

MODUL PRAKTIKUM
SISTEM DIGITAL DAN SISTEM MIKROPROSESOR



TIM DOSEN TEKNIK TELEKOMUNIKASI

**DEPARTEMEN PENDIDIKAN TEKNIK ELEKTRO
FAKULTAS PENDIDIKAN TEKNOLOGI DAN KEJURUAN
UNIVERSITAS PENDIDIKAN INDONESIA**

2019

DAFTAR ISI

JOBSHEET 1	3
Menginstal Program Arduino UNO	3
JOBSHEET 2	12
Program LED Berkedip.....	12
JOBSHEET 3	17
Program LED Berderet.....	17
JOBSHEET 4	21
Program Traffict Light	21
JOBSHEET 5	24
Output Seven Segment Pada Arduino	24
JOBSHEET 6	31
MOTOR STEPPER.....	31
JOBSHEET 7	37
KONTROL MOTOR SERVO MENGGUNAKAN ARDUINO	37
JOBSHEET 8	42
ARDUINO SEBAGAI PENGONTROL LED DAN BUZZER.....	42
JOBSHEET 10	59
DISAIN SKEMATIK MENGGUNAKAN SOFTWARE XILINX ISE	59
JOBSHEET 11	77
PEMROGRAMAN VHDL MENGGUNAKAN ISE DESIGN.....	77

JOBSHEET 1

Menginstal Program Arduino UNO

A. Tujuan Setelah mengikuti menyelesaikan materi

Setelah membaca modul diharapkan siswa dapat :

1. Peserta dapat menginstal program Arduino pada komputer/laptop masing- masing dengan benar

B. Indikator Pencapaian Kompetensi :

1. Mengenal mikrokontroler Arduino UNO
2. Menginstal program mikrokontroller Arduino
3. Menjalankan program mikrokontroler Arduino

C. Uraian Materi

Jobsheet ini dimaksudkan agar Anda yang masih pemula dalam dunia mikrokontroller dapat mengikuti dan mempelajari Arduino dengan mudah dan segera dapat mempraktekkannya. Oleh sebab itu, di sini akan dibahas tentang konsep elektronik, sensor, dan bahasa pemrograman secukupnya dengan harapan Anda bisa segera praktek tanpa memikirkan konsep elektronika yang relatif rumit.

Apa itu mikrokontroller?

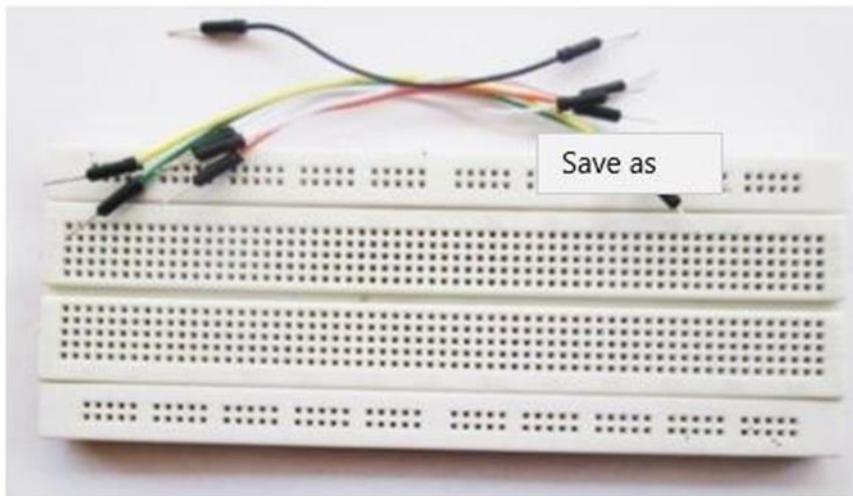
Dalam diskusi sehari-hari dan di forum internet, mikrokontroller sering dikenal dengan sebut μ C, uC, atau MCU. Terjemahan bebas dari pengertian tersebut, bisa dikatakan bahwa mikrokontroller adalah komputer yang berukuran mikro dalam satu chip IC (integrated circuit) yang terdiri dari processor, memory, dan antarmuka yang bisa diprogram. Jadi disebut komputer mikro karena dalam IC atau chip mikrokontroller terdiri dari CPU, memory, dan I/O yang bisa kita kontrol dengan memprogramnya. I/O juga sering disebut dengan GPIO (General Purpose Input Output Pins) yang berarti : pin yang bisa kita program sebagai input atau output sesuai kebutuhan.



Gambar 1.1 Arduino UNO

Dalam bahasan ini kita akan menggunakan board Arduino Uno (Gambar 1.1). Board Arduino terdiri dari hardware/modul mikrokontroller yang siap pakai dan software IDE yang digunakan untuk memprogram sehingga kita bisa belajar dengan mudah. Kelebihan dari Arduino yaitu kