MODUL III

Perancangan Aritmetic Logic Unit (ALU) pada FPGA

I. Tujuan

Pada Percobaan ini praktikan akan mempelajari tentang bagaimana cara mengembangkan Aritmetic Logic Unit (ALU) pada IC FPGA dengan pendekatan hirachical (berjenjang) berbasis perpaduan Schematic-VHDL. Pengembangan dilakukan dengan cara membuat sub-module sistem satu persatu kemudian diintegrasi menjadi satu sistem utuh.

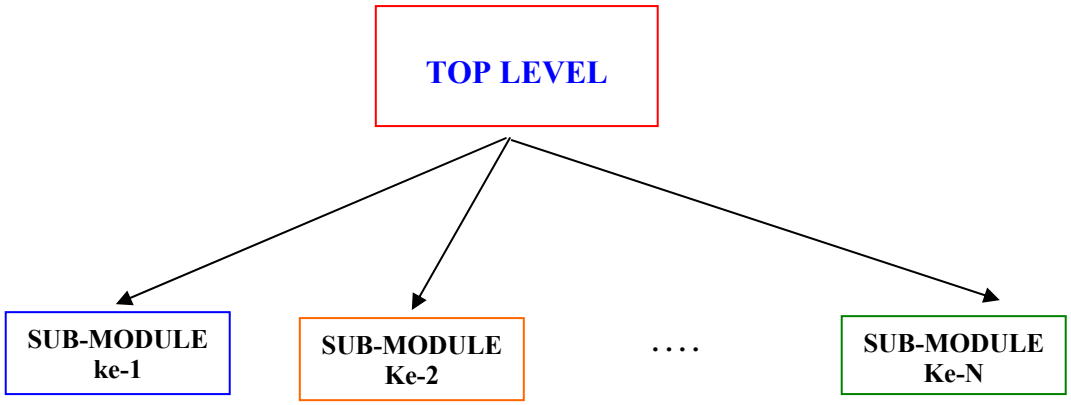
II. Kompetensi

Setelah meneyelesaikan percobaan ini diharapkan praktikan mempunyai kemampuan untuk :

- Mengembangkan Aritmetic Logic Unit (ALU) pada IC FPGA
- Mengembangkan perancangan FPGA dengan pendekatan Hierarchical (Berjenjang)
- Memadukan schematic dan VHDL untuk suatu perancangan

III. DASAR TEORI

HIERARCHICAL DESIGN yaitu perancangan circuit yang terdiri atas beberapa sub-module, dilakukan dengan cara membuat module-module terkecil terlebih dahulu kemudian diintegrasikan menjadi module utama dalam **TOP LEVEL**.



TOP LEVEL dan LEVEL dibawahnya bisa berupa kombinasi VHDL-VHDL atau SCHEMATIC-SCHEMATIC atau campuran/mix dari SCHEMATIC dan VHDL.

Pada percobaan ini akan dirancang Aritmatic Logic Unit (ALU) 4-bit yang terdiri atas beberapa level module dan akan diselesaikan dengan pendekatan HIERARCHICAL DESIGN serta perpaduan SCHEMATIC-VHDL. Langkah penyelesaian yang dilakukan adalah dengan cara membuat semua sub-module yang ada dengan VHDL , setelah dipastikan rancangan sub-module telah benar maka digenerate schematic simbolnya. Langkah berikutnya dengan cara membuat TOP LEVEL berupa schematic, pada top level ini semua sub-module yang telah dibuat diintegrasi menjadi satu sistem.

Mode kalkulasi dipilih dengan cara menekan sebuah push-button. Mode yang dipilih akan ditampilkan pada led. Bilangan A dan B masing-masing selebar 4-bit yang akan dimasukkan menggunakan DIP-switch. Apabila push-button lainnya ditekan, yang berfungsi sebagai tombol enter, maka hasil kalkulasi akan ditampilkan pada dua buah seven segmen dalam format heksadesimal.

Nomor Mode	Fungsi
000	**A + B**
001	**A - B**
010	**A * B**
011	**2s Complement A**
100	**2s Complement B**
101	**A AND B**
110	**A OR B**
111	**A XOR B**

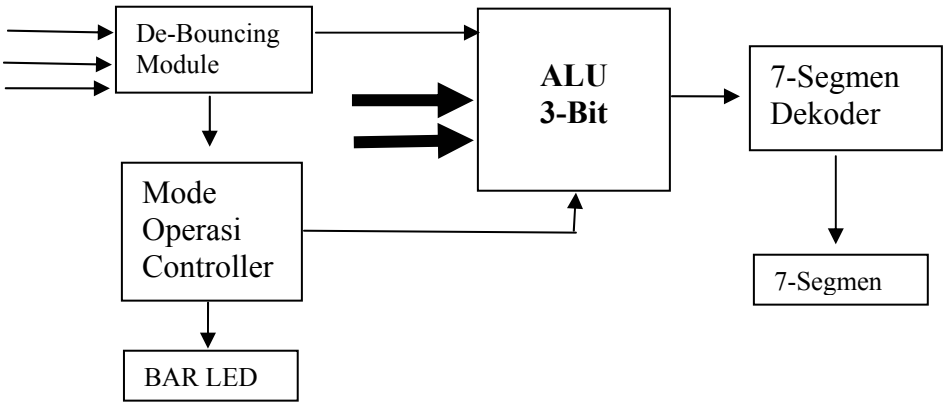
IV. PERALATAN

- 1 set PC yang dilengkapi dengan software ISE WebPack versi 6.1 atau lebih serta software ModelSim.
- 1 development board XSA-50 dan XSTend Board.
- 1 power-supply +9VDC.

V. PROSEDUR PRAKTIKUM

Gambaran Disain

Pada praktikum kali ini akan dibuat sebuah Arithmatic Logic Unit (ALU) 4-bit sederhana. ALU yang dibuat mempunyai 10 mode pilihan operasi , output dari ALU akan ditampilkan pada dua buah seven segmen.



Untuk merealisasikan ALU 2-bit ini, maka disain besar tersebut dapat dibuat empat buah blok disain yang lebih kecil yaitu:

1. Blok sinkronisasi input
2. Blok pemilih mode kalkulasi
3. Blok kalkulator
4. Blok tampilan ke dua seven segmen

Empat blok diatas didisain menggunakan VHDL. Kemudian buat (create) menjadi skematik. Hubungkan keempat blok secara skematik (top-level disain project adalah skematik).Kemudian mendefinisikan koneksi pin-pin IC nya menggunakan file *.ucf. Maka project siap disintesis, implementasi, dan mendownload bit-stream ke dalam divais.

Secara garis besar, langkah-langkah praktikum sebagai berikut:

1. Buatlah project baru dengan tipe **top-level modul skematik**, tempatkan project tersebut pada direktori e:\VLSI\perc3

2. Pilih nilai parameter berikut :

   device family: Spartan2

   device: xc2s50

   package: tq144

   speed grade: -6

3. Tambahkan (new source) kedalam project tersebut empat disain modul VHDL, lakukan check-syntax, bila tidak ada error buatlah (create) simbol skematiknya.

4. Tambahkan kedalam project sebuah modul skematik sebagai top-level. Hubungkan keempat modul tersebut secara skematik.

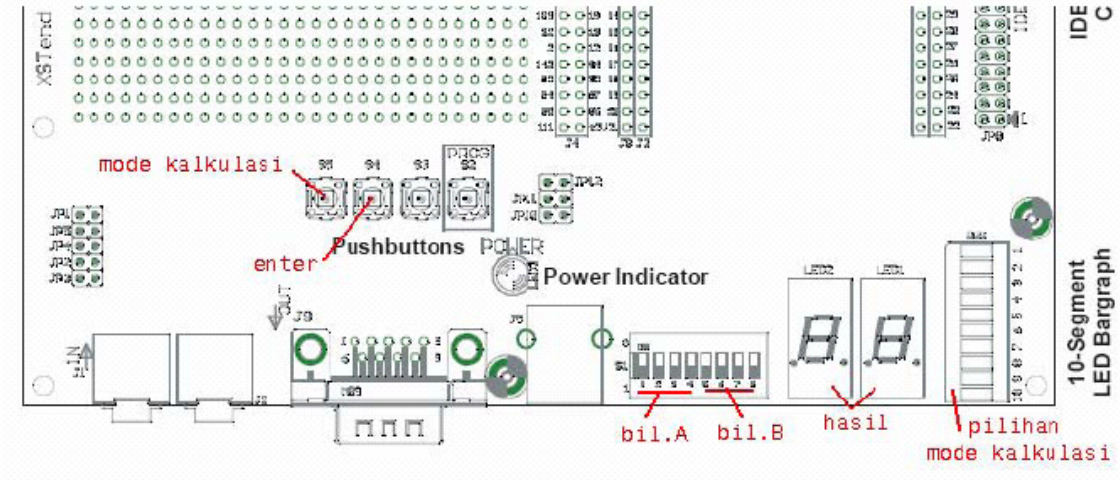
5. Tambahkan kedalam project sebuah file ucf untuk mendefinisikan pin-pin yang dipakai, sesuaikan dengan nama pin pada modul skematik.

6. Highlight modul skematik, klik ganda *Generate Programming File* untuk melakukan sintesis dan implementasi. Apabila tidak ada error maka akan dihasilkan file bitstream dengan ekstensi *.BIT

7. Simulasikan hasil rancangan.

8. Dengan menggunakan GXSLOAD, lakukan download kedalam target.

9. Pada board, pilih mode kalkulasi, masukkan nilai bilangan pada dip-switch, tekan enter, maka seven segmen akan menampilkan hasilnya.

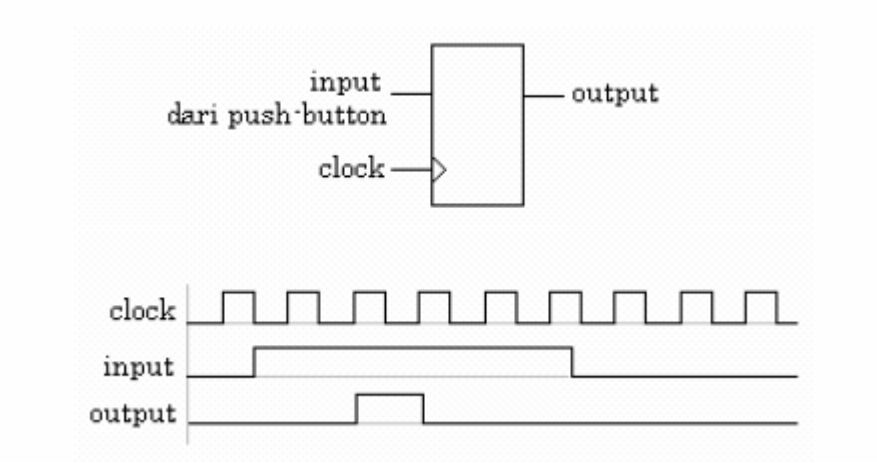


Gambar. Posisi I/O pada board yang digunakan

Berikut panduan untuk membuat kode VHDLnya.

Blok sinkronisasi input

Semua blok didisain secara sinkron menggunakan sebuah sinyal clock yang sama, penekanan tombol push-button akan disinkronkan terhadap clock. Clock yang digunakan (default) mempunyai frekuensi 50MHz atau 20ndetik sedangkan penekanan tombol push-button dapat menghasilkan pulsa yang lebih lebar sekitar beberapa milidetik. Ilustrasinya diperlihatkan oleh gambar . Blok tersebut bertugas menghasilkan satu pulsa selebar satu periode clock.



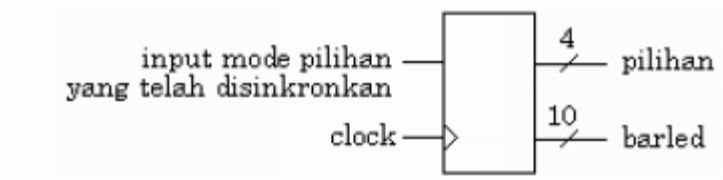
Gambar . Mensinkronkan input yang lebih lebar terhadap clock

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity debouncing is
port(
      clk : in std_logic;
      enter, mode : in std_logic;
      enter_db, mode_db : out std_logic
      );
end debouncing;
architecture Behavioral of debouncing is
signal A,B : std_logic_vector(2 downto 0) := "000";
begin
      process(clk,enter)
```

```
      begin
      if (clk'event and clk='1') then
            A <= A(1 downto 0) & enter; -- shift left
      end if;
      end process;

      process(clk,mode)
      begin
      if (clk'event and clk='1') then
            B <= B(1 downto 0) & mode; -- shift left
      end if;
      end process;

      enter_db <= not(A(2)) and A(1) and A(0);
      mode_db <= not(B(2)) and B(1) and B(0);

end Behavioral;
```

Blok pemilih mode kalkulasi

Bagian ini bertugas untuk menghasilkan hitungan dari 0 sampai 7. Hasil hitungan tersebut digunakan untuk menentukan mode pilihan kalkulasi seperti pada tabel. Mode pilihan kalkulasi ini ditampilkan pada 8-buah led yang terdapat pada board.



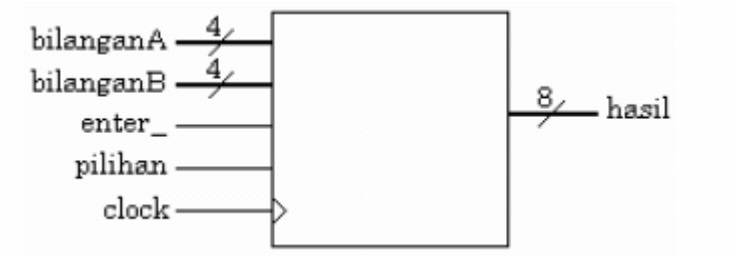
Gambar. Mendapatkan mode pilihan dari 0 sampai 7

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity pilih_mode is
```

```
port(
      clk : in std_logic;
      mode_db : in std_logic;
      barled : out std_logic_vector(7 downto 0);
      pilihan : out std_logic_vector(2 downto 0);
      );
end pilih_mode;

architecture Behavioral of pilih_mode is
      signal count : std_logic_vector(2 downto 0);
begin
      process(clk,mode_db)
      begin
      if (clk'event and clk='1') then
            if (mode_db='1') then
                  if (count = "111") then
                        count <= "000";
                  else
                        count <= count + "001";
                  end if;
            end if;
      end if;
      end process;
-- keluarkan
      pilihan <= count;
-- tampilkan
      barled <=   "00000001" when (count="000") else
                  "00000010" when (count="001") else
                  "00000100" when (count="010") else
                  "00001000" when (count="011") else
                  "00010000" when (count="100") else
                  "00100000" when (count="101") else
                  "01000000" when (count="110") else
                  "10000000" when (count="111") else
                  "00000000";
end Behavioral;
```

Blok ALU 3-Bit

Bagian ini adalah inti dari ALU 4-bit. Blok ini mendapat masukan dua buah bilangan A dan B masing-masing selebar 4-bit yang didapat dari DIP-switch pada board. Penekanan enter_ (sinyal ini didapat dari keluaran blok sinkronisasi input) akan menghasilkan keluaran hasil selebar 8-bit sesuai dengan sinyal pilihan yang didapat dari blok pemilih mode kalkulasi.

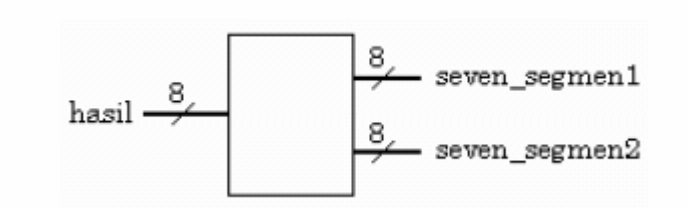



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity ALU is
port(
      clk : in std_logic;
      enter_db : in std_logic;
      bilA : in std_logic_vector(3 downto 0);
      bilB : in std_logic_vector(3 downto 0);
      pilihan : in std_logic_vector(2 downto 0);
      hasil : out std_logic_vector(7 downto 0)
      );
end ALU;

architecture Behavioral of ALU is
signal temp : std_logic_vector(7 downto 0) := "00000000";
begin
      process(clk,enter_db,bilA,bilB,pilihan)
      begin
      if (clk'event and clk='1') then
            if (enter_db='1') then
                  case pilihan is
                        when "000" => -- (+)
```

```
                        temp <= ("0000"&bilA) + ("0000"&bilB);
                        when "001" => -- (-)
                        temp <= ("0000"&bilA) - ("0000"&bilB);
                        when "010" => -- (*)
                        temp <= bilA * bilB;
                        when "011" => --2s complement
                        temp <= not("0000"&bilA) + "00000001";
                        when "100" => --2s complement
                        temp <= not("0000"&bilB) + "00000001";
                        when "101" =>
                        temp <= ("0000"&bilA) AND ("0000"&bilB);
                        when "110" =>
                        temp <= ("0000"&bilA) OR ("0000"&bilB);
                        when "111" =>
                        temp <= ("0000"&bilA) XOR ("0000"&bilB);
                        when others =>
                        temp <= "00000000";
                end case;
            end if;
      end if;
end process;

