

# Praktikum Sistem Digital Lanjut

## Percobaan 1: Pengenalan Xilinx ISE dan Sistem Masukan-Keluaran

### 1 Tujuan dan Sasaran

Kegiatan praktikum ini bertujuan untuk mengenalkan praktikan software Xilinx ISE Webpack untuk mengembangkan dan mengimplementasikan sistem digital terprogram di atas board Spartan-3E FPGA Starter Kit. Sistem digital yang akan dirancang adalah sistem masukan-keluaran menggunakan modul I/O tombol tekan, saklar geser, *rotary-knob* dan LED.

Sasaran kegiatan praktikum adalah:

1. Praktikan dapat membuat sebuah proyek sistem digital;
2. Praktikan dapat membuat file desain kombinasional untuk mengontrol nyala LED dengan menggunakan masukan tombol tekan, saklar geser dan *rotary-knob*;
3. Praktikan dapat mengkompilasi project: sintesis dan implementasi desain;
4. Praktikan dapat menganalisis hasil implementasi;
5. Praktikan dapat mendownload desain file ke board Starter Kit menggunakan Impact ;
6. Praktikan dapat menganalisis perilaku masukan-keluaran desain di board Starter Kit;

Sumber referensi yang bisa digunakan:

1. UG230: Spartan-3E FPGA Starter Kit Board User Guide, Xilinx, June 2008
2. DS312: Spartan-3E FPGA Family Data Sheet, Xilinx, August 2009
3. Spartan-3E Starter Board Schematic, Digilent, Feb 2006
4. Xilinx ISE Design Suite 11 Software Manual, Xilinx, 2009
5. Verilog Tutorial (online): <http://www.asic-world.com/verilog/veritut.html>

### 2 Alat, Software Bantu dan Komponen

Alat dan bahan yang digunakan adalah:

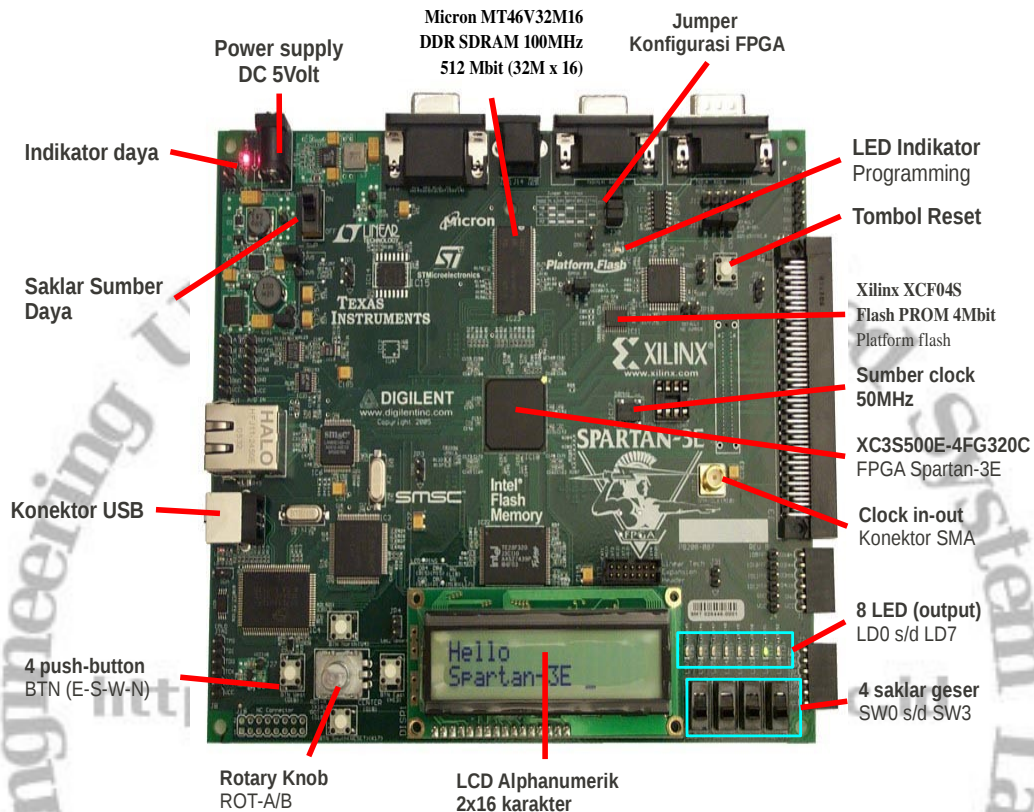
1. Board Starter Kit Spartan-3E berbasis Xilinx FPGA XC3S500E-4FG320C;  
Modul (komponen) I/O yang akan digunakan dalam praktikum:
  - a) 4 buah tombol push-button (BTN North, BTN East, BTN South, BTN West);
  - b) 1 buah rotary-knob (ROT-A/B);
  - c) 4 buah saklar geser (SW0, SW1, SW2, SW3);
  - d) 8 buah LED (LD0-7);
2. Kabel USB dengan konektor tipe-B;
3. Adaptor sumber daya DC 5 Volt;
4. Software Xilinx ISE Webpack 11.1;

### 3 Dasar Teori

Di bab ini akan dijelaskan tentang board Starter Kit Spartan-3E dan Xilinx ISE Webpack.

### 3.1 Board Starter Kit Spartan-3E

Board starter kit Spartan-3E (kode produk: HW-SPAR3E-SK-US-G ) akan digunakan dalam praktikum ini. Board ini menggunakan Xilinx FPGA Spartan-3E (XC3S500E-4FG320C) yang mempunyai 500K gerbang dan tersusun atas komponen-komponen (Gambar 1). Dalam praktikum 1 ini, komponen yang akan digunakan adalah keluaran LED LD0-7, masukan saklar geser SW0-3, dan tombol BTN (E-S-W-N). Selanjutnya board ini disebut Starter Kit.



Gambar 1: Board Starter Kit Xilinx Spartan-3E beserta komponen-komponennya

### 3.2 Xilinx ISE Webpack 11.1

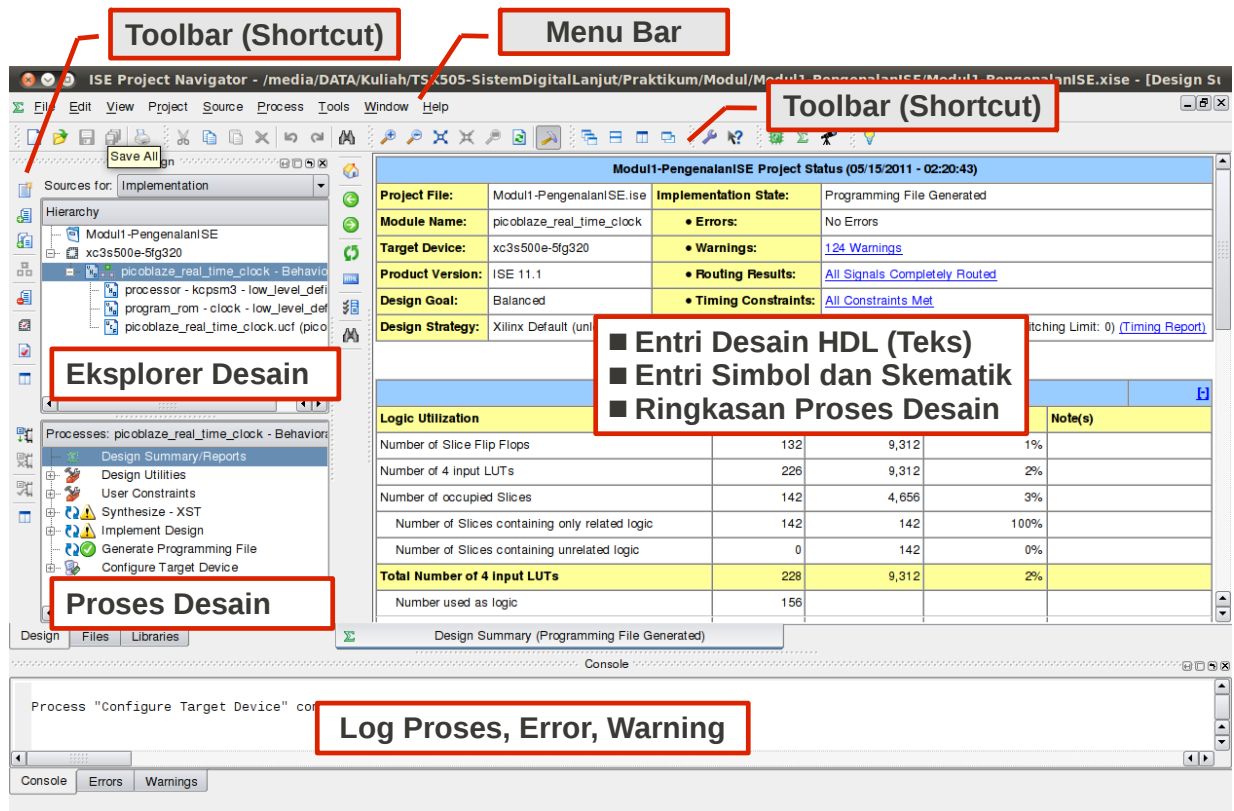
Software Xilinx ISE Webpack 11.1 akan digunakan sebagai GUI untuk merancang problem sistem digital yang diinginkan. Selanjutnya Xilinx ISE Webpack 11.1 disebut sebagai ISE. ISE telah terinstall di komputer masing-masing.

ISE berisi tool-tool untuk mengembangkan rancangan sistem digital terprogram. Tool yang akan digunakan dalam praktikum adalah:

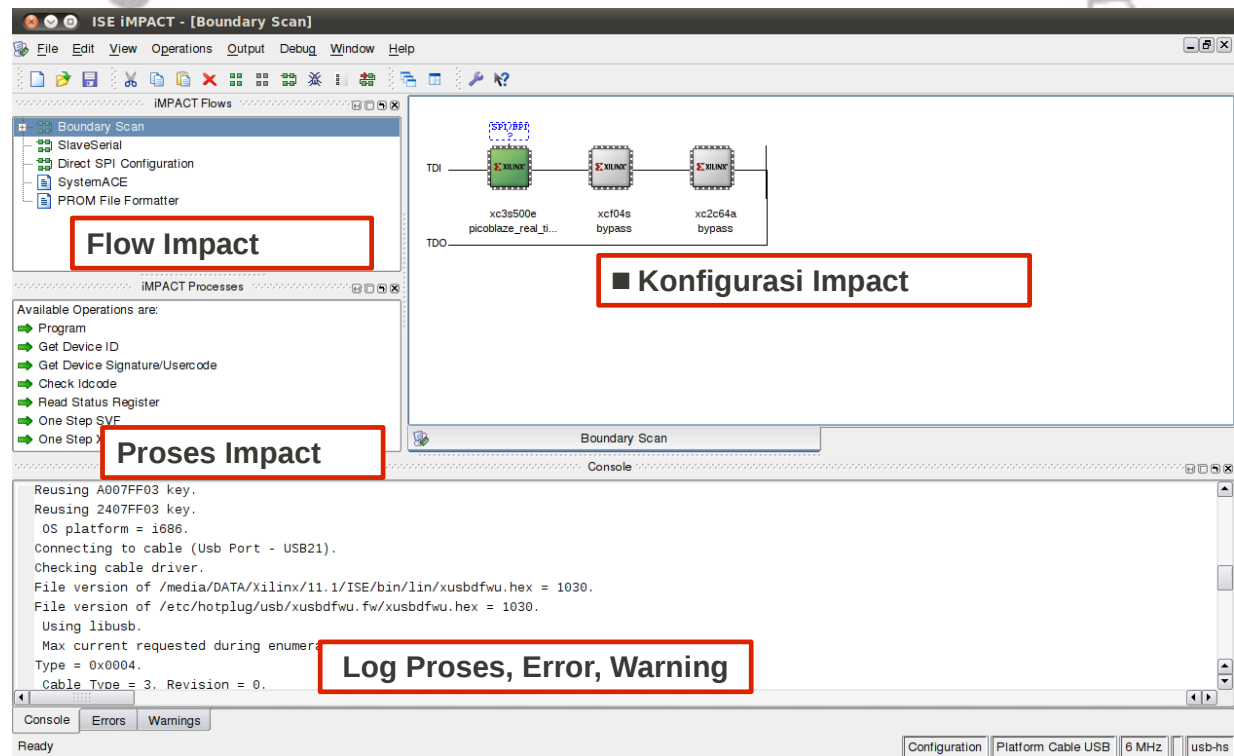
1. **ISE Project Navigator** digunakan untuk membuat proyek baru, memasukkan entry desain (HDL, skematik), kompilasi desain (sintesis), mengimplementasikan desain dan membangkitkan file programming (konfigurasi FPGA). Tool lain dari ISE dipanggil dari Project Navigator ini;
2. **ISE iMPACT** digunakan untuk menuliskan file konfigurasi FPGA ke board Starter Kit;

GUI dari program ISE Project Navigator diperlihatkan dalam Gambar 2.

GUI dari program ISE iMPACT diperlihatkan dalam Gambar 3.



Gambar 2: ISE Project Navigator GUI: Eksplorer, Proses, Entry dan Log



Gambar 3: ISE iMPACT: Konfigurasi Boundary Scan (JTAG)

### 3.3 Metodologi Rancangan dengan ISE

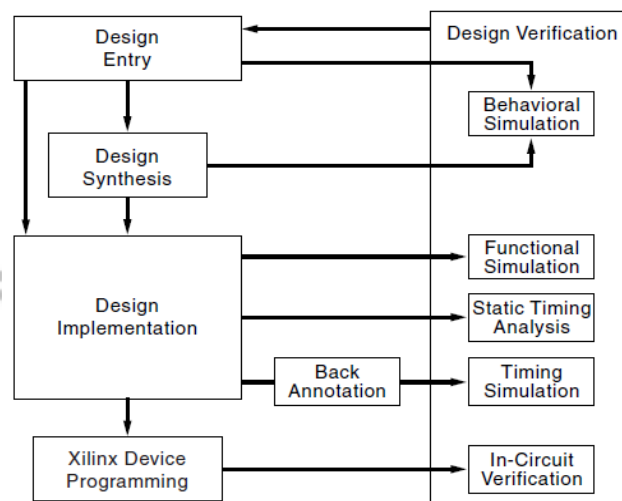
Perancangan sistem digital menggunakan ISE mempunyai metodologi (Gambar 4). Metodologi tersebut meliputi tahapan sebagai berikut:

1. **Entri desain.** Program HDL (*Hardware Description Language*) dibuat untuk memenuhi kebutuhan dan konstrain sistem yang ingin dirancang. Program ini

mendeskrripsikan struktural dan perilaku sistem. HDL yang dikenal ada 3, yaitu VHDL, Verilog dan Altera HDL. Praktikan dapat memilih bahasa HDL yang akan digunakannya. Dalam modul praktikum, bahasa Verilog akan digunakan. Sebelum pelaksanaan praktikum, praktikan **harus** telah menguasai sintaks dan struktur program HDL yang akan digunakannya, baik Verilog atau VHDL;

2. **Sintesis.** Dari desain HDL, skematik RTL (Register Transfer Level) dibangkitkan oleh ISE sesuai teknologi yang digunakan (Xilinx FPGA). Salah satu tujuan praktikum adalah melihat skematik RTL dari desain yang telah ditentukan;
3. **Implementasi.** Tahap meliputi *translating*, *mapping/fitting* dan *placing&routing* untuk mengimplementasikan desain ke teknologi device Xilinx yang sesuai. Desain dioptimasi agar memenuhi kebutuhan fungsional dan konstrainnya (*power*, *area*, *speed*). Hasil implementasi adalah berupa file konfigurasi FPGA (dengan ekstensi \*.bit) yang siap untuk diprogramkan ke device XC3S500E;
4. **Pemrograman/download** ke device Xilinx. Tahap ini dilakukan menggunakan program IMPACT. File konfigurasi didownload ke device melalui interface USB;

Dalam setiap tahap, verifikasi dilakukan untuk memastikan kebutuhan dan konstrain terpenuhi.



Gambar 4: Metodologi desain sistem digital menggunakan ISE: entri desain, sintesis, implementasi dan pemrograman

### 3.4 Tipe File yang Digunakan

Suatu proyek desain dapat mempunyai beberapa file desain, dengan satu file yang menjadi modul paling atas (*top module*), sedangkan lainnya sebagai modul komponen. ISE mengenal paling tidak 4 tipe file berikut:

1. File teks HDL (ekstensi \*.v untuk verilog, \*.vhd untuk VHDL)

Satu file HDL mendefinisikan satu struktur modul komponen (entitas) dan perilaku modul tersebut. Misalnya: satu modul full-adder 1-bit (FA) akan terdiri atas 3 masukan ( $x$ ,  $y$  dan  $c_{in}$ ) dan 2 keluaran ( $sum$ ,  $c_{out}$ ). Perilaku FA adalah  $sum = x \oplus y \oplus c_{in}$  dan  $c_{out} = xy + (x + y)c_{in}$

Satu modul dapat menjadi *top module* dan menggunakan modul lain sebagai komponen penyusunnya.

2. File skematik (ekstensi \*.sch)

File ini merupakan alternatif masukan desain secara grafis. Simbol-simbol modul komponen saling diinterkoneksi untuk menghasilkan satu modul baru.

### 3. File definisi pin/ *User Constraint File* (ekstensi \*.ucf)

File ini mendefinisikan konstrain yang harus dipenuhi oleh desain, meliputi konstrain waktu/speed, daya/power dan penempatan/*placement* pad logika di pin XC3S500E. Misalnya: pin LD-7 disambungkan dengan pin F9 FPGA, dengan TTL low-voltage

`NET "led<7>" LOC = "F9" | IOSTANDARD = LVTTTL ;`

### 4. File konfigurasi/programming

- a) File konfigurasi bitstream FPGA yang bisa diprogram langsung lewat JTAG (ekstensi \*.bit). File ini dibangkitkan dari ISE Project Navigator;
- b) File PROM dan konfigurasi FPGA yang akan disimpan di flash secara permanen dan digunakan dalam mode Slave-Serial (ekstensi \*.mcs). File ini dibangkitkan dengan PROM File Formatter di program ISE iMPACT;

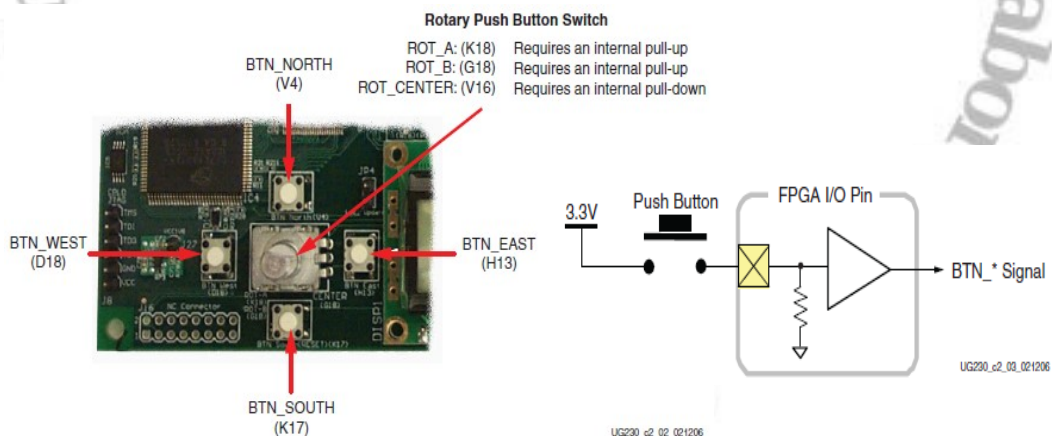
## 3.5 Komponen Board Starter Kit

Komponen Starter Kit yang digunakan dalam kegiatan praktikum 1 ini adalah:

1. 4 buah tombol tekan (BTN North, BTN East, BTN South, BTN West);
2. 1 buah rotary-knob (ROT-A/B);
3. 4 buah saklar geser (SW0, SW1, SW2, SW3);
4. 8 buah LED (LD0-7);

### 3.5.1 Tombol Tekan

Di Starter Kit terdapat 4 tombol tekan yang dapat digunakan sebagai masukan (Gambar 5). Keluaran sinyal BTN\_\* akan '1' (high) saat tombol ditekan.



Gambar 5: Tombol tekan dan rotary-knob beserta koneksi pinnya di FPGA. Push-button akan menghasilkan keluaran '1' saat ditekan (sumber: UG230)

Konstrain push-button didefinisikan di file UCF meliputi koneksi pin I/O, resistor pull-down dan tegangan TTL low-voltage, sebagai berikut:

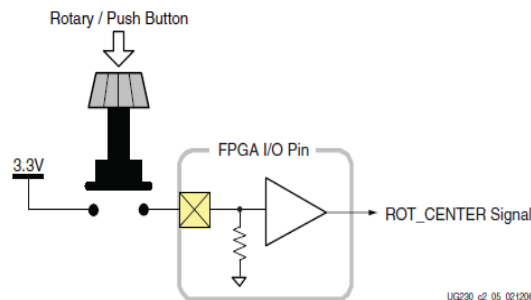
```
NET "BTN_EAST" LOC = "H13" | IOSTANDARD = LVTTTL | PULLDOWN ;
NET "BTN_NORTH" LOC = "V4" | IOSTANDARD = LVTTTL | PULLDOWN ;
NET "BTN_SOUTH" LOC = "K17" | IOSTANDARD = LVTTTL | PULLDOWN ;
NET "BTN_WEST" LOC = "D18" | IOSTANDARD = LVTTTL | PULLDOWN ;
```

Listing 1: Masukan UCF untuk komponen push-button



### 3.5.2 Rotary-knob: Tombol Tekan dan Encoder Putar

Rotary-knob berisi tombol tekan dan enkoder putar yang digunakan sebagai masukan. Dalam praktikum ini, hanya tombol tekan yang digunakan. Tombol tekan ini menghasilkan sinyal keluaran ROT\_CENTER dengan operasi (Gambar 1).



Gambar 6: Operasi rotary-knob: saklar tekan (sumber: UG230)

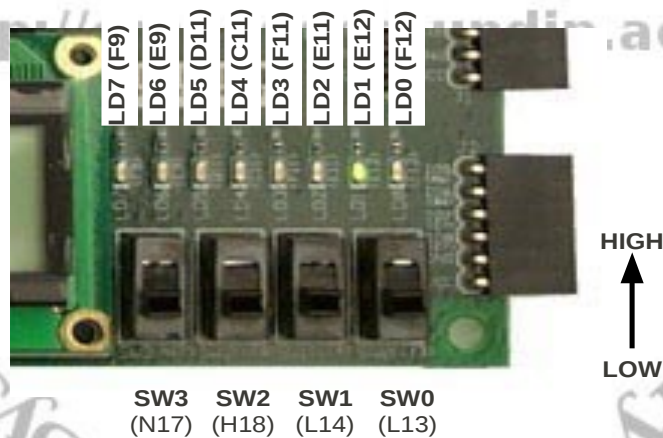
Konstrain rotary-knob didefinisikan di file UCF meliputi koneksi pin I/O, resistor pull-down dan tegangan TTL low-voltage, sebagai berikut:

```
NET "ROT_CENTER" LOC = "V16" | IOSTANDARD = LVTTTL | PULLDOWN ;
```

Listing 2: Masukan UCF untuk komponen tombol tekan rotary-knob

### 3.5.3 Saklar Geser

Starter kit mempunyai 4 saklar geser sebagai masukan (Gambar 7).



Gambar 7: Masukan saklar geser (SW3-0) dan keluaran LED (LD7-0).  
LED menyala kalau masukannya '1'

Konstrain saklar geser didefinisikan di file UCF meliputi koneksi pin I/O, resistor pull-up dan tegangan TTL low-voltage, sebagai berikut:

```
NET "SW<0>" LOC = "L13" | IOSTANDARD = LVTTTL | PULLUP ;  
NET "SW<1>" LOC = "L14" | IOSTANDARD = LVTTTL | PULLUP ;  
NET "SW<2>" LOC = "H18" | IOSTANDARD = LVTTTL | PULLUP ;  
NET "SW<3>" LOC = "N17" | IOSTANDARD = LVTTTL | PULLUP ;
```

Listing 3: Masukan UCF untuk komponen saklar geser

Resistor PULL-UP mendefinisikan nilai input '1' saat terjadi transisi (nilai output floating).

### 3.5.4 LED


Starter kit mempunyai 8 buah LED sebagai keluaran (Gambar 7).

Konstrain LED didefinisikan di file UCF meliputi koneksi pin I/O, slew-rate dan arus keluaran, sebagai berikut:

```
NET "LED<7>" LOC = "F9" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<6>" LOC = "E9" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<5>" LOC = "D11" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<4>" LOC = "C11" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<3>" LOC = "F11" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<2>" LOC = "E11" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<1>" LOC = "E12" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;  
NET "LED<0>" LOC = "F12" | IOSTANDARD = LVTTTL | SLEW = SLOW | DRIVE = 8 ;
```

Listing 4: Masukan UCF untuk komponen LED

## 4 Kegiatan Praktikum

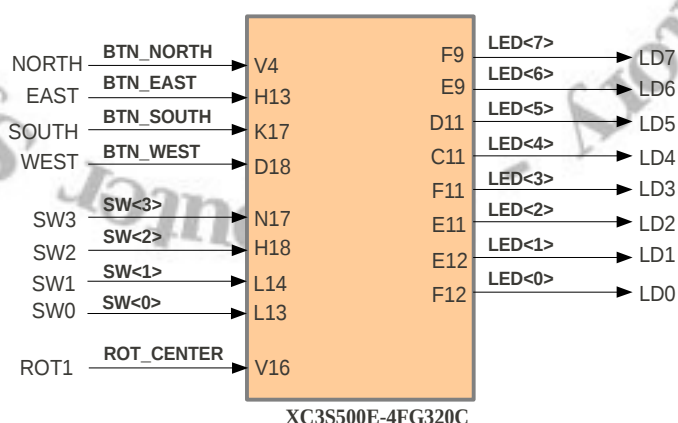
Kegiatan praktikum dilakukan untuk memenuhi kebutuhan desain yang diinginkan. Setiap tahap dilakukan berdasarkan cara kerja yang diuraikan dalam Subbab 4.2. Hasil kegiatan praktikum yang ditandai dengan ikon  dituliskan dalam Lembar Isian Kegiatan. Laporan akhir disusun dengan menyertakan hasil kegiatan praktikum.

### 4.1 Kebutuhan Desain

Sistem yang akan dirancang adalah sistem masukan-keluaran dengan struktur dan perilakunya seperti yang dijabarkan dalam subbab ini.

#### 4.1.1 Diagram Blok

Gambar 8 menunjukkan diagram sistem masukan-keluaran yang hendak dirancang. Diagram tersebut memperlihatkan interkoneksi antara device masukan, FPGA dan device keluaran. Nama sinyal masukan dan keluaran diberikan di atas garis panah, sedangkan pinout FPGA untuk sinyal tersebut diberikan sebagai nama pin. Nama sinyal dan pin akan digunakan sebagai masukan file UCF.



Gambar 8: Diagram blok rancangan sistem masukan-keluaran

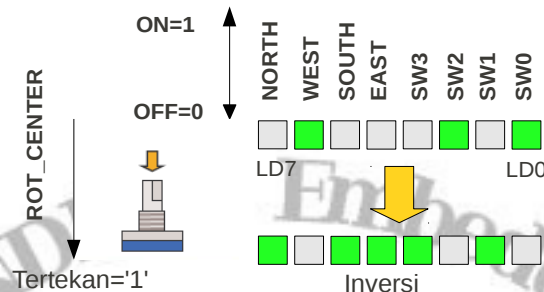
#### 4.1.2 Perilaku Sistem

Perilaku sistem yang diinginkan adalah sebagai berikut (Gambar 9):

- Saat keadaan rotary-knob UP/tidak tertekan (ROT\_CENTER=0), penyalan LED mengikuti keadaan saklar geser dan tombol tekan. Misalnya: saat SW0=SW2=0 maka

LD0 dan LD2 akan menyala (bernilai 1). Saat tombol WEST ditekan LD6 akan menyala (bernilai 1);

- Saat keadaan rotary-knob DOWN/tertekan ( $ROT\_CENTER=1$ ), penyalan LED merupakan inversi dari keadaan saklar geser dan tombol tekan. Misalnya: saat  $SW0=SW2=0$  maka LD0 dan LD2 akan mati, yang lainnya menyala. Saat tombol WEST ditekan LD6 akan mati (bernilai 0);



Gambar 9: Perilaku sistem yang ditunjukkan oleh LED terhadap masukannya



## 4.2 Langkah Kerja

Kegiatan praktikum meliputi hal-hal berikut:

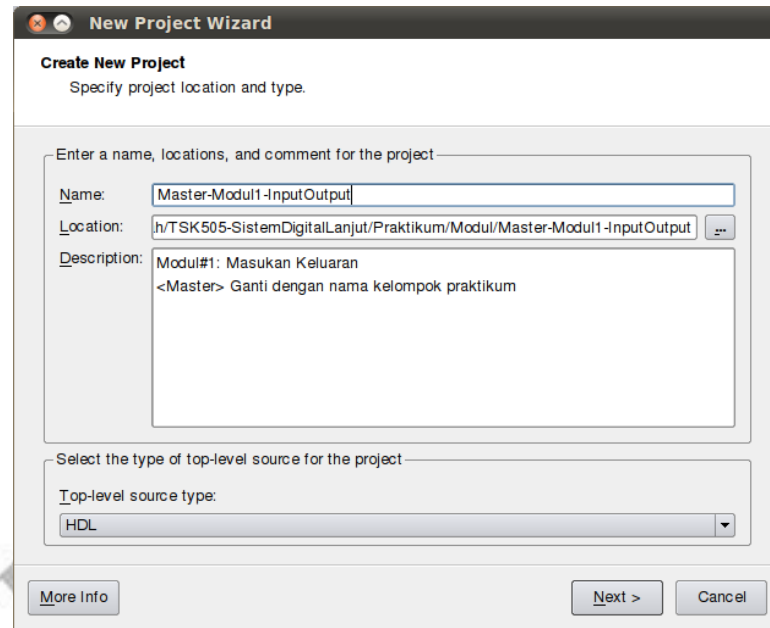
1. Membuat proyek;
2. Menambahkan file desain ke dalam proyek, yaitu file \*.v dan \*.ucf;
3. Mengkompilasi proyek: sintesis dan implementasi;
4. Membangkitkan file programming \*.bit;
5. Mengkonfigurasi (memprogram) device target FPGA menggunakan iMPACT;

### 4.2.1 Membuat Proyek

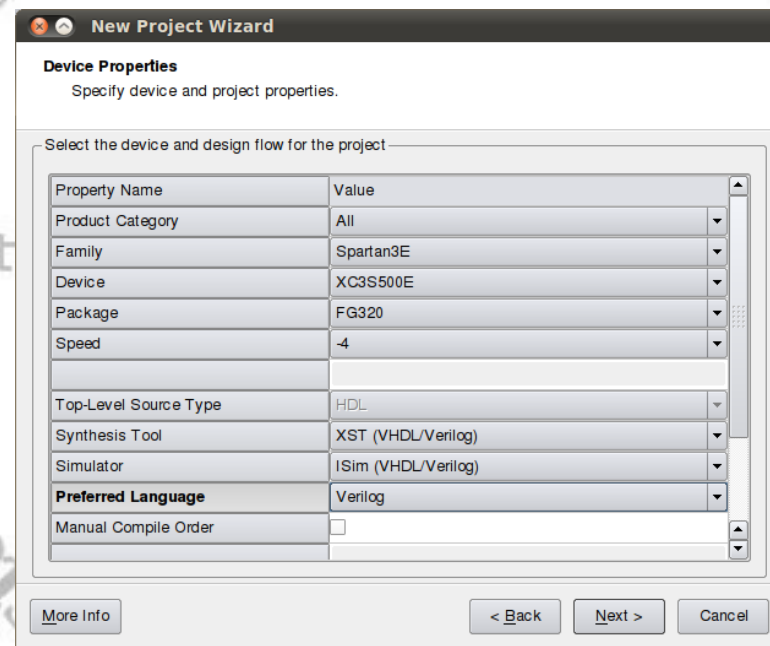
Berikut langkah-langkah membuat proyek baru di ISE:

1. Jalankan program ISE Project Navigator (jika belum dijalankan);
2. Pilih menu **File**→**New Project** (Alt+F W). Dialog pop-up **New Project Wizard** akan muncul (Gambar 10). Ketikkan field **Name** dengan nama proyek (Format: <nama\_kelompok>-Modul1-InputOutput). Browse lokasi proyek/**Location** (folder \$HOME\_DIR/<nama\_kelompok>). Isi field **Description** dengan penjelasan tentang proyek. Tipe source top-level menggunakan HDL.  
 Tuliskan field **Name**, **Location** dan **Description** dalam lembar kegiatan;
3. Klik tombol **Next**. Jendela **Device Properties** muncul (Gambar 11). Pilih **Family** (Spartan3E), **Device** (XC3S500E), **Package** (FG320) dan **Speed** (-4). Properti ini adalah device-dependent (Starter Kit menggunakan FPGA XC3S500E-4FG320C), jadi harus dimasukkan dengan benar. Set **Preferred Language** dengan Verilog.  
 Catat field Family, Device, Package dan Speed ini dalam lembar kegiatan;
4. Klik tombol **Next**, jendela **Create New Source** muncul. Klik tombol **Next**, jendela **Add Existing Sources** muncul. Klik tombol **Next**, jendela **Project Summary** muncul. Pastikan ringkasan proyek sesuai dengan yang tercatat di lembar kegiatan. Klik tombol **Finish** untuk mengakhiri wizard. File project <project\_name>.xise akan dibuat;



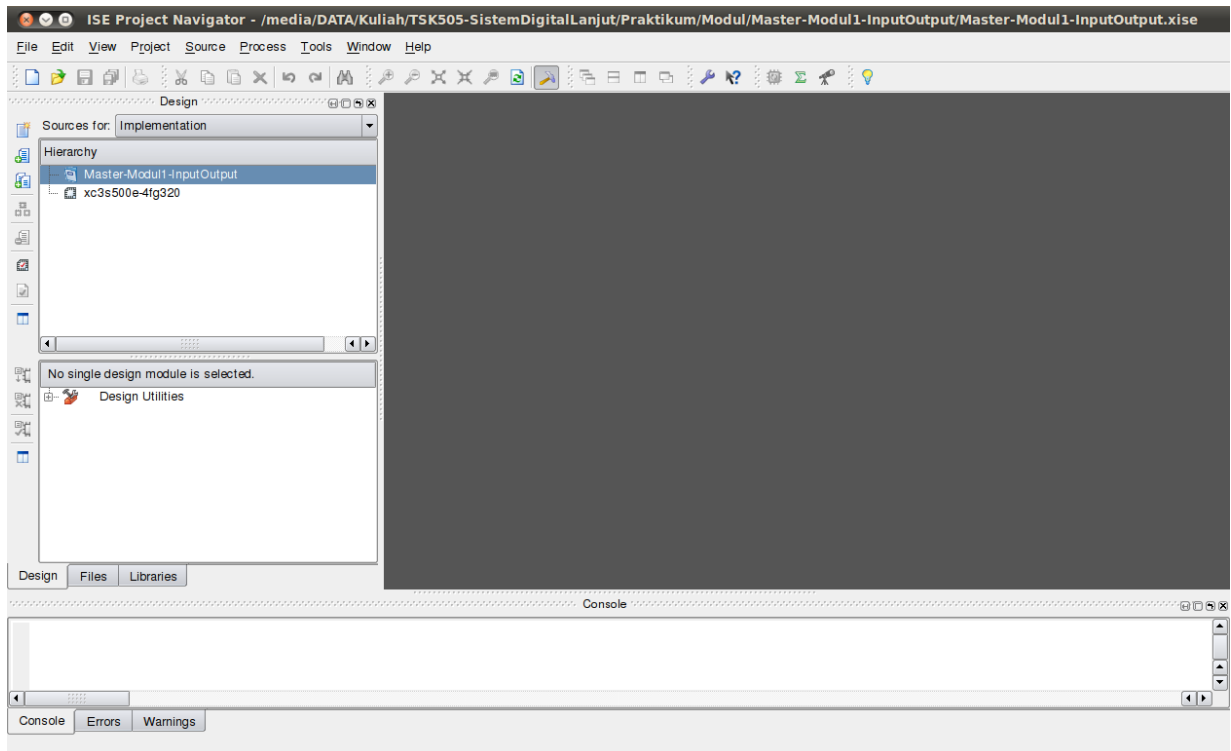


Gambar 10: Dialog New Project Wizard



Gambar 11: Jendela Device Properties

5. ISE Project Navigator menampilkan lingkungan *workspace* untuk mengembangkan sistem (Gambar 12).

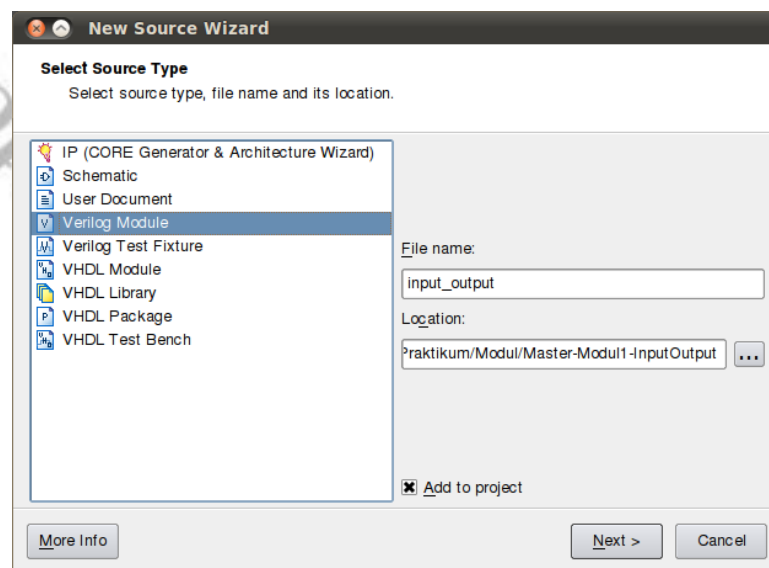


Gambar 12: Workspace proyek telah siap untuk digunakan

## 4.2.2 Membuat File Desain

File desain akan ditambahkan di proyek, yaitu 1) file HDL **input\_output.v** dan 2) file UCF **input-output.ucf**.

1. Dari ISE, klik menu **Project**→**New Source** (Alt+r N) untuk membuat file source baru<sup>1</sup>. Jendela **New Source Wizard** akan muncul (Gambar 13). Klik Verilog Module sebagai **Source Type**. Isikan field **File name** dengan input\_output<sup>2</sup>, **Location** sama dengan lokasi proyek. Pastikan checkbox **Add to project** tercentang. Klik tombol **Next**;



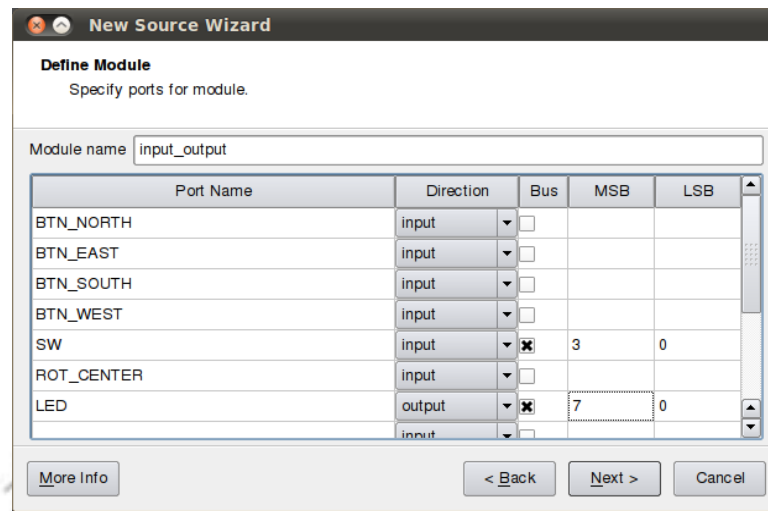
Gambar 13: Jendela New Source Wizard

2. Spesifikasikan port untuk modul input-output sesuai dengan diagram blok di Gambar 8. Nama port, arah (input/output) beserta ukuran datanya terlihat dalam Gambar 14.

<sup>1</sup> Cara lain adalah dengan klik ikon sidebar New Source

<sup>2</sup> Hanya sebuah nama file. Sesuai dengan nama file HDL yang ingin dibuat dan akan menjadi nama module

Misalnya port SW merupakan bus input dengan 4 buah jalur (SW<3> s/d SW<0>) dan LED merupakan bus output dengan 8 buah jalur (LED<7> s/d LED<0>). Klik **Next**.



