MODUL II

Perancangan FPGA untuk Implementasi Rangkaian Sequential dan Kombinational

I. Tujuan

Pada Percobaan ini praktikan akan mempelajari tentang bagaimana cara mengembangkan rangkaian logika sequential dan kombinational pada FPGA. Metode yang digunakan untuk dengan pemrograman VHDL.

II. Kompetensi

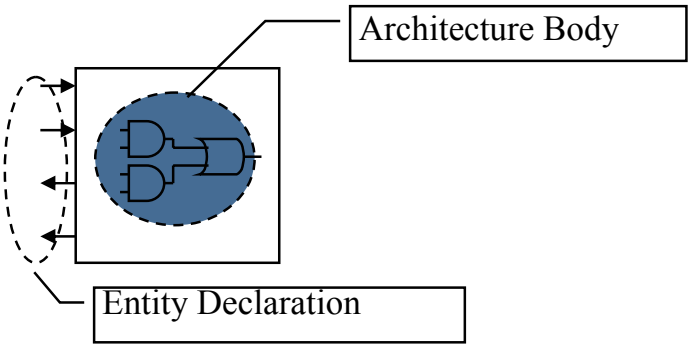
Setelah meneyelesaikan percobaan ini diharapkan praktikan mempunyai kemampuan untuk :

- Mengembangkan rangakaian logika sequential dan komninational pada FPGA
- Menjelaskan struktur pemrograman VHDL
- Mampu mendiskripsikan circuit secara Behavioural, Dataflow atau Structural

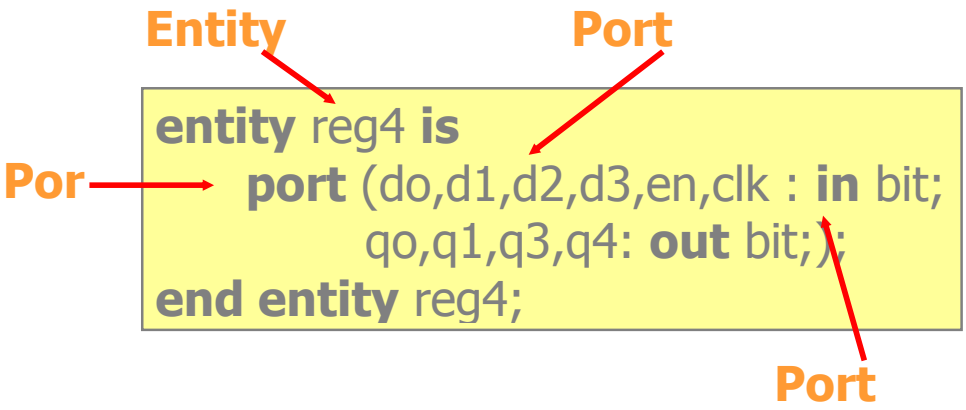
III. Dasar Teori

VHDL adalah kependekan dari **V**ery High Speed Integrated Circuit **H**ardware **D**escription **L**anguage, yaitu bahasa pemrograman yang digunakan untuk mendeskripsikan logic circuit yang dikehendaki

Secara umum struktur dari pemrograman VHDL terdiri atas dua bagian yaitu bagian **ENTITY** dan bagian **ARCHITECTURE.**

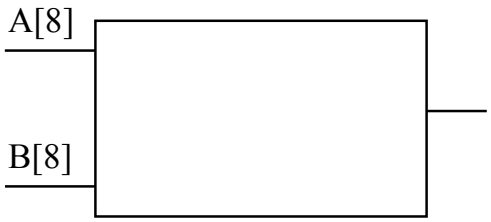


Bagian **ENTITY** menjelaskan spesifikasi pin-pin eksternal yang digunakan dari circuit atau rancangan yang akan dibuat.



Bagian **ARCHITECTURE** menjelaskan atau mewakili fungsi sesungguhnya dari circuit atau rangkaian.

Contoh :



8-bit Comparator

```
ENTITY compare IS
  PORT(a, b: IN bit_vector(0 TO 7);
       eq: OUT bit);
END compare;

ARCHITECTURE compare1 OF compare IS
BEGIN
```

Ada tiga cara untuk mendiskripsikan Circuit dalam VHDL:

1. Behavioural

Didesain berdasarkan Algorithma

2. Dataflow (RTL)

   Didesain berdasarkan alur register data

3. Structural

   Didesain berdasarkan perkomponen dan "merangkai komponen tersebut"

Contoh Deskripsi behavioural

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL
ENTITY eqcomp4 IS PORT(
     a, b:      IN std_logic_vector(3 DOWNTO 0);
    equals:  OUT std_logic);
END eqcomp4;
ARCHITETURE behavioral OF eqcomp4 IS
BEGIN
comp: PROCESS (a, b)
     BEGIN
        IF a = b then
            equals <= '1';
        ELSE
            equals <= '0';
        END IF;
     END PROCESS comp;
END behavioral;
```

Contoh Deskripsi DataFlow

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL
ENTITY eqcomp4 IS PORT(
     a, b:      IN std_logic_vector(3 DOWNTO 0);
    equals:  OUT std_logic);
END eqcomp4;
ARCHITETURE dataflow OF eqcomp4 IS
BEGIN
     equals <= '1' WHEN (a=b) ELSE '0';
END dataflow;
```

Contoh Deskripsi Structural

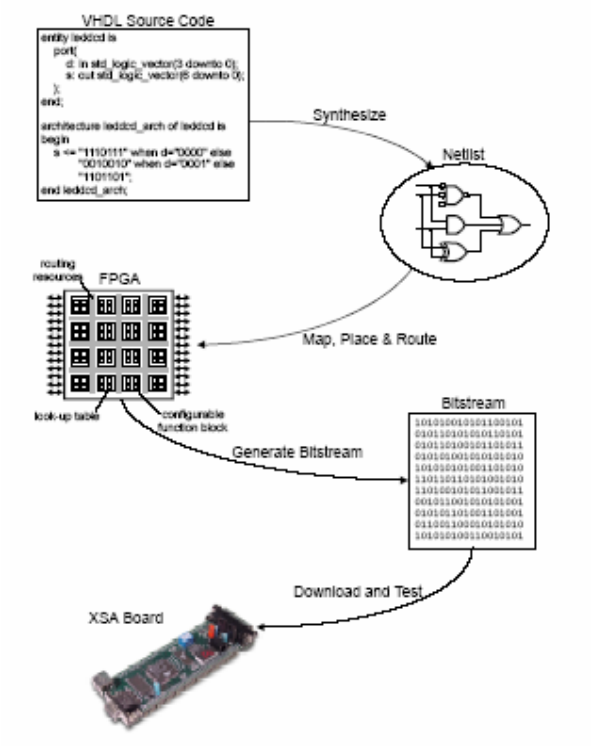
```
LIBRARY ieee;
```

```
USE ieee.std_logic_1164.ALL
ENTITY eqcomp4 IS PORT(
      a, b:       IN std_logic_vector(3 DOWNTO 0);
     equals:  OUT std_logic);
END eqcomp4;

USE work.gatespkg.ALL;
ARCHITETURE struct OF eqcomp4 IS
      SIGNAL x: std_logic_vector(0 TO 3);
BEGIN
      u0: xnor2 PORT MAP (a(0),b(0),x(0));
      u1: xnor2 PORT MAP (a(1),b(1),x(1));
      u2: xnor2 PORT MAP (a(2),b(2),x(2));
      u3: xnor2 PORT MAP (a(3),b(3),x(3));
      u4: and4 PORT MAP (x(0),x(1),x(2),x(3),equals);
END struct;
```

Tipe data yang ada dalam pemrograman VHDL yaitu :

Data	Value	Contoh
Bit	'1','0'	Q <= '1';
Bit_Vector	(array of bits)	DataOut <= "00010101";
Boolean	True, False	EQ <= True;
Integer	-2,-1,0,1,2,3 …	Count <= Count + 2;
Real	1.0, -1.0E5	V1 = V2 / 5.3;
Time	1us,7ns,100ps	Q <= '1' after 6 ns;
Character	'a','b','c',etc.	CharData <= 'X';
String	(Array of Char)	Msg <= "MEM:" & Addr

Pemrograman IC FPGA



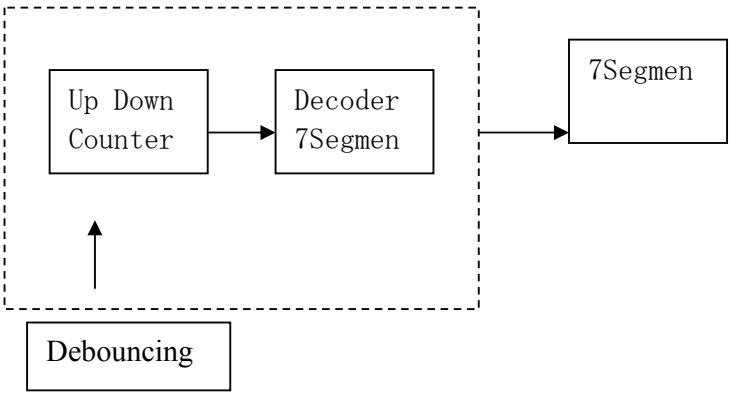
IV. Peralatan

1. 1 set PC yang dilengkapi dengan software ISE WebPack versi 6.1 atau lebih serta software ModelSim.
2. 1 development board XSA-50 + XSTENDBOARD
3. 1 power-supply +9V
4. 1 kabel data

V. Prosedur Percobaan

Gambaran Disain

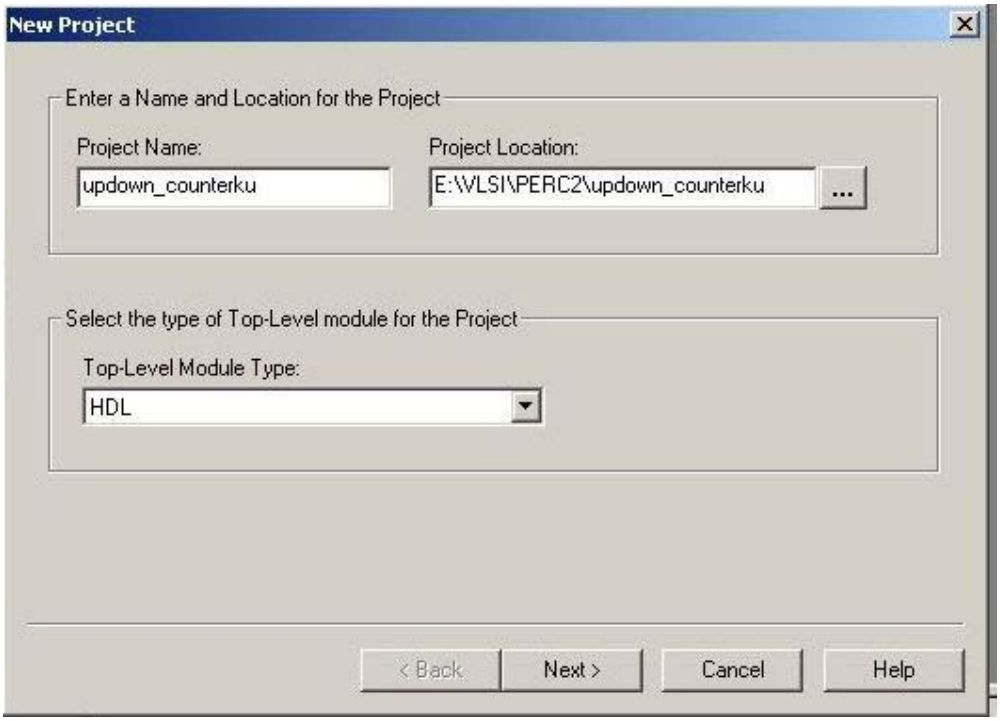
Pada percobaan ini akan dibuat sebuah counter up_down dan decoder seven segmen dengan menggunakan VHDL editor.



1. Jalankan software **ISE WebPACK** dengan mengklik ganda icon 

pada desktop.

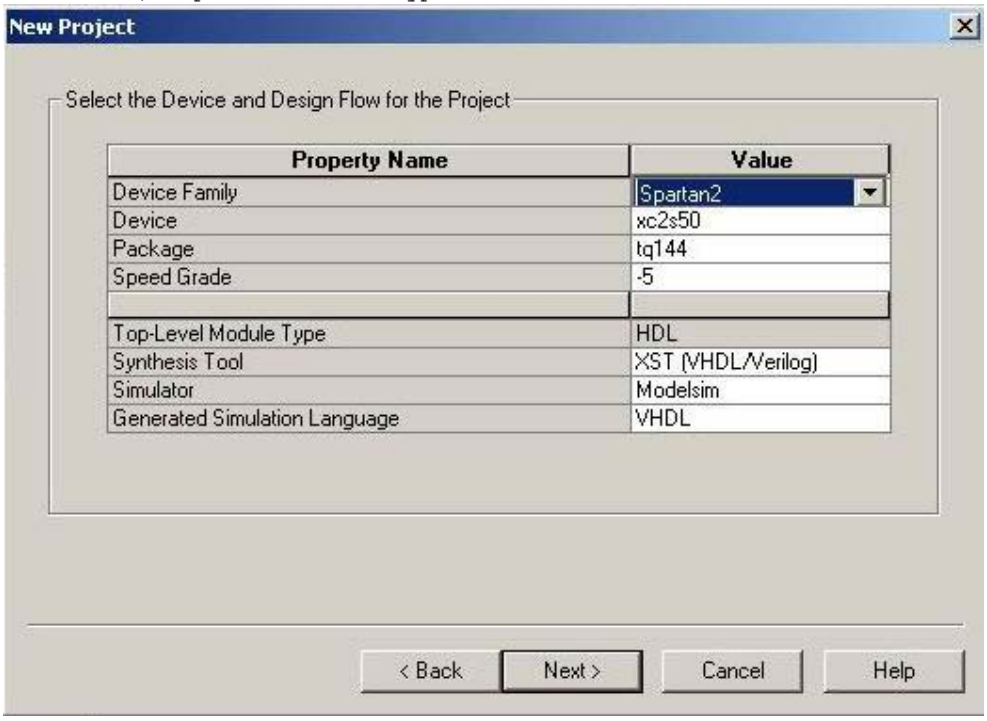
2. Buat **new project** dengan nama **"updown_counterku"** simpan di direktori **E:/VLSI/PERC2/**



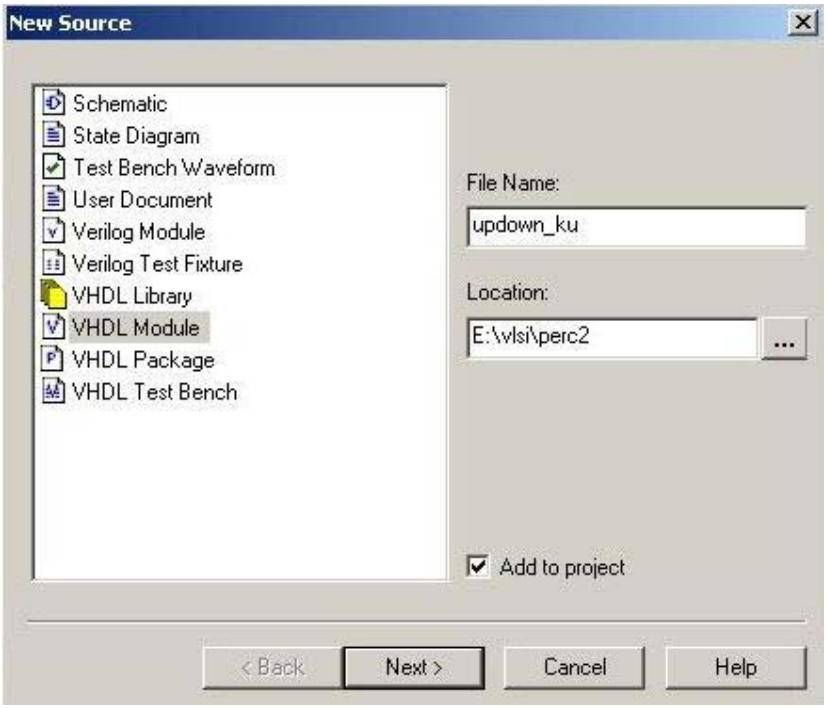
3. 6. Masukkan parameter seperti gambar berikut. Kemudian klik tombol NEXT sampai tiga kali, konfigurasi yang ada tidak perlu diubah, kemudian klik tombol FINISH.

`Device Family: Spartan2 , Device: Xc2s50, Package: tq144`

`Speed Grade: -5, Top-Level Modul Type: HDL`

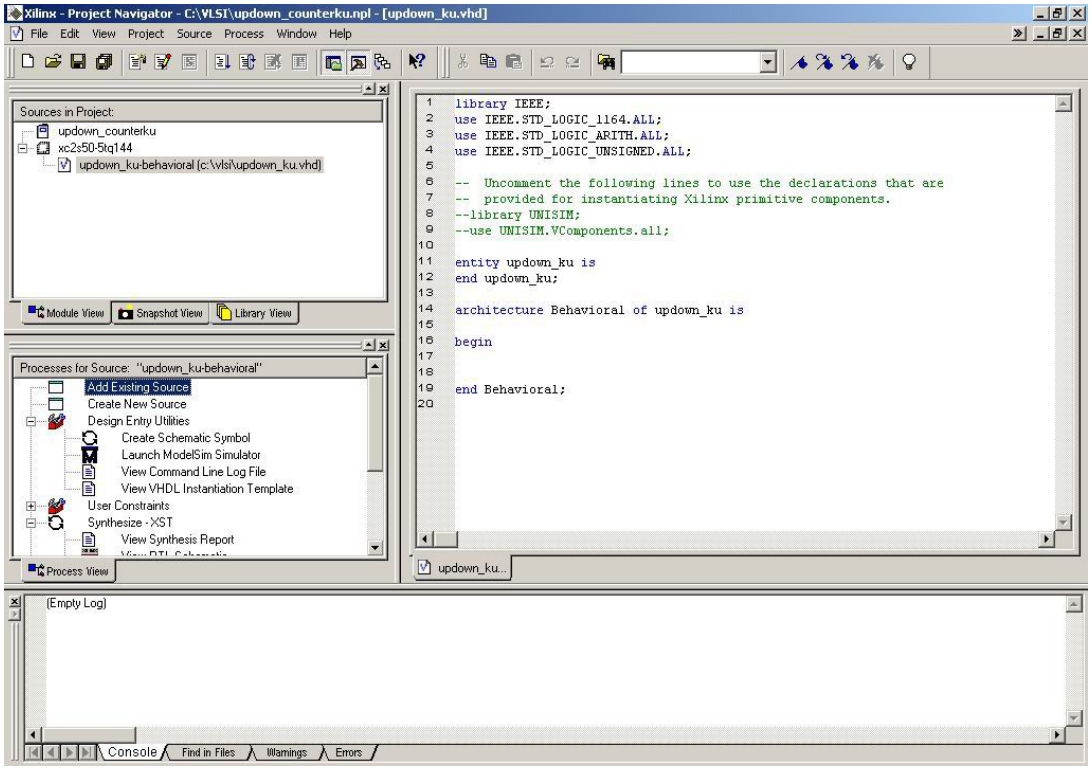


4. Buat **new source** jenis **VHDL Module** dengan nama **updown_ku**



Selanjutnya klik NEXT sampai FINISH

5. Maka anda akan mendapatkan jendela HDL editor

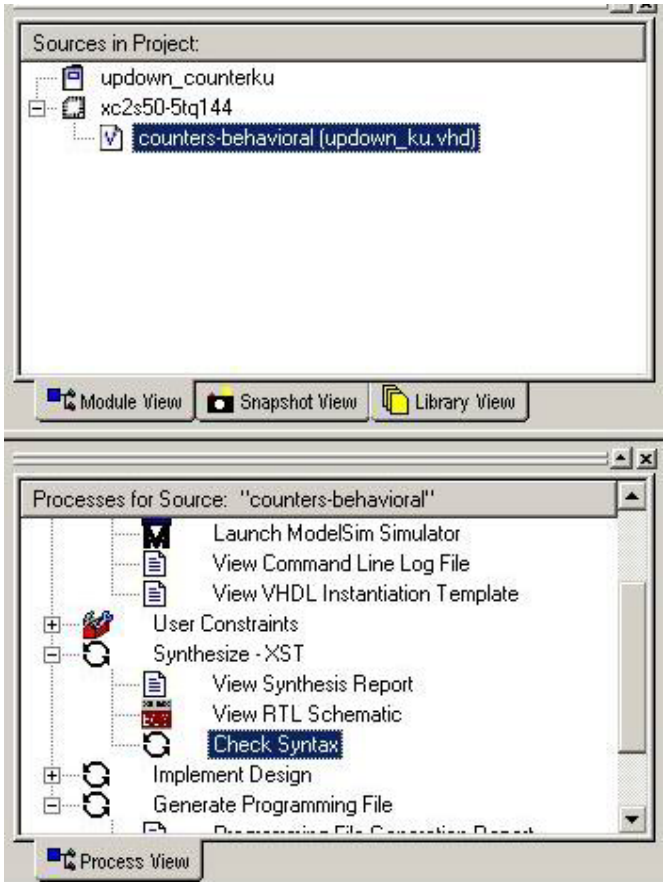


Selanjutnya ketikkan program berikut pada jendela HDL editor :

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

--  Uncomment the following lines to use the declarations that are
--  provided for instantiating Xilinx primitive components.
--library UNISIM;
--use UNISIM.VComponents.all;

ENTITY counters IS
   PORT(
      d      : IN    std_logic_vector(3 downto 0);
      pulsa     : IN    std_logic;
      clear   : IN    std_logic;
      load    : IN    std_logic;
      up_down : IN    std_logic;
      qd      : OUT   std_logic_vector(3 downto 0)
                  );
END counters;

ARCHITECTURE Behavioral OF counters IS
BEGIN
   -- An up/down counter
   PROCESS (pulsa, up_down)
      VARIABLE    cnt       : std_logic_vector(3 downto 0);
      VARIABLE    direction : std_logic_vector(3 downto 0);
   BEGIN
      IF (up_down = '1') THEN  --Generate up/down counter
         direction := "0001";
         ELSE
            direction := "1111";
```

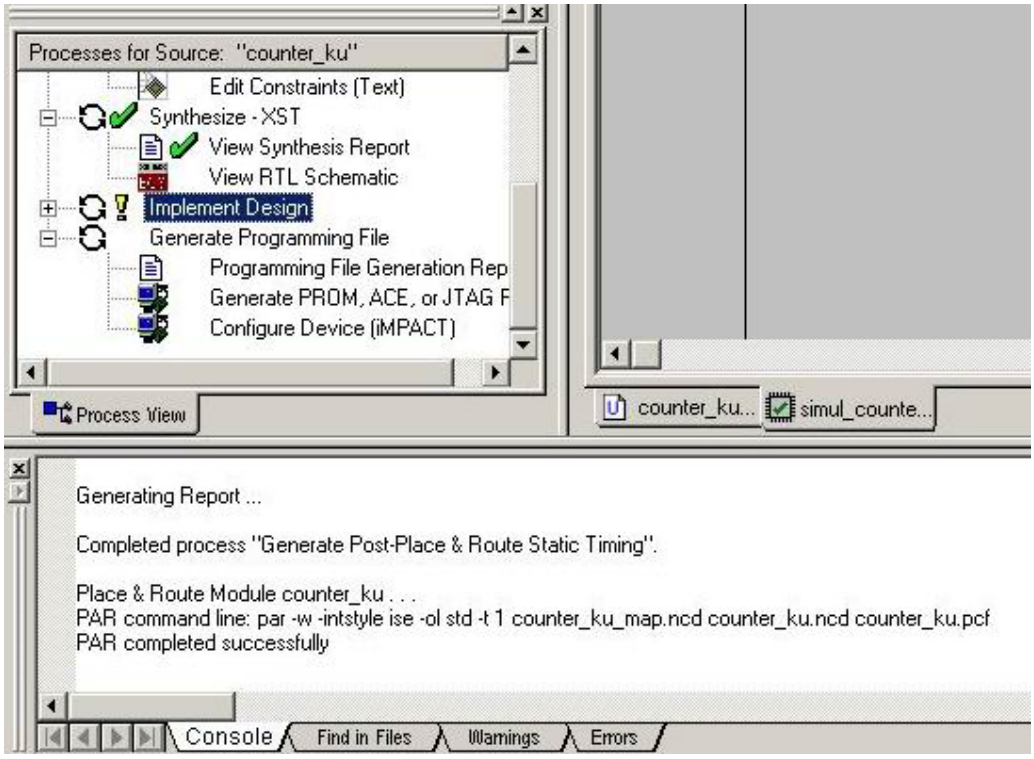
```
      END IF;
        IF (pulsa 'EVENT AND pulsa = '1') THEN
          IF (load = '1') THEN        --Generate loadable
            cnt := d;              --counter. Take these
            ELSE                    --lines out to increase performance.
            cnt := cnt + direction;
           END IF;
          --The following lines will produce a synchronous
          --clear on the counter
           IF (clear = '0') THEN
             cnt := "0000";
           END IF;
         END IF;
  qd <= cnt;   --Generate outputs
  END PROCESS;
END Behavioral;
```

6. Kemudian **SAVE** file dan periksa penulisan program VHDL dengan cara melakukan klik ganda pada **check syntax**



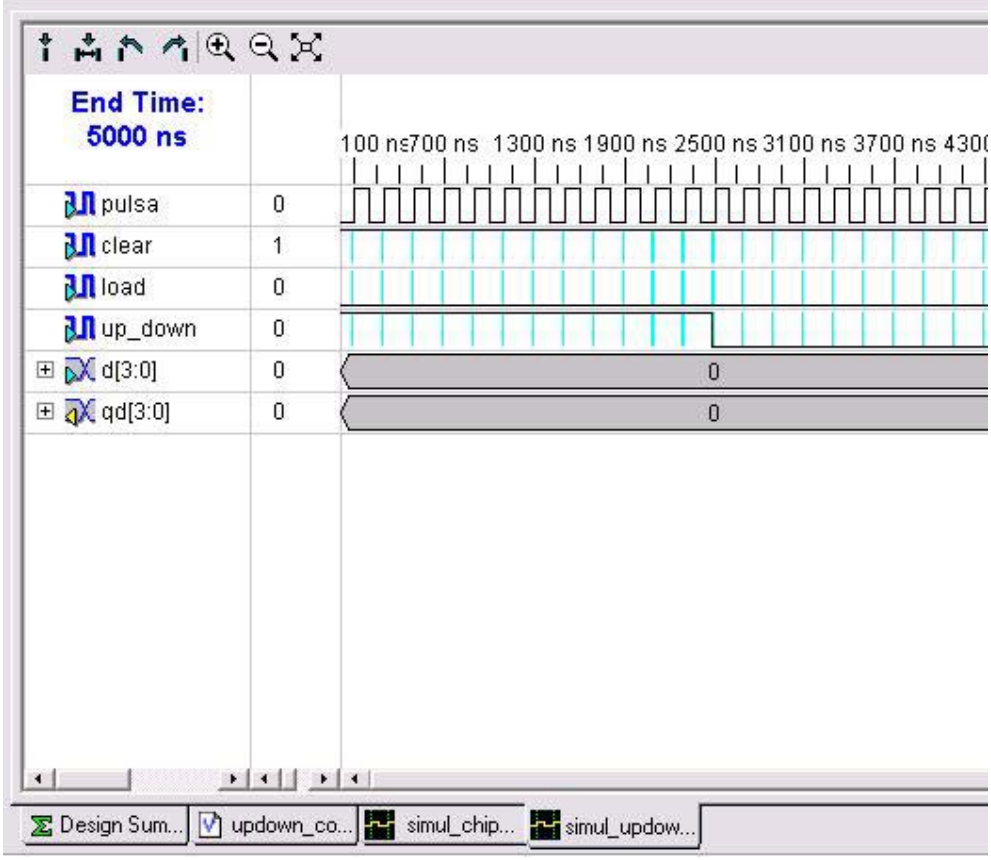
Ulangi langkah ini sampai tidak ada kesalahan, apabila sudah tidak ada error maka ikutilah langkah selanjutnya.

7. Kemudian **synthesize modul** dengan cara mengklik ganda pada **Synthesize-XST** serta **Implementasi Design** dengan cara mengklik ganda pada **Implement Design**.



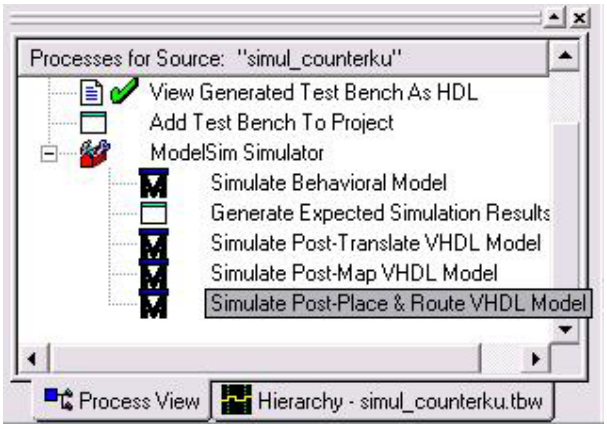