hasil <= temp;
end Behavioral;
```

Blok tampilan ke 2 seven segmen

Bagian ini akan menampilkan hasil keluaran dari blok kalkulasi selebar 8-bit pada dua buah seven segmen dalam format heksadesimal.

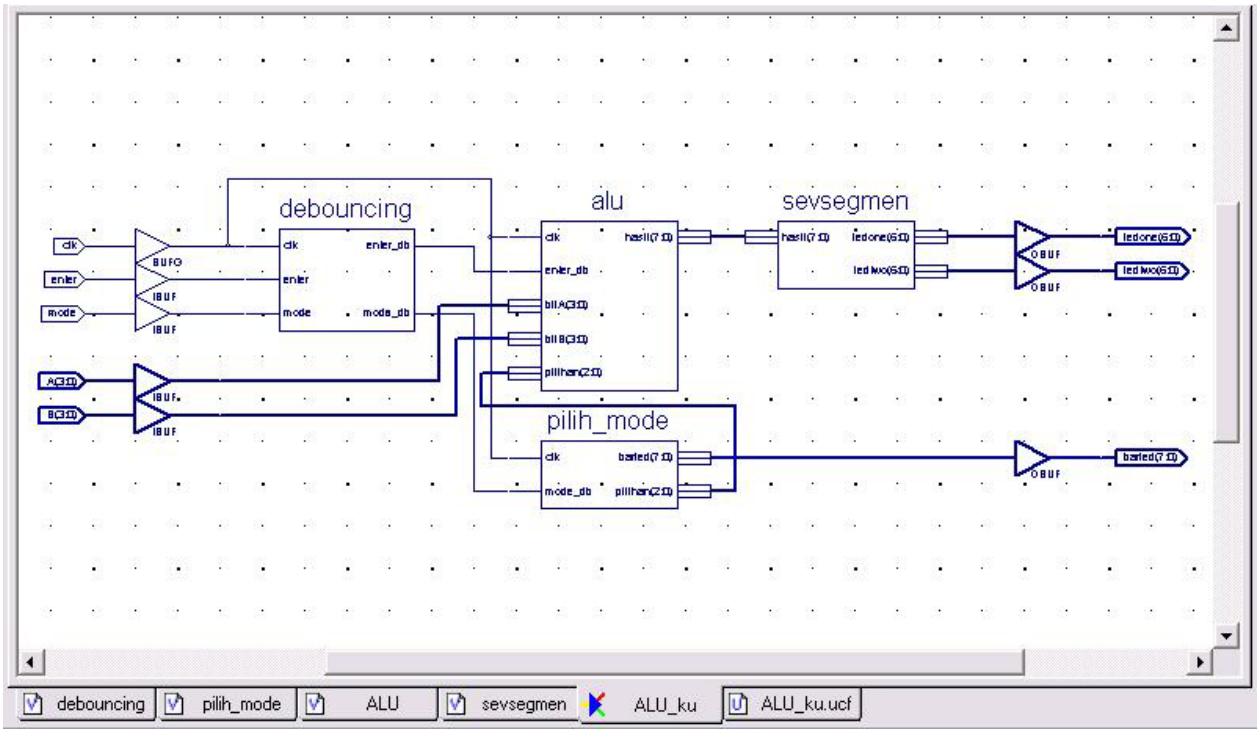



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity sevsegmen is
port(
hasil : in std_logic_vector(7 downto 0);
ledone,ledtwo : out std_logic_vector(6 downto 0)
);
end sevsegmen;

architecture Behavioral of sevsegmen is
signal LSB, MSB : std_logic_vector(3 downto 0);
begin
LSB <= hasil(3 downto 0);
MSB <= hasil(7 downto 4);

ledone <= "1110111"          when LSB="0000" else --0
            "0010010"        when LSB="0001" else --1
            "1011101"        when LSB="0010" else --2
            "1011011"        when LSB="0011" else --3
            "0111010"        when LSB="0100" else --4
            "1101011"        when LSB="0101" else --5
            "1101111"        when LSB="0110" else --6
            "1010010"        when LSB="0111" else --7
            "1111111"        when LSB="1000" else --8
            "1111011"        when LSB="1001" else --9
            "1111110"        when LSB="1010" else --A
            "0101111"        when LSB="1011" else --B
            "1100101"        when LSB="1100" else --C
            "0011111"        when LSB="1101" else --D
            "1101101"        when LSB="1110" else --E
            "1101100" ;                          --F

ledtwo <= "1110111"          when MSB="0000" else --0
            "0010010"        when MSB="0001" else --1
            "1011101"        when MSB="0010" else --2
            "1011011"        when MSB="0011" else --3
            "0111010"        when MSB="0100" else --4
            "1101011"        when MSB="0101" else --5
```

```
            "1101111"        when MSB="0110" else --6
            "1010010"        when MSB="0111" else --7
            "1111111"        when MSB="1000" else --8
            "1111011"        when MSB="1001" else --9
            "1111110"        when MSB="1010" else --A
            "0101111"        when MSB="1011" else --B
            "1100101"        when MSB="1100" else --C
            "0011111"        when MSB="1101" else --D
            "1101101"        when MSB="1110" else --E
                        "1101100" ;                          --F

end Behavioral;
```

File Schematic hasil penggabungan 4 buah modul VHDL



File ucf untuk mendefinisikan pin-pin IC

Nomor-nomor pin yang dipakai untuk seven segmen, barled, push-button dan dip-switch.

```
net clk loc=p88;
#net pushsw3 loc=p78; # pushbuttons
net mode loc=p26;
net enter loc=p23;
```

```
net ledtwo<0> loc=p47; # rightmost 7-segment LED
net ledtwo<1> loc=p40;
net ledtwo<2> loc=p28;
net ledtwo<3> loc=p29;
net ledtwo<4> loc=p27;
net ledtwo<5> loc=p42;
net ledtwo<6> loc=p48;
net ledone<0> loc=p64; # leftmost 7-segment LED
net ledone<1> loc=p65;
net ledone<2> loc=p76;
net ledone<3> loc=p50;
net ledone<4> loc=p51;
net ledone<5> loc=p54;
net ledone<6> loc=p56;
net barled<0> loc=p68; # bargraph LED
net barled<1> loc=p44;
net barled<2> loc=p46;
net barled<34> loc=p49;
net barled<4> loc=p57;
net barled<5> loc=p62;
net barled<6> loc=p60;
net barled<7> loc=p67;
net A<3> loc=p30; # DIP switches
net A<2> loc=p58;
net A<1> loc=p74;
net A<0> loc=p75;
net B<3> loc=p66;
net B<2> loc=p77;
net B<1> loc=p80;
net B<0> loc=p79;
```

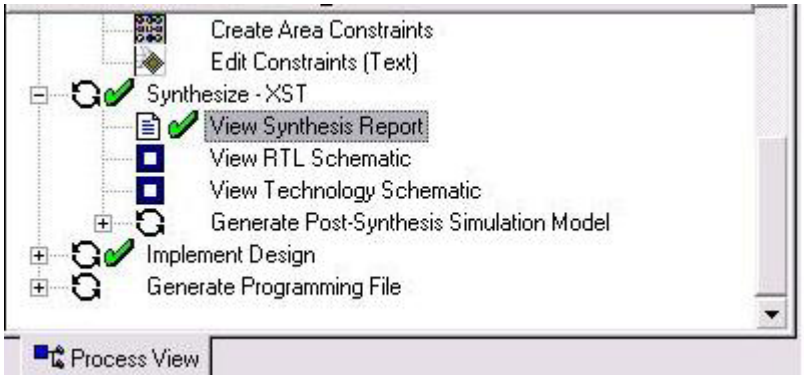
VI. Hasil Pengujian

Sebagai laporan sementara dan data pengujian maka lakukan percobaan untuk melengkapi lampiran Hasil Pengujian pada lembar yang telah tersedia (***Lampiran 1. Lembar Hasil Pengujian).***

VII. Analisa

Pada Laporan Resmi lakukan analisa kinerja dari alat yang dibuat dengan membandingkan antara hasil simulasi dengan hasil pengujian. Serta analisa perbedaan antara simulasi FUNCTIONAL dengan simulasi TIMING.

Analisa informasi yang anda dapat dari file synthesis report.



VIII. Tugas

Jawablah pertanyaan berikut:

1. Apa yang dimaksud dengan perancangan Hirarki (Hirarchical design) pada pengembangan FPGA?
2. Gambarkan disain top-level skematik yang dibuat.
3. Jelaskan kembali fungsi dan cara kerja dari masing-masing blok (modul) yang telah dibuat.

IX. Daftar Pustaka

1. Kevin Skahill, “VHDL for Programmable Logic”, Addison Wesley
2. M. Morris Mano, “Digital Design” (3rd Edition), Prentice Hall
3. M. Morris Mano & C. Kime, “Logic and Computer Design Fundamentals” Prentice Hall
4. Stefan Sjoholm & L. Lindh, “VHDL for Designers” Prentice Hall
5. Xilinx FPGA IseWebpack 7.0 Tutorial

Lampiran 1.