Gambar 14: Spesifikasi port modul input-output

3. Dialog **Summary** akan muncul;

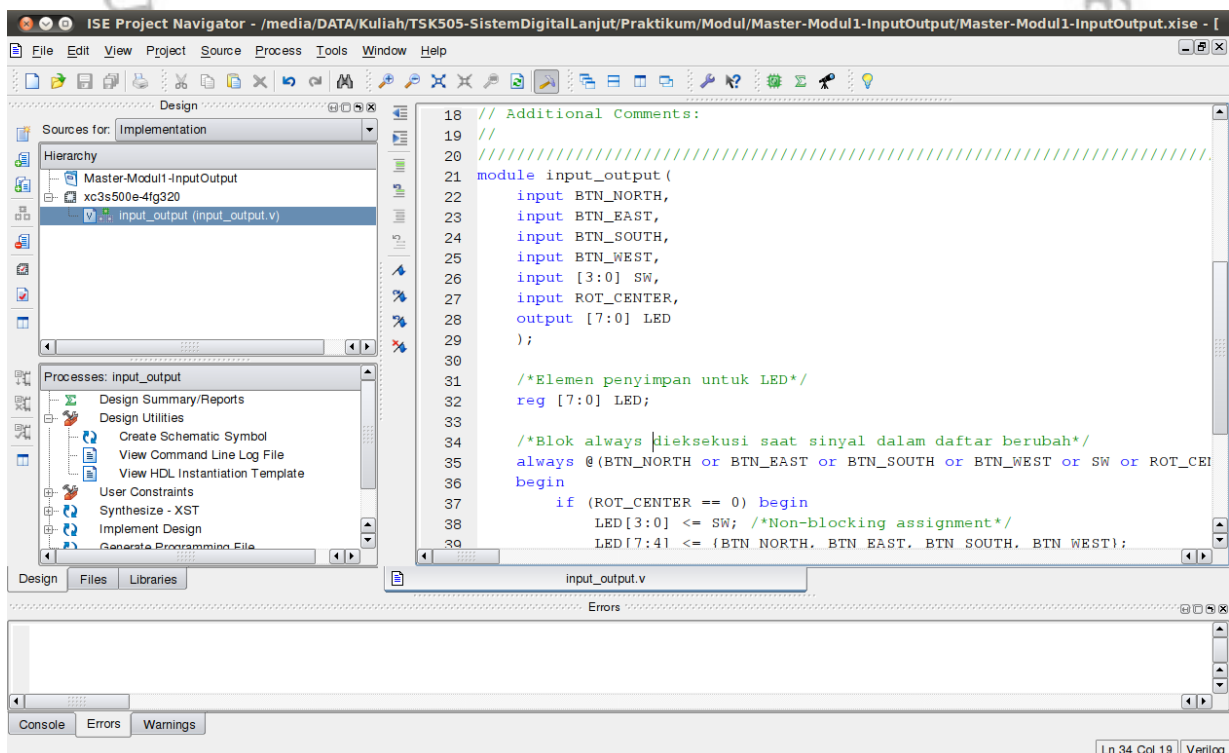


Catat ringkasan tersebut dalam lembar kegiatan

4. Klik tombol **Finish** untuk mengakhiri dialog. Sebuah file `input_output.v` yang mendefinisikan satu module bernama `input_output` akan muncul di workspace editor (Gambar 15). Modul tersebut telah mendefinisikan port-port yang tersedia dalam modul. Isikan deskripsi modul dan keterangan lainnya dalam komentar header<sup>3</sup>;



Bagaimana jika ingin menambah port bertipe bus input-output 8-jalur bernama



Gambar 15: Wizard membuat file source `input-output.v` yang berisi modul dan definisi portnya

<sup>3</sup> Komentar ditandai dengan karakter `//` atau `/*...*/`

DATA\_IO dalam daftar port modul. Tulis jawaban dalam lembar kegiatan

5. Pastikan modul `input_output` sebagai *top module*. Edit file `input_output.v` untuk menambah perilaku sistem yang diinginkan (Listing 5);


```
module input_output(
    input BTN_NORTH,
    input BTN_EAST,
    input BTN_SOUTH,
    input BTN_WEST,
    input [3:0] SW,
    input ROT_CENTER,
    output [7:0] LED
);


/*Elemen penyimpan untuk LED*/
reg [7:0] LED_REG;

/*Blok always dieksekusi saat sinyal dalam daftar berubah*/
always @(BTN_NORTH or BTN_EAST or BTN_SOUTH or BTN_WEST or SW
        or ROT_CENTER)
begin
    if (ROT_CENTER == 0) begin
        LED_REG[3:0] <= SW; /*Non-blocking assignment*/
        LED_REG[7:4] <= {BTN_NORTH, BTN_EAST, BTN_SOUTH, BTN_WEST};
    end else begin /*Invers*/
        LED_REG[3:0] <= ~SW; /*Non-blocking assignment*/
        LED_REG[7:4] <= {!BTN_NORTH, !BTN_EAST, !BTN_SOUTH, !BTN_WEST};
    end
end

assign LED = LED_REG; /*Continous assignment*/
endmodule
```

Listing 5: Definisi port dan perilaku modul `input_output`

 Jelaskan perilaku sistem masukan-keluaran dalam listing kode Verilog tersebut di atas dan tuliskan di lembar kegiatan serta lengkapi tabel kebenarannya

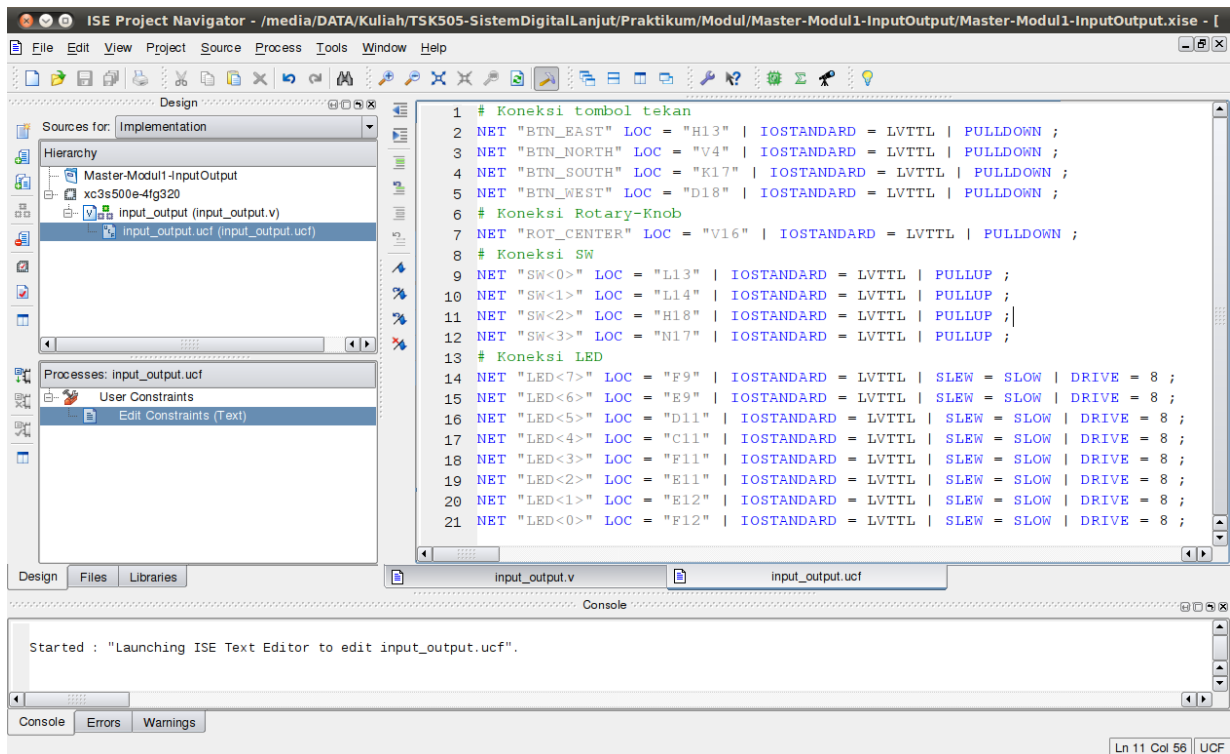
 Jelaskan perbedaan blocking dan non-blocking assignment

6. Buat file UCF. Klik menu **Project→New Source** (Alt+r N). Jendela **New Source Wizard** akan muncul. Klik “Implementation Constraints File” sebagai **Source Type**. Isikan field **File name** dengan `input_output4`, **Location** sama dengan lokasi proyek. Pastikan checkbox **Add to project** tercentang. Klik tombol **Next**. Jendela **Summary** muncul. Klik tombol **Finish** untuk mengakhiri pembuatan source baru;
7. File `input_output.ucf` akan dibuat. Klik file `input_output.ucf` dari Project Explorer, dan klik Edit Constraints (Text) dari Processes. Tambahkan isi file tersebut dari gabungan Listing 1, Listing 2, Listing 3, dan Listing 4 untuk mendefinisikan koneksi pin I/O, slew-rate dan arus keluaran dari tombol tekan, rotary-knob, saklar geser dan LED (Gambar 16). Desain siap untuk dikompile (sintesis dan implementasi);

---

4 Bisa sebarang nama

**Note:** setiap kali mengubah file input-output.ucf, project sebaiknya di-*cleanup*-kan dengan menjalankan “Cleanup Project Files” dari menu Project ISE untuk menghapus file-file yang dibangkitkan selama proses sintesis dan implementasi

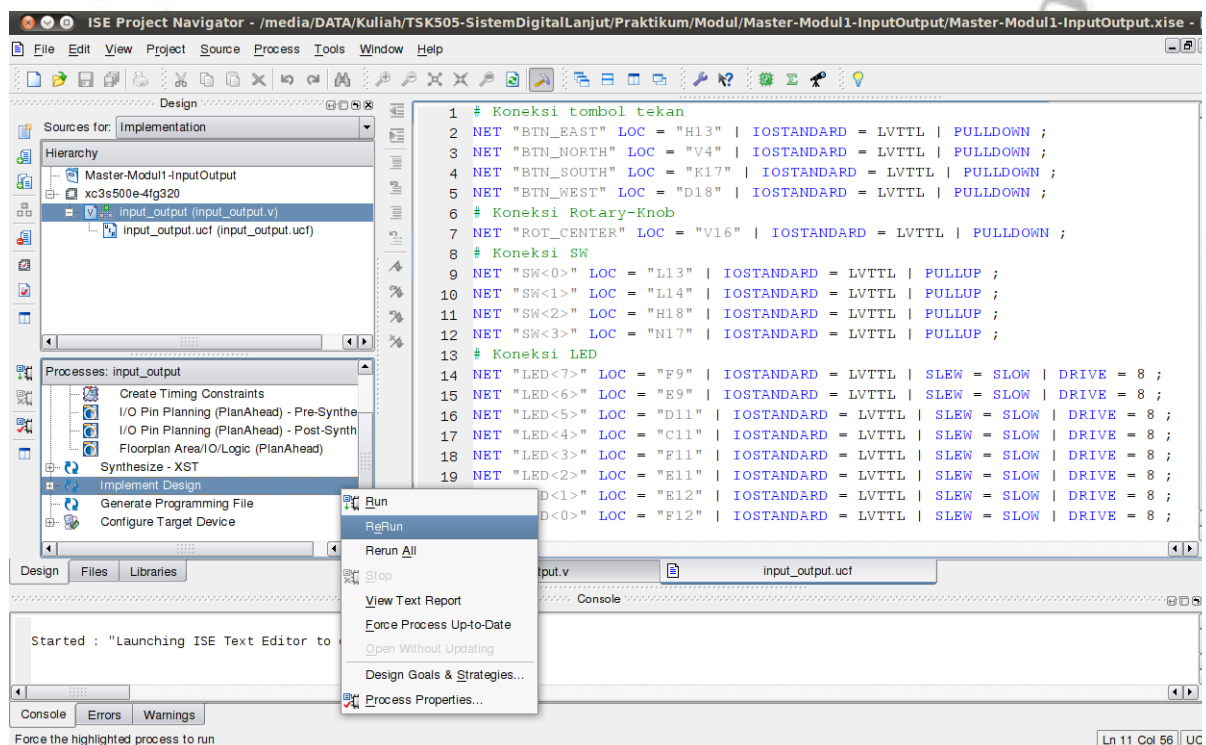


Gambar 16: Mengedit file constraints untuk implementasi desain

### 4.2.3 Mengkompilasi Desain dan Membangkitkan File Programming

Desain siap untuk disintesis dan dibangkitkan file programmingnya.

1. Klik *top module* (modul input\_output). Klik kanan proses *Implement Design* dan klik *Rerun* dari pop-up menu (Gambar 17). Sintesis dan implementasi akan dijalankan ISE;



Gambar 17: Implementasi desain



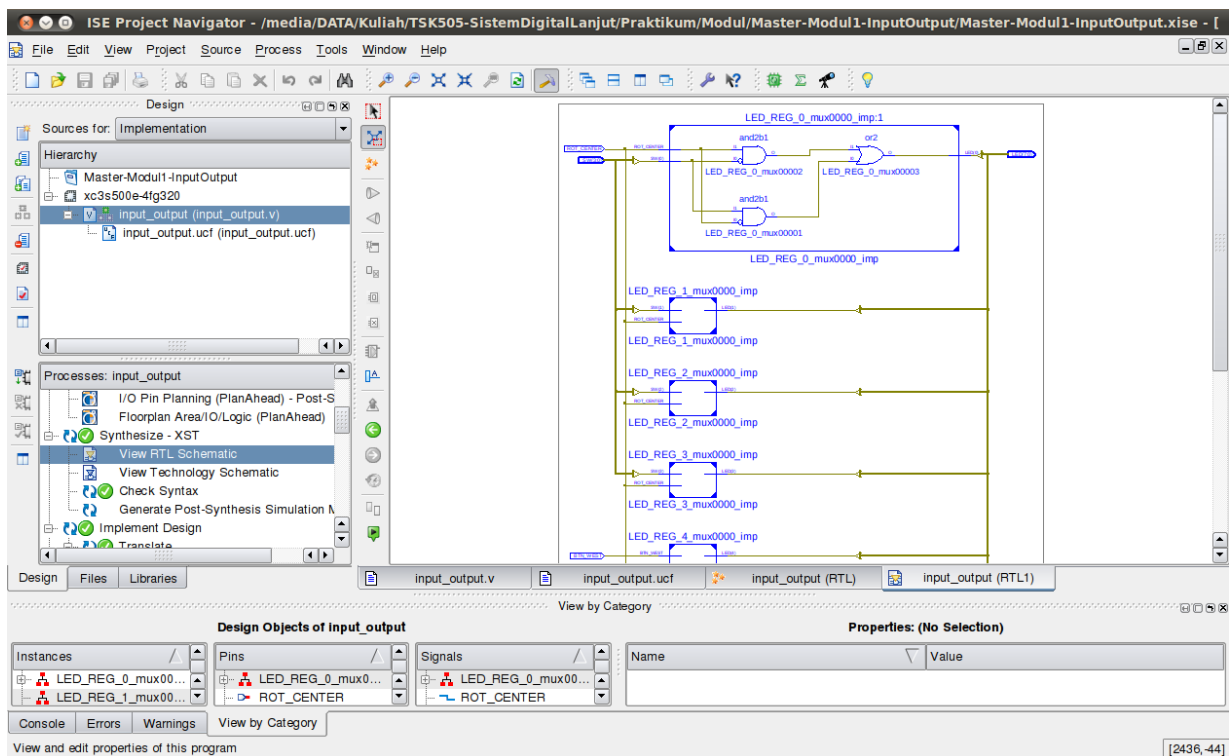
2. Bangkitkan file konfigurasi input\_output.bit. Klik ganda proses *Generate Programming File* atau klik kanan proses dan pilih *Run*;



Lihatlah skematik RTL yang dibangkitkan (Ekspand proses Synthesize, klik View RTL Schematic, dan pilih serta tambahkan semua LED register dari elemen yang tersedia. Klik tombol Create Schematic). Ekspand salah satu blok register LED. Gambarkan rangkaian logika untuk satu register LED tersebut di lembar kegiatan (Gambar 18)



Klik menu **Project**→**Design Summary/Reports**. Catat utilisasi device di lembar kegiatan



Gambar 18: Skematik RTL dari desain sistem

#### 4.2.4 Menyalakan Board

Sebelum diprogram, board Starter Kit harus diaktifkan.

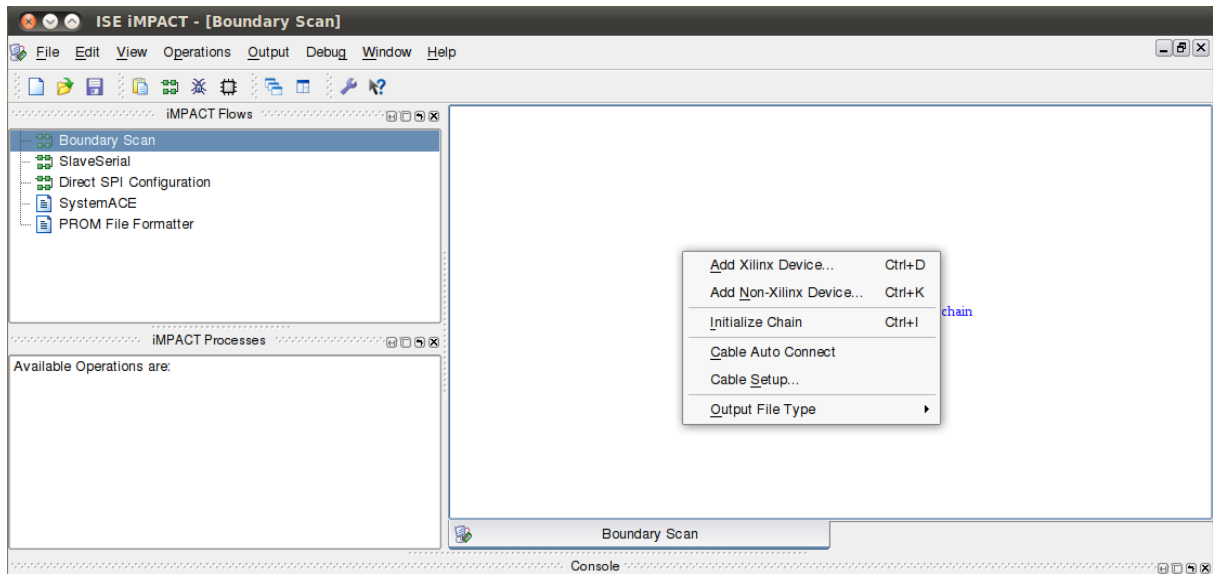
1. Pastikan saklar Sumber Daya (SWP) dalam posisi OFF. Colokkan power supply 5V ke konektor Power (J22) di board;
2. Sambungkan kabel USB dari komputer ke konektor USB-B di board;
3. Geser saklar Sumber Daya ke posisi ON. LED indikator akan menyala yang menandakan board telah aktif;

#### 4.2.5 Mengkonfigurasi FPGA dengan iMPACT

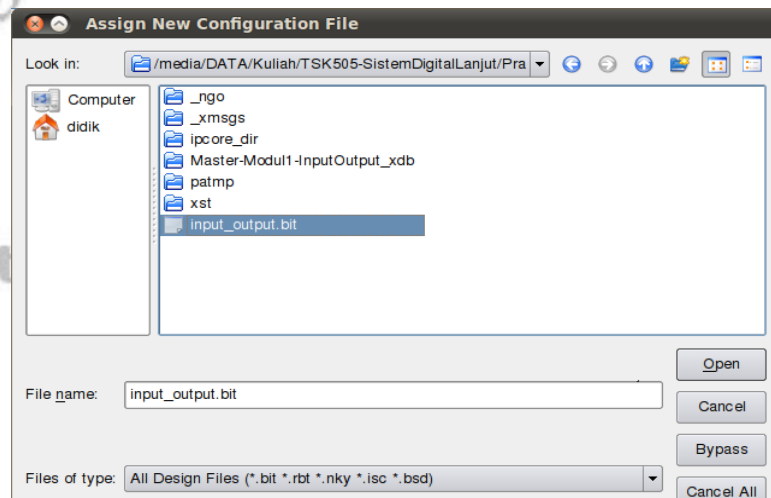
FPGA dikonfigurasi (diprogram) menggunakan program iMPACT.

1. Dari ISE, klik proses “Manage Configuration Project (iMPACT)” untuk menjalankan program iMPACT;
2. Klik ganda “Boundary Scan”. Kemudian klik kanan dan pilih Initialize Chain (Ctrl+I) untuk menginisialisasi Boundary Scan (Gambar 19);
3. Jendela untuk memasukkan file konfigurasi FPGA muncul. Pilih/buka input\_output.bit sebagai file konfigurasi FPGA (Gambar 20). Device lain bypass;

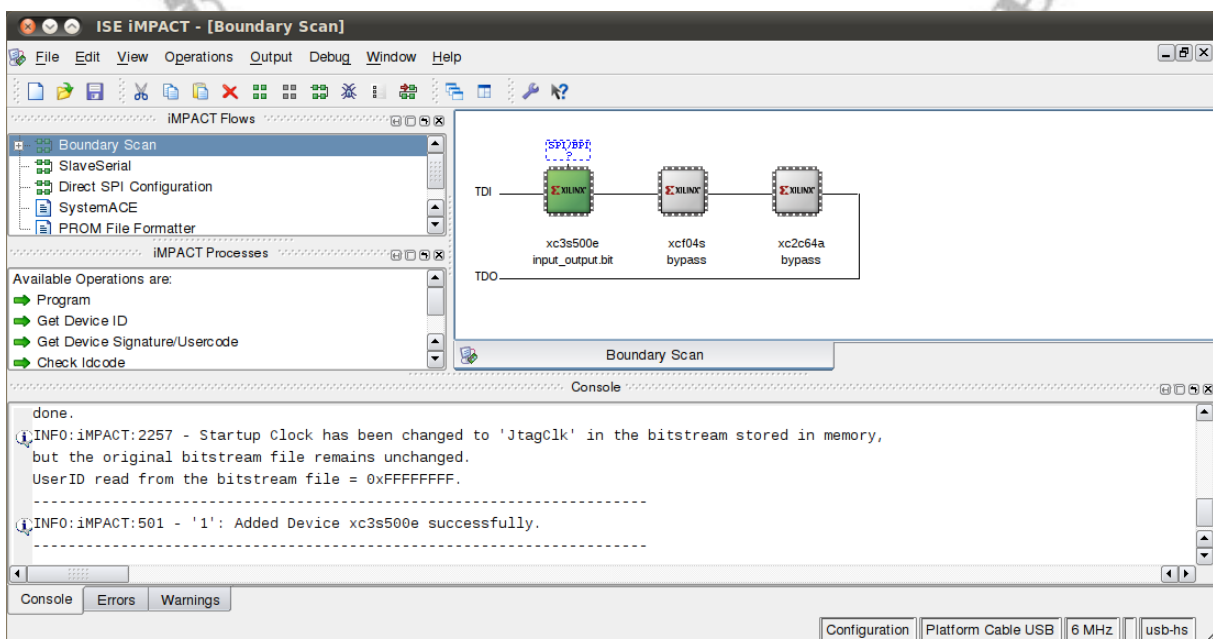
4. Chain JTAG sudah terkonfigurasi, dengan FPGA menggunakan input\_output.bit, flash XCF04S bypass, dan CPLD XC2C64A bypass (Gambar 21);



Gambar 19: ISE iMPACT untuk memprogram device lewat JTAG



Gambar 20: Memasukkan file konfigurasi FPGA



Gambar 21: Chain JTAG untuk memprogram device

5. Klik kanan device xc3s500e dan pilih “*Program*” untuk menuliskan konfigurasi ke device. LED indikator programming akan menyala.



Tuliskan device code dari FPGA yang digunakan (Hint: klik kanan FPGA dan pilih “Get Device ID”)



Amati perilaku sistem dan isi tabel di lembar kegiatan



Buat satu source file baru (modul) `input_output_xor` dengan port sama dengan modul `input_output`. Namun, modul `input_output_xor` menggunakan XOR daripada if-then-else dan menggunakan tipe kode struktural daripada prosedural seperti Listing 6. Jadikan sebagai top module, implemen modul dan bangkitkan file programming serta program ke board Starter Kit. Amati perilaku sistem dan isi tabel di lembar kegiatan. Gambarkan pula skematik RTL. Tuliskan pendapat Anda di lembar kegiatan.

```
module input_output_xor(  
    input BTN_NORTH,  
    input BTN_EAST,  
    input BTN_SOUTH,  
    input BTN_WEST,  
    input [3:0] SW,  
    input ROT_CENTER,  
    output [7:0] LED  
);  
  
    /*structural code*/  
    assign LED[7] = BTN_NORTH ^ ROT_CENTER;  
    assign LED[6] = BTN_EAST ^ ROT_CENTER;  
    assign LED[5] = BTN_SOUTH ^ ROT_CENTER;  
    assign LED[4] = BTN_WEST ^ ROT_CENTER;  
    assign LED[3] = SW[3] ^ ROT_CENTER;  
    assign LED[2] = SW[2] ^ ROT_CENTER;  
    assign LED[1] = SW[1] ^ ROT_CENTER;  
    assign LED[0] = SW[0] ^ ROT_CENTER;  
  
endmodule
```

Listing 6: Kode struktural dengan menggunakan XOR daripada if-then-else