8. Buat **new source** untuk simulasi modul VHDL hasil perancangan , berilah nama file **simul_updown**

9. Berikanlah nilai logic pada parameter input *(yang berwarna biru)* sebelum melakukan simulasi. Setelah itu lakukan simulasi dan gambarkan hasil simulasi pada lembar laporan sementara, **Jangan lupa simpan file hasil simulasi**.



10. Selanjutnya lakukan simulasi FUNCTIONAL (Simulate Behavioral Model) dan simulasi TIMING (Simulate Post Place&Route VHDL Model).

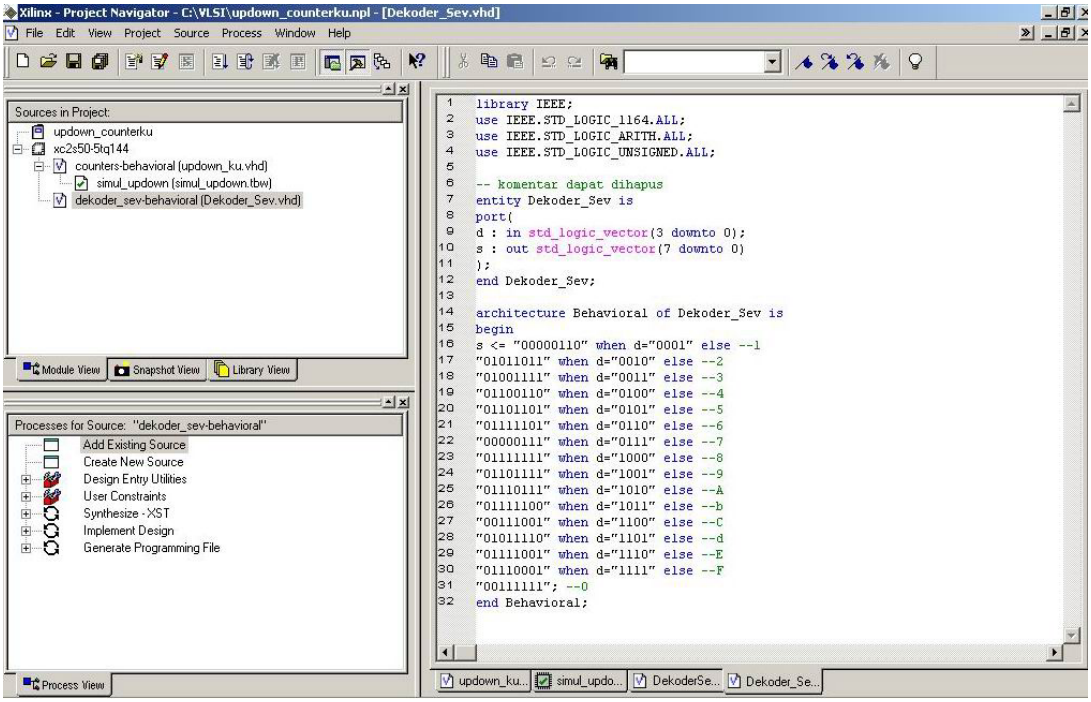
CATATAN : Perhatikan perbedaan hasilnya



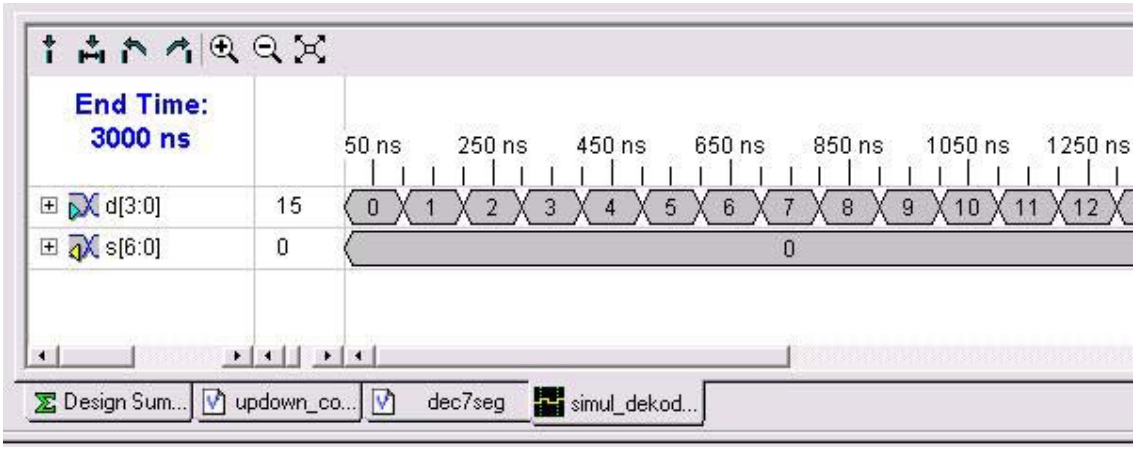
11. Setelah itu lanjutkan dengan membuat VHDL modul baru dengan cara membuat **new source,** beri nama modul ini **Dekoder_Sev**

12. Ketikkan listing program berikut pada jendela VHDL editor

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

-- komentar dapat dihapus
entity Dekoder_Sev is
port(
d : in std_logic_vector(3 downto 0);
s : out std_logic_vector(6 downto 0)
);
end Dekoder_Sev;

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



13. Dengan cara yang sama dengan **<u>langkah 6 s.d 10</u>** sebelumnya, **CHECK SYNTAX**, **SYNTHESIZE, IMPLEMENT DESIGN** dan **SIMULASIKAN** module dekoder ini. Berilah nama file simulasi **simul_dekoder.**

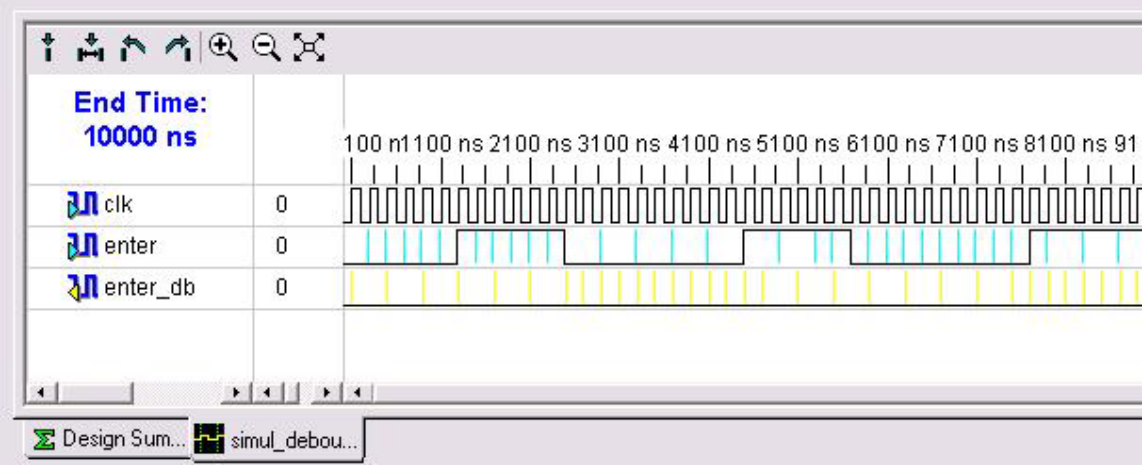


14. Setelah itu lanjutkan dengan membuat VHDL modul baru dengan cara membuat **new source,** beri nama modul ini **debouncing**

15. Ketikkan listing program berikut pada jendela VHDL editor

```
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 : in std_logic;
enter_db : out std_logic
);
end debouncing;
architecture Behavioral of debouncing is
signal A : 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;

enter_db <= not(A(2)) and A(1) and A(0);
end Behavioral;
```

16. Dengan cara yang sama dengan **langkah 6 s.d 10** sebelumnya, **CHECK SYNTAX**, **SYNTHESIZE, IMPLEMENT DESIGN** dan **SIMULASIKAN** module **debouncing** ini. Berilah nama file simulasi **simul_debouncing**



17. Buat Source VHDL baru yang berfungsi menghubungkan tiga komponen/blok diatas, beri nama **chip_updown**

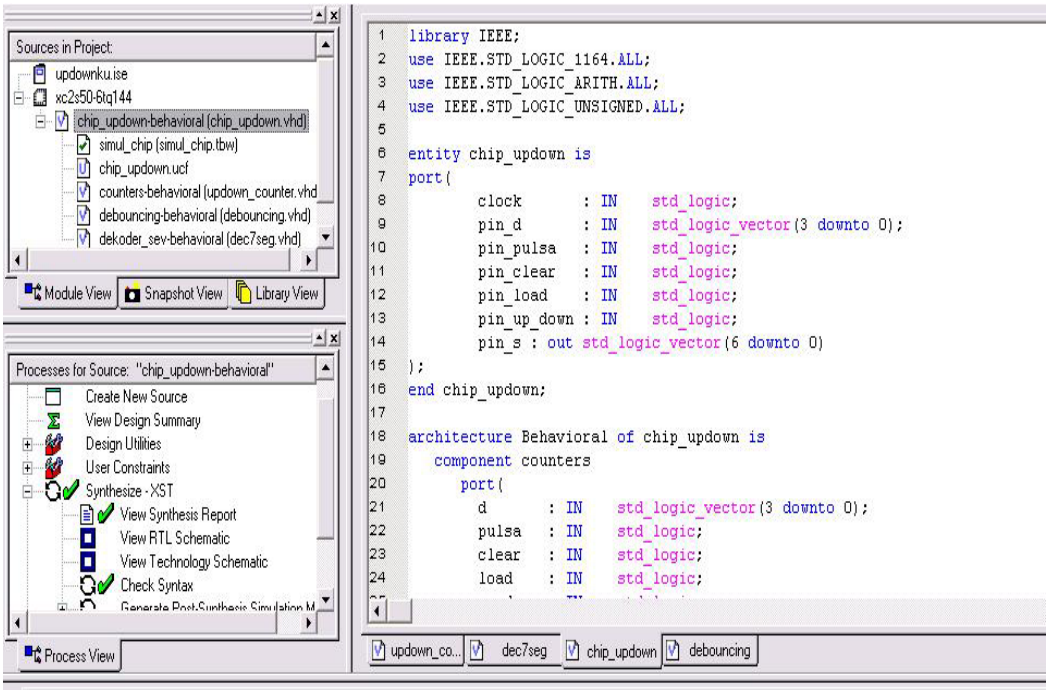
18. Ketikkan listing program berikut pada jendela VHDL editor

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

```
entity chip_updown is
port(
      clock                  : IN    std_logic;
      pin_d                  : IN    std_logic_vector(3 downto 0);
      pin_pulsa   : IN    std_logic;
      pin_clear   : IN    std_logic;
      pin_load    : IN    std_logic;
      pin_up_down : IN    std_logic;
                    pin_s : out std_logic_vector(6 downto 0)
);
end chip_updown;

architecture Structural of chip_updown is
          component counters
                    port(
                      d           : IN    std_logic_vector(3 downto 0);
      pulsa   : IN    std_logic;
      clear   : IN    std_logic;
      load    : IN    std_logic;
      up_down : IN    std_logic;
      qd      : OUT   std_logic_vector(3 downto 0)
                    );
          end component;

          signal out_counter : std_logic_vector(3 downto 0);

          component Dekoder_Sev
                    port(
                      d : in std_logic_vector(3 downto 0);
                      s : out std_logic_vector(6 downto 0)
                    );
          end component;

          component debouncing
                    port(
                      clk : in std_logic;
                      enter : in std_logic;
                      enter_db : out std_logic
                    );
          end component;
          signal s_pulsa  : std_logic;

begin
counter_ku:
counters port map
                              (
                               d => pin_d, pulsa => s_pulsa, clear => pin_clear,
                               load => pin_load, up_down => pin_up_down, qd => out_counter
                              );

decoder_ku:
Dekoder_Sev port map
                              (
                              d=> out_counter, s=> pin_s
                              );

debouncing_ku:
debouncing port map
                              (
```

```
                clk => clock, enter => pin_pulsa , enter_db => s_pulsa
                );

end Structural;
```

19. Selanjutnya **Simpan, Check Syntax** dan **Synthesized**



20. Kemudian tentukan pin-pin pada FPGA yang akan kita gunakan dengan mengedit file constraint dengan cara melakukan klik ganda pada **Edit Constraint (Text)**

Pilih pin-pin yang digunakan pada FPGA sesuai dengan datasheet dari XSTENDBoard untuk dapat menggunakan pushbutton, Barled, Switch maupun SevenSegment.

File ucf untuk mendefinisikan pin-pin IC

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

```
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;
```

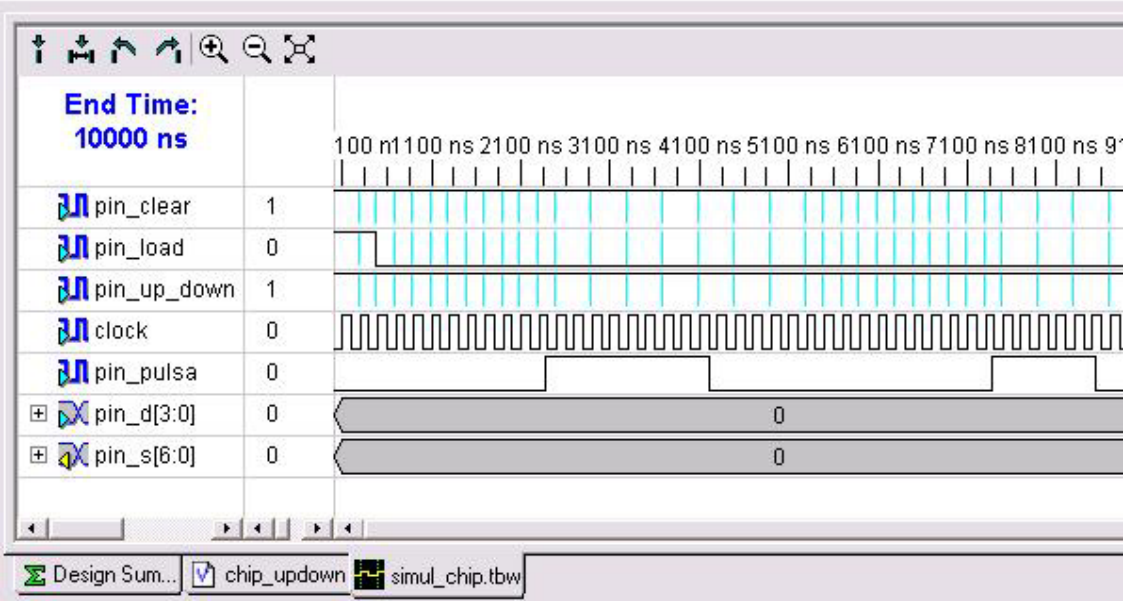
```
net ledtwo<6> loc=p48;
net ledtwo<7> loc=p38;

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 ledone<7> loc=p63;

net barled<1> loc=p68; # bargraph LED
net barled<2> loc=p44;
net barled<3> loc=p46;
net barled<4> loc=p49;
net barled<5> loc=p57;
net barled<6> loc=p62;
net barled<7> loc=p60;
net barled<8> loc=p67;
net barled<9> loc=p39;
net barled<10> loc=p59;

net pushsw<3> loc=p78; # pushbuttons
net pushsw<4> loc=p26;
net pushsw<5> loc=p23;

net dipsw<1> loc=p30; # DIP switches
net dipsw<2> loc=p58;
net dipsw<3> loc=p74;
net dipsw<4> loc=p75;
net dipsw<5> loc=p66;
net dipsw<6> loc=p77;
net dipsw<7> loc=p80;
net dipsw<8> loc=p79;
```

21. Edit Constraint file sebagai berikut:

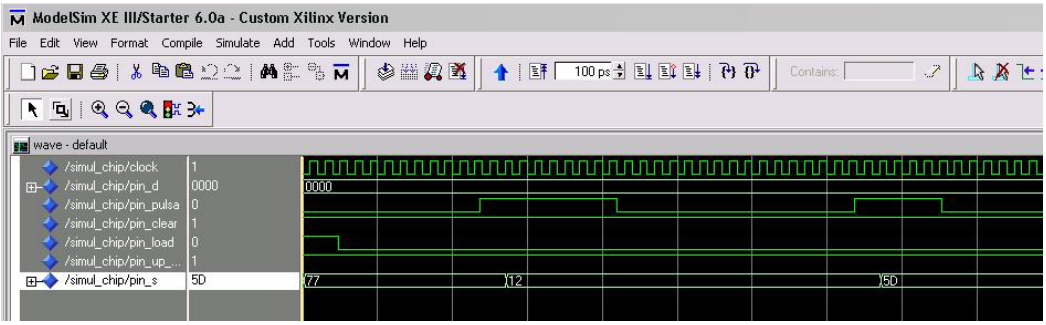
```
# INPUT
net pin_pulsa loc=p23; #pin CLOCK berada pada pin ke-23 pada FPGA  atau tombol S5 pada
XSTENBoard
net clock loc=p88;
net pin_clear loc=p79;
net pin_load loc=p77;
net pin_up_down loc=p80;
net pin_d<0> loc=p30;
net pin_d<0> loc=p74;
net pin_d<0> loc=p75;
net pin_d<0> loc=p66;
# OUTPUT
net pin_s<0> loc=p64; # leftmost 7-segment LED
net pin_s<1> loc=p65;
net pin_s<2> loc=p76;
net pin_s<3> loc=p50;
net pin_s<4> loc=p51;
net pin_s<5> loc=p54;
net pin_s<6> loc=p56;
```

22. Kemudian **Implement Design**

23. Buat **new source** untuk simulasi modul VHDL hasil perancangan , berilah nama file **simul_chip**



24. Selanjutnya lakukan **SIMULASI** FUNCTIONAL (Simulate Behavioral Model) dan simulasi TIMING (Simulate Post Place&Route VHDL Model).dan **ambil data simulasi**



25. **Generate Programming File**

26 Upload file *.bit yang dihasilkan dengan menggunakan program **GXSLOAD**

27. Ambil data hasil percobaan

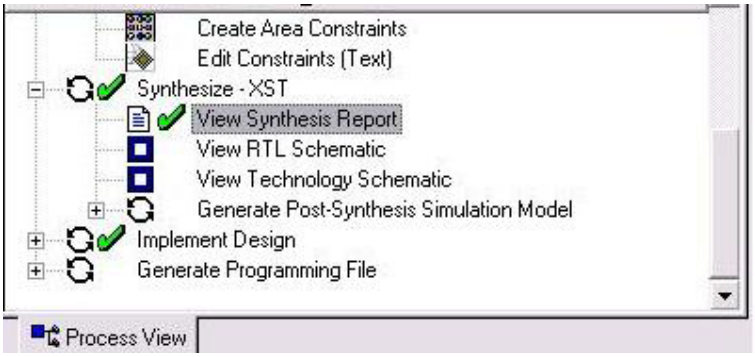
VI. Hasil Pengujian

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

1. Rubahlah module VHDL yang ada sehingga counter naik atau turun sebesar 2 desimal.
2. Dari synthesis report TOPLEVEL, berapakah jumlah slice yang dipakai serta berapa kecepatan frekuensi maksimum?
3. Jelaskan kenapa terdapat perbedaan hasil yang sangat besar antara simulasi Functional dengan simulasi timing saat simulasi :
   (1) **simul_updown**
   (2) **simul_chip**
4. Kenapa modul VHDL **chip_updown** dikategorikan menggunakan deskripsi **structural** ?

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