioural;

b) Structural style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder3to8 is
port (
X0, X1, X2, Enable : in STD_LOGIC;
Y : out STD_LOGIC_VECTOR (7 downto 0);
);
end Decoder3to8;

Date: __/__/__ Page No. :____

architecture structural of Decoder3to8 is
signal X0_not, X1_not, X2_not : STD_LOGIC;
begin
-- NOT Gates
X0_not <= not X0;
X1_not <= not X1;
X2_not <= not X2;

-- AND gates for each output
Y(0) <= Enable and X2_not and X1_not and X0_not;
Y(1) <= Enable and X2_not and X1_not and X0;
Y(2) <= Enable and X2_not and X1 and X0_not;
Y(3) <= Enable and X2_not and X1 and X0;
Y(4) <= Enable and X2 and X1_not and X0_not;
Y(5) <= Enable and X2 and X1_not and X0;
Y(6) <= Enable and X2 and X1 and X0_not;
Y(7) <= Enable and X2 and X1 and X0;

end Structural;

c) Dataflow style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Decoder3to8 is
port (
X0, X1, X2, Enable : in STD_LOGIC;
Y : out STD_LOGIC_VECTOR (7 downto 0);
);
end Decoder3to8;

Date: __/__/__ Page No. : ____

architecture Dataflow of Decoder1to8 is
begin
Y(0) <= Enable and (not X2) and (not X1) and (not X0);
Y(1) <= Enable and (not X2) and (not X1) and X0;
Y(2) <= Enable and (not X2) and X1 and (not X0);
Y(3) <= Enable and (not X2) and X1 and X0;
Y(4) <= Enable and X2 and (not X1) and (not X0);
Y(5) <= Enable and X2 and (not X1) and X0;
Y(6) <= Enable and X2 and X1 and (not X0);
Y(7) <= Enable and X2 and X1 and X0;
end Dataflow;

2) 1:8 Demux

a) Behavioural style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Demux1to8 is
port (
D : in STD_LOGIC;
S : in STD_LOGIC_VECTOR (2 downto 0);
Y : out STD_LOGIC_VECTOR (7 downto 0));
end Demux1to8;

architecture Behavioural of Demux1to8 is
begin
process (D, S)
begin
Y <= "00000000";
case S is
when "000" => Y(0) <= D;
when "001" => Y(1) <= D;
when "010" => Y(2) <= D;
when "011" => Y(3) <= D;
when "100" => Y(4) <= D;
when "101" => Y(5) <= D;
when "110" => Y(6) <= D;
when "111" => Y(7) <= D;
when others => Y <= "00000000";
end case;
end process;
end Behavioural;

b) Structural style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Demux1to8 is
port (
D : in STD_LOGIC;
S : in STD_LOGIC_VECTOR (2 downto 0);
Y : out STD_LOGIC_VECTOR (7 downto 0));
);
end Demux1to8;

architecture Structural of Demux1to8 is

05/03/2025 08:07

Date: __/__/__ Page No. : ____

signal s0_not, s1_not, s2_not: STD_LOGIC;
begin
-- NOT Gate :
s0_not <= not s(0);
s1_not <= not s(1);
s2_not <= not s(2);

-- AND Gate for each output
Y(0) <= D and s2_not and s1_not and s0_not;
Y(1) <= D and s2_not and s1_not and s(0);
Y(2) <= D and s2_not and s(1) and s0_not;
Y(3) <= D and s2_not and s(1) and s(0);
Y(4) <= D and s(2) and s1_not and s0_not;
Y(5) <= D and s(2) and s1_not and s(0);
Y(6) <= D and s(2) and s(1) and s0_not;
Y(7) <= D and s(2) and s(1) and s(0);
end;

end Structural;

3) Dataflow Style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity Demux1to8 is
port (
D : in STD_LOGIC;
S : in STD_LOGIC_VECTOR (2 downto 0);
Y : out STD_LOGIC_VECTOR (7 downto 0);
);
end Demux1to8;

Date: __/__/__ Page No. : ____

architecture Dataflow of Demux1to8 is
begin
Y(0) <= D when S = "000" else '0';
Y(1) <= D when S = "001" else '0';
Y(2) <= D when S = "010" else '0';
Y(3) <= D when S = "011" else '0';
Y(4) <= D when S = "100" else '0';
Y(5) <= D when S = "101" else '0';
Y(6) <= D when S = "110" else '0';
Y(7) <= D when S = "111" else '0';

end Dataflow.

3) Full Subtractor

a) Behavioural Style

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FullSubtractor is
port (
A, B, Bin : in STD_LOGIC;
D, Bout : out STD_LOGIC;
);
end FullSubtractor;

architecture Behavioural of FullSubtractor is
begin
process (A, B, Bin) is

05/03/2025-08-07

Date: __/__/__ Page No. : ____

```
begin
    D <= A xor B xor Bin;
    Bout <= (not A and B) or (not A and Bin)
            or (B and Bin);
end process;
end Behavioral;
```

b) Structural style

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FullSubtractor is
    port (
        A, B, Bin : in STD_LOGIC;
        D, Bout : out STD_LOGIC
    );
end FullSubtractor;

architecture Structural of FullSubtractor is
    signal D1, B1, B2 : STD_LOGIC;
begin
    -- First Half Subtractor
    D1 <= A xor B;
    B1 <= (not A) and B;

    -- Second Half Subtractor
    D <= D1 xor Bin;
    B2 <= (not D1) and Bin;
```

Date: __/__/__ Page No. : ____

```
    -- OR gate for Borrow-out
    Bout <= B1 or B2;
end Structural;
```

c) Dataflow style

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity FullSubtractor is
    port (
        A, B, Bin : in STD_LOGIC;
        D, Bout : out STD_LOGIC
    );
end FullSubtractor;

architecture Dataflow of FullSubtractor is
begin
    D <= A xor B xor Bin;
    Bout <= (not A and B) or (not A and Bin) or
            (B and Bin);
end Dataflow;
```

05/03/2025 08:07