

1 D4 - TEKKOM B

PRAKTIKUM 2

PENDALAMAN SIMULASI ISIM



Nama	:	Septian Bagus Jumanoro
Kelas	:	1 – D4 Teknik Komputer B
NRP	:	3221600039
Dosen	:	Reni Soelistijorin B.Eng., M.T.
Mata Kuliah	:	Praktikum Rangkaian Logika 2
Hari/Tgl. Praktikum	:	Jumat, 18 Maret 2022



1. Tujuan

- Mahasiswa dapat mendisain rangkaian dari persamaan Boolean
- Mahasiswa dapat membuat RTL dari rangkaian logika
- Mahasiswa dapat membuat simulasi testbench dari rangkaian logika
- Mahasiswa mampu mendesain dan mensimulasikan rangkaian adder 4 bit

2. Teori

Ketika sebuah rancangan rangkaian diimplementasikan ke dalam FPGA, baik menggunakan metode schematic atau metode HDL, hendaknya perlu diuji agar kita dapat mengetahui hasil/output dari rangkaian tersebut. Proses pengujian rancangan rangkaian ini sering disebut proses simulasi. Melalui proses simulasi, kita dapat mengetahui apakah hasil rancangan rangkaian yang sudah dibuat sudah sesuai dan memenuhi tujuan yang diinginkan atau belum. Proses ini biasanya dilakukan sebelum rancangan rangkaian diimplementasikan ke dalam FPGA. Jadi urutan sebagai berikut:

1. Pembuatan rancangan rangkaian (Metode *schematic* atau metode HDL).
2. Proses simulasi rancangan rangkaian.
3. Implementasi ke dalam FPGA.

Nantinya, proses simulasi berfungsi untuk mengetahui sekaligus menguji apakah rancangan rangkaian yang telah dibuat mampu berjalan dengan baik atau tidak. Selain itu, lewat proses simulasi dapat diketahui bagaimana output dari rancangan rangkaian tadi. Selanjutnya, proses simulasi membutuhkan suatu bentuk stimulus/pemicu. Stimulus ini akan bertindak sebagai input awal bagi rancangan rangkaian yang hendak diuji. Kemudian, setelah diberikan stimulus maka rancangan rangkaian tersebut dapat diketahui bagaimana hasil outputnya. Keseluruhan proses simulasi ini dilakukan dengan bantuan perangkat lunak (software) yang ada.

Pada umumnya, proses simulasi terbagi atas 2 bentuk yakni:

Bentuk *testbench*.

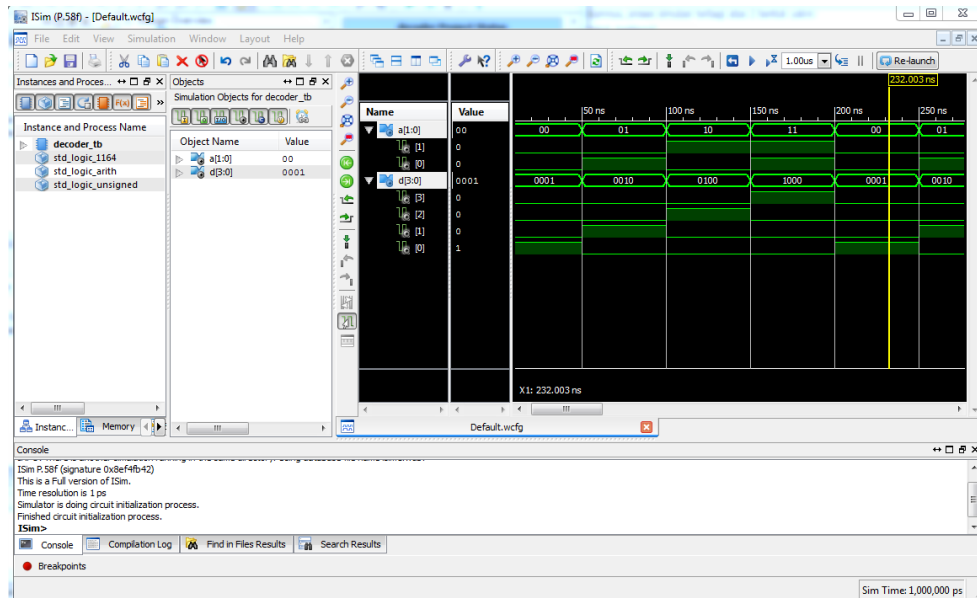
Bentuk *timing diagram*.

Timing Diagram

Proses simulasi dapat diketahui dengan membuat gambar timing diagram. Dengan menggunakan gambar timing diagram, dapat dilihat mengenai kondisi input, output serta hal-hal lain yang terkait. Bentuk *timing diagram* ini dibuat berdasarkan satuan waktu. Pada mulanya, untuk membuat timing diagram ini diperlukan stimulus untuk menentukan kondisi input awal. Selanjutnya, software simulator akan melakukan simulasi guna menghasilkan kondisi output sesuai dengan kondisi input tadi.

Cara ini merupakan cara yang sederhana dan mudah dilakukan, apalagi bagi para pengguna awal FPGA, bila dibandingkan dengan bentuk testbench. Hal ini disebabkan karena tampilan

timing diagram yang berupa gambar sehingga memudahkan siapa saja untuk mengamati, menganalisa, dan menjelaskan proses simulasi.



Testbench

Selain cara pertama, dikenal pula bentuk simulasi yang lain yaitu *testbench*. **Testbench** adalah proses pengujian suatu rancangan rangkaian. Dalam proses perancangan rangkaian, testbench akan menguji design rangkaian apakah sudah sesuai atau belum. Testbench ini dilakukan dengan menggunakan file HDL (berbentuk kode, baik VHDL maupun Verilog). Sehingga berbeda dengan bentuk diagram yang menggunakan gambar sebagai tampilannya, testbench menggunakan kode- kode tulisan sebagai tampilannya. Berikut contoh testbench :

```

59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96

```

```

constant PERIOD : time := 10 ns;

BEGIN
  -- Instantiate the Unit Under Test (UUT)
  uut: circuit3 PORT MAP (
    AT => AT,
    BT => BT,
    CT => CT,
    DT => DT,
    YT => YT
  );

  -- No clocks detected in port list. Replace <clock> below with
  -- appropriate port name

  -- Stimulus process
  stim_proc: process
  begin
    wait for PERIOD;
    AT<= '1';

    wait for PERIOD;
    BT<= '1';

    wait for PERIOD;
    AT<= '0';
    CT<= '1';

    wait for PERIOD;
    DT<= '1';

    wait for PERIOD;
  end process;
END;
  
```

3. Alat dan Bahan

1. PC yang sudah terinstal ISE 13.1
2. Xilinx Spartan 3
3. Downloader JTAG USB
4. Power Supply 5 volt

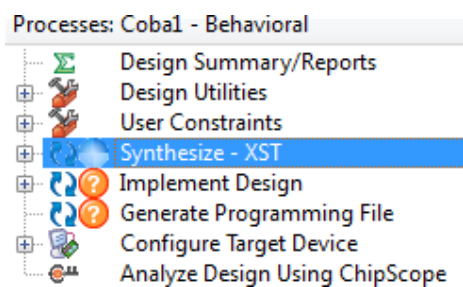
4. Langkah Percobaan

A. RTL dan Simulasi Test Bench

1. Sebelum melakukan test bench, buatlah sebuah program .vhd misal nya membuat rangkaian logika AND.

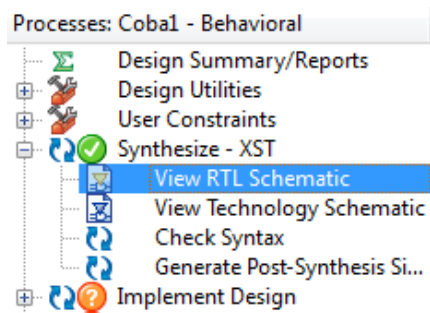
```
Library IEEE;  
  
use IEEE.STD_LOGIC_1164.ALL;  
  
entity Coba1 is  
  
    port(  
  
        a :in STD_LOGIC;  
        b :in STD_LOGIC;  
  
        y :out STD_LOGIC  
  
    );  
  
end Coba1;  
  
architecture Behavioral of Coba1 is  
begin  
  
    y <= a and b;  
  
end Behavioral;
```

2. Kemudian lakukan Synthesize-XST.



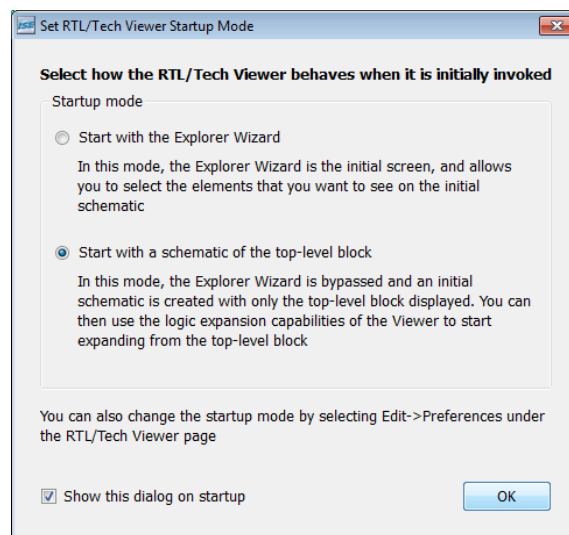
Gambar 2. 3 Proses Synthesize-XST

3. Klik 2x pada View RTL Simulation,



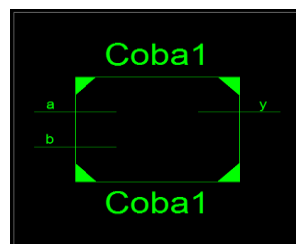
Gambar 2. 4 View RTL Simulation

4. Kemudian pilih *Start with a schematic of the top-level block* pada window Set RTL/Tech ViewerStartup Mode kemudian OK.



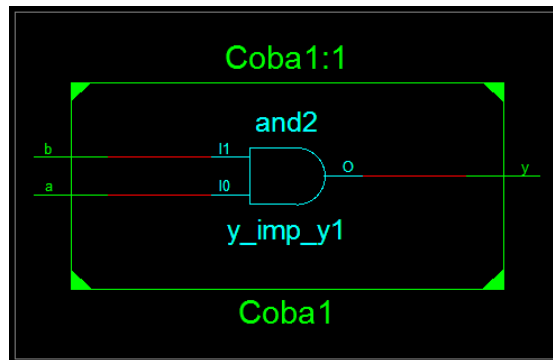
Gambar 2. 5 Set RTL/Tech Viewer Startup Mode

5. Akan muncul RTL skematik yaitu rangkaian yang sudah kita buat.



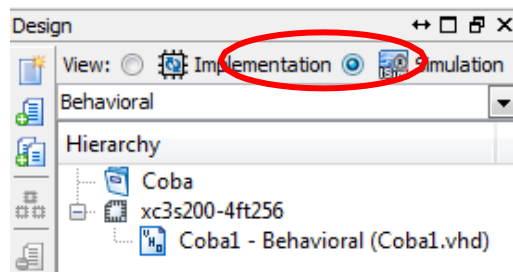
Gambar 2. 6 2.6 RTL Skematik

6. Klik 2x di dalam diagram block, maka akan muncul rangkaian logikanya



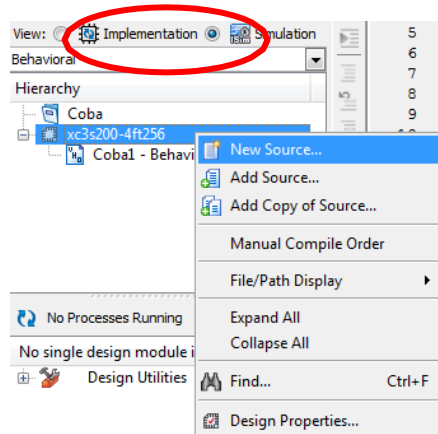
Gambar 2. 7 Rangkaian Logika AND

7. Kembali ke kode program, jika tidak terdapat error, pilih **Simulation** pada **View** di window kolom **Design**



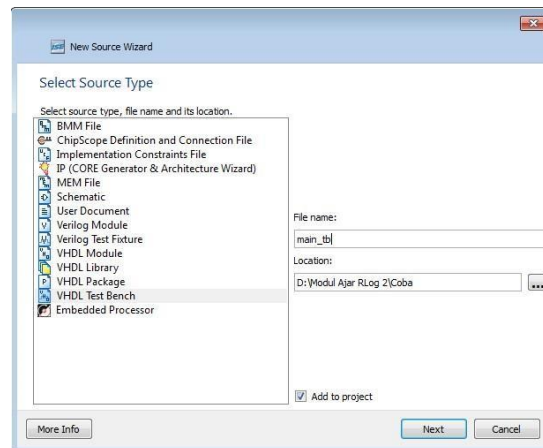
Gambar 2. 8 Simulation View

8. Kemudian buat project baru dengan memilih- New Source



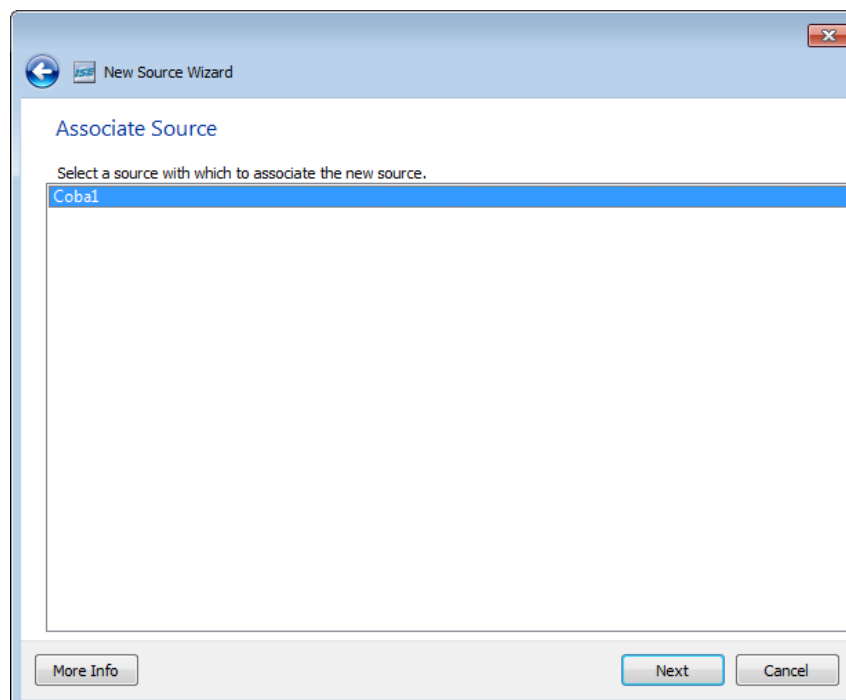
Gambar 2. 9 Membuat New Source

9. Pilih **VHDL Test bench** dan isi **file name**-nya dengan nama **main_tb**, kemudian klik Next.



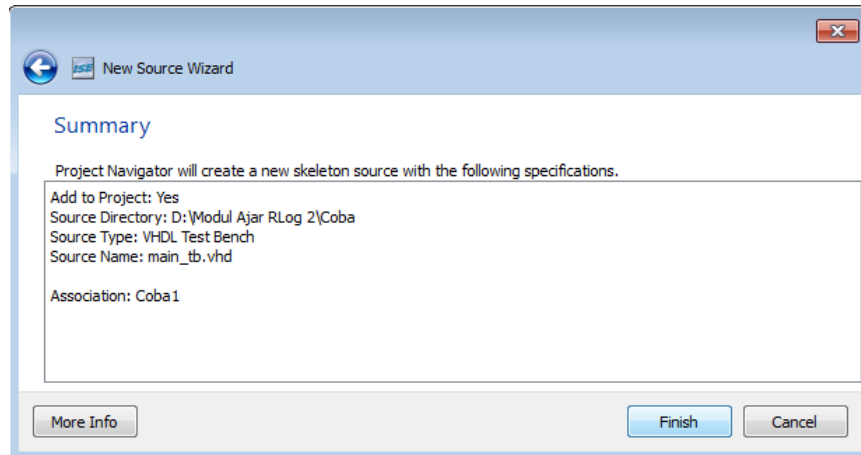
Gambar 2. 10 VHDL Test Bench

10. Bagian ini menunjukkan tentang file yang akan disimulasikan. Kemudian pilih Next. (Pada gambar 2.10 file yang akan disimulasikan adalah program untuk membuat rangkaian logika “AND” dan nama filenya adalah Coba1)



Gambar 2. 11 VHDL Test Bench

11. Kemudian pilih finish.



12. Setelah itu akan muncul kode dari main_tb.

```
35 ENTITY main_tb IS
36 END main_tb;
37
38 ARCHITECTURE behavior OF main_tb IS
39
40     -- Component Declaration for the Unit Under Test (UUT)
41
42     COMPONENT coba1
43     PORT (
44         a : IN  std_logic;
45         b : IN  std_logic;
46         y : OUT std_logic
47     );
48     END COMPONENT;
49
50
51     --Inputs
52     signal a : std_logic := '0';
53     signal b : std_logic := '0';
54
55     --Outputs
56     signal y : std_logic;
57     -- No clocks detected in port list. Replace <clock> below with
58     -- appropriate port name
59
60     constant <clock>_period : time := 10 ns;
```



```

61
62 BEGIN
63
64 -- Instantiate the Unit Under Test (UUT)
65 uut: cobal PORT MAP (
66     a => a,
67     b => b,
68     y => y
69 );
70
71 -- Clock process definitions
72 <clock>_process :process
73 begin
74     <clock> <= '0';
75     wait for <clock>_period/2;
76     <clock> <= '1';
77     wait for <clock>_period/2;
78 end process;
79
80
81 -- Stimulus process
82 stim_proc: process
83 begin
84     -- hold reset state for 100 ns.
85     | wait for 100 ns;
86
87     wait for <clock>_period*10;
88
89     -- insert stimulus here
90
91     wait;
92 end process;
93
94 END;

```

13. Pada source code uut-main_src, comment baris program yang dilingkari warna merah.

```

59
60 -- constant <clock>_period : time := 10 ns;
61 BEGIN
62 -- Instantiate the Unit Under Test (UUT)
63 uut: cobal PORT MAP (
64     a => a,
65     b => b,
66     y => y
67 );
68 -- Clock process definitions
69 -- <clock>_process :process
70 -- begin
71 --     <clock> <= '0';
72 --     wait for <clock>_period/2;
73 --     <clock> <= '1';
74 --     wait for <clock>_period/2;
75 --     end process;
76 --
77
78 -- Stimulus process
79 stim_proc: process
80 begin
81     -- hold reset state for 100 ns.
82     wait for 100 ns;
83
84     -- wait for <clock>_period*10;
85

```

14. Pada bagian **stim_proc : process**, ubahlah baris kodenya menjadi seperti skrip 2.2.

Script 2. 2 Membagi periode waktu

```
-- Stimulus process
stim_proc:process
begin

-- hold reset state for 100 ns.
Wait for 100 ns
;a <=not a;

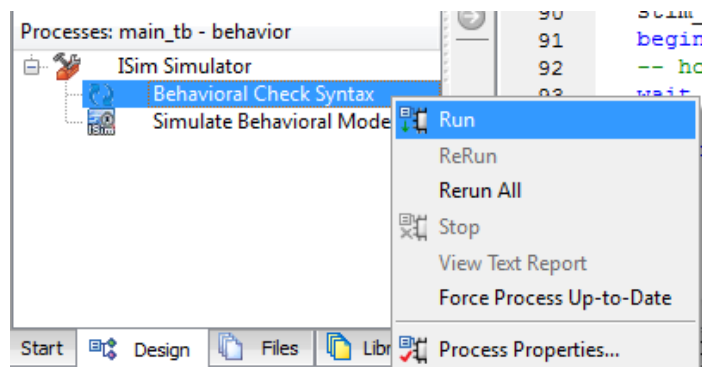
end process;

-- Stimulus process
process stim_proc
1:process begin
-- hold reset state for 100 ns.
Wait for 50 ns
;b <=not b;

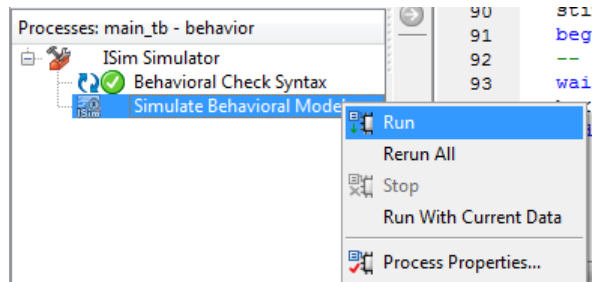
end process;
```

*“Catatan : Dalam kode diatas, maksudnya adalah waktu sinyal **a** akan high selama 100ns dan low selama 100ns, demikian juga pada signal **b** akan high selama 50ns dan low selama 50ns. “*

15. Kemudian Cek syntax, dengan klik kanan pada **Behavioral Check Syntax-Run**.

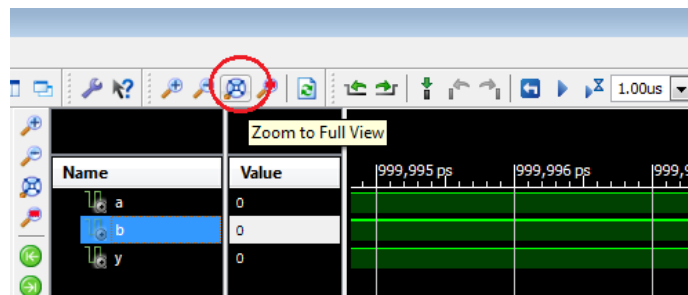


16. Kemudian klik kanan **Simulate Behavioral Model-Run** untuk menampilkan simulasi.



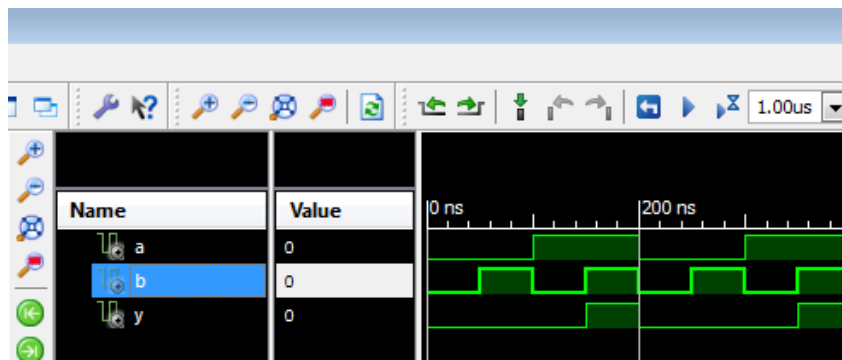
Gambar 2. 14 Menampilkan Simulasi Isim

17. Pilih **Zoom to full view** untuk melihat secara detail.



Gambar 2. 14 Zoom to full view

18. Kemudian akan tampak timing diagram seperti gambar 2.16



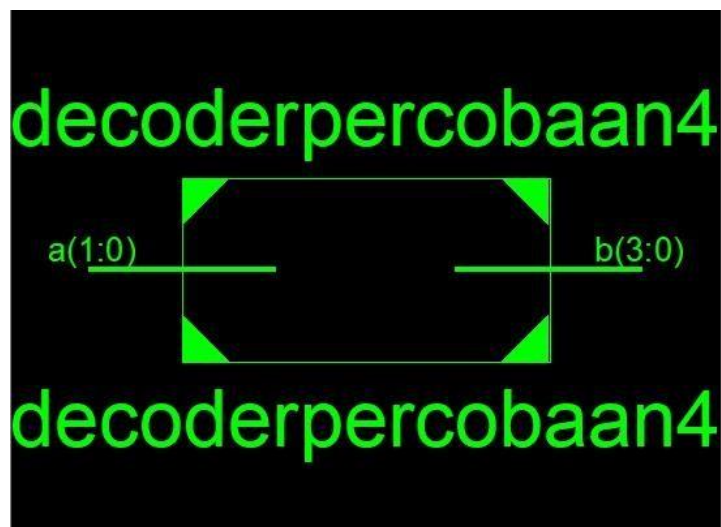
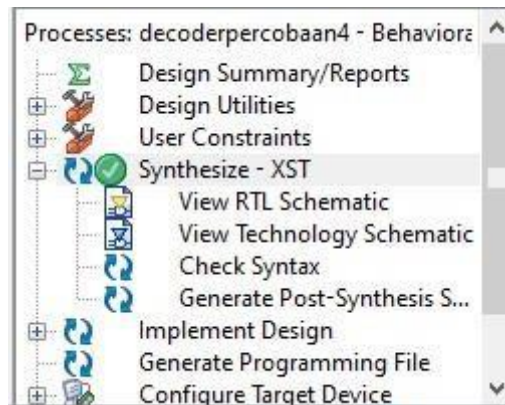
Gambar 2. 15 Timing Diagram rangkaian logika AND

“Pada timing diagram gambar 2.16 hanya ketika sinyal a dan b berlogika high, sinyal output akan mengeluarkan logika high ($C=A.B$) “

B. Project 2

1. Buatlah project baru untuk disain decoder 2 to 4. Lakukan seperti langkah sebelumnya. Berikut program decoder 2 to 4.

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3
4  entity decoderpercobaan4 is
5  Port ( a : in STD_LOGIC_VECTOR (1 downto 0);
6        b : out STD_LOGIC_VECTOR (3 downto 0)
7        );
8  end decoderpercobaan4;
9
10 architecture Behavioral of decoderpercobaan4 is
11
12 begin
13     with a select
14         b(3 downto 0) <= "0001" when "00",
15                          "0010" when "01",
16                          "0100" when "10",
17                          "1000" when "11",
18                          "0000" when others;
19 end Behavioral;
20
21
```



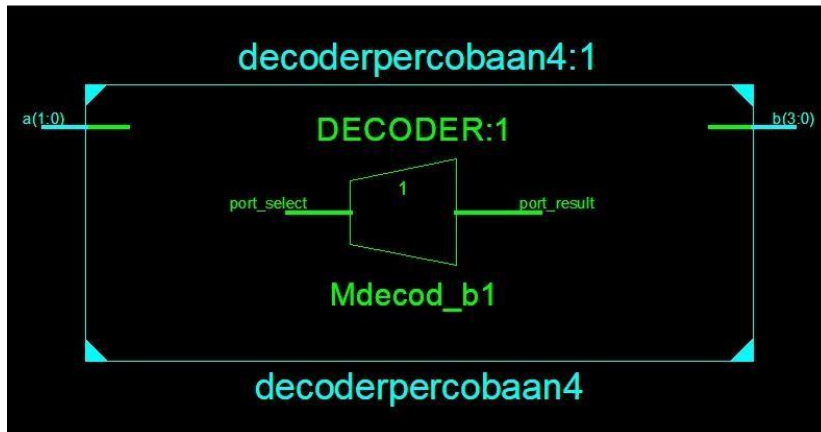
2. Berikut program testbench.

```
BEGIN
--
-- Instantiate the Unit Under Test (UUT)
uut: decoder PORT MAP (
    a => a,
    d => d
);

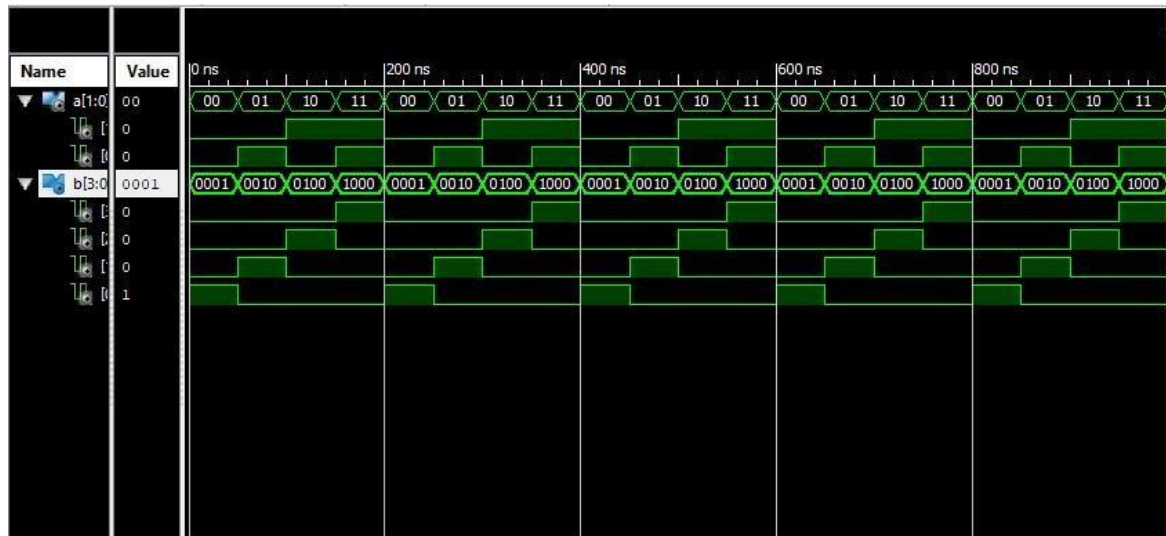
-- Clock process definitions
-- <clock>_process :process
-- begin
-- <clock> <= '0';
-- wait for <clock>_period/2;
-- <clock> <= '1';
-- wait for <clock>_period/2;
-- end process;

-- Stimulus process
stim_proc: process
begin
    wait for 50 ns;
    a(0) <= not a(0);
end process;

stim_procl: process
begin
    wait for 100 ns;
    a(1) <= not a(1);
end process;
END;
```



3. Dan hasil dari simulasinya ditampilkan seperti gambar dibawah ini.



5. Latihan

a. Tabel Kebenaran Rangkaian Biner to Gray 4 bit

NO	BINER	GRAY
1	0000	0000
2	0001	0001
3	0010	0011
4	0011	0010
5	0100	0110
6	0101	0111
7	0110	0101
8	0111	0100
9	1000	1100
10	1001	1101
11	1010	1111
12	1011	1110
13	1100	1010
14	1101	1011
15	1110	1001
16	1111	1000

b. K-Map

$$G1 = \sum(8,9,10,11,12,13,14,15)$$

$$G2 = \sum(4,5,6,7,8,9,10,11)$$

$$G3 = \sum(2,3,4,5,10,11,13)$$

$$G4 = \sum(1,2,6,9,10,13,14)$$

G1:

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

$$G1 = A$$

G2:

AB \ CD	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	0	0	0	0
10	1	1	1	1

$$G2 = A'B + AB' = A \oplus B$$

G3:

AB \ CD	00	01	11	10
00	0	0	1	1
01	1	1	0	0
11	1	1	0	0
10	0	0	1	1

$$G3 = BC' + B'C = B \oplus C$$

G4:

AB \ CD	00	01	11	10
00	0	1	0	1
01	0	1	0	1
11	0	1	0	1
10	0	1	0	1

$$G4 = C'D + CD' = C \oplus D$$

c. Persamaan Boolean

$$G1 = A$$

$$G2 = A'B + AB' = A \oplus B$$

$$G3 = BC' + B'C = B \oplus C$$

$$G4 = C'D + CD' = C \oplus D$$

d. Program VHDL Biner To Gray 4 Bit

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

entity latihan1_1 is
    port(A, B, C, D: in STD_LOGIC;
          G1, G2, G3, G4: out STD_LOGIC
    );
end latihan1_1;

architecture Behavioral of latihan1_1 is
begin

    G1 <= A;
    G2 <= A xor B;
    G3 <= B xor C;
    G4 <= C xor D;

end Behavioral;
```

e. TestBench

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY latihan1_1 IS
END latihan1_1;
ARCHITECTURE behavior OF latihan1_1 IS
    -- Component Declaration for the Unit Under Test (UUT)
    COMPONENT latihan1_1
    PORT(
        a : IN  std_logic_vector(3 downto 0);
        y : OUT std_logic_vector(3 downto 0)
    );
    END COMPONENT;
    --Inputs
    signal a : std_logic_vector(3 downto 0) := (others => '0');

    --Outputs
    signal y : std_logic_vector(3 downto 0);
    -- No clocks detected in port list. Replace <clock> below with
    -- appropriate port name
BEGIN
```

```

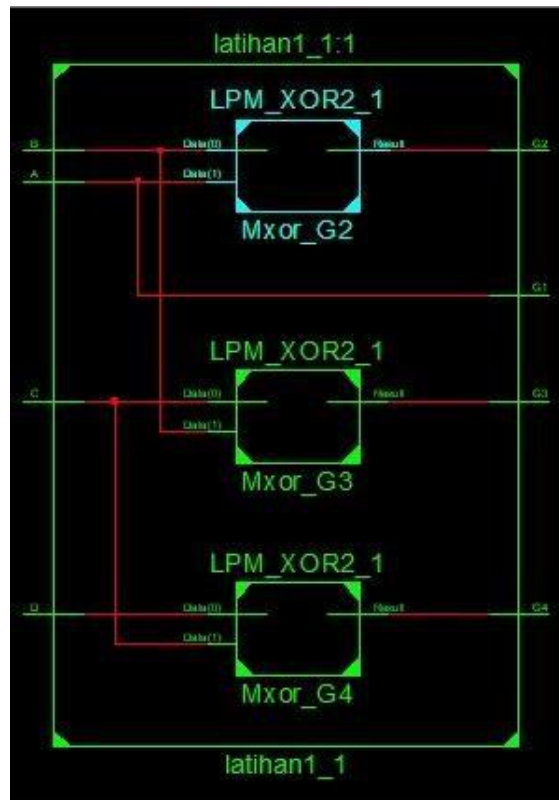
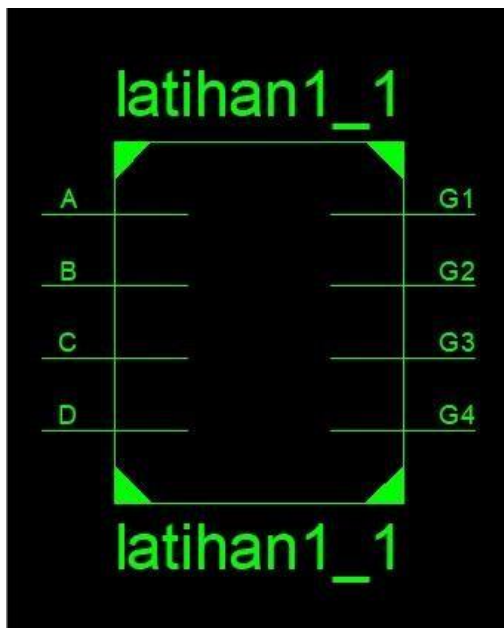
-- Instantiate the Unit Under Test (UUT)

uut: latihan1_1 PORT MAP (
    a => a,
    y => y
);

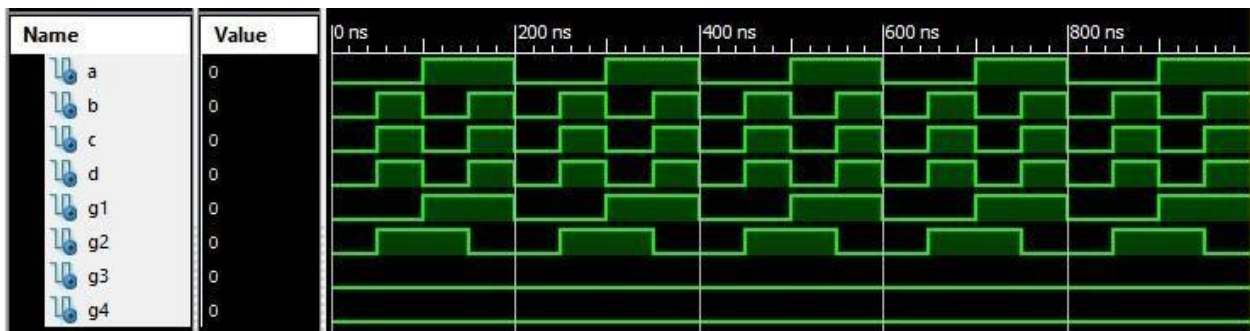
-- Stimulus process
a0_proc: process
begin
    wait for 25 ns;
    a(0) <= not a(0);
end process;

```

f. Tampilan RTL Schematic



g. Timing Diagram



6. ANALISA

No. _____
Date: _____

☐ Pada percobaan tersebut berfungsi untuk melihat timing diagram

☐ dari sebuah program rangkaian logika yang telah melalui test bench.

☐ Pada program biner to gray 4 bit berfungsi mengkonversi binary digit

☐ menjadi nilai yang sama dengan gray digit

☐ Persamaan berikut ini :

☐ $\Rightarrow G_1 = A$

☐ $\Rightarrow G_2 = \bar{A}B + A\bar{B} = A \oplus B$

☐ $\Rightarrow G_3 = B\bar{C} + \bar{B}C = B \oplus C$

☐ $\Rightarrow G_4 = C\bar{D} + \bar{C}D = C \oplus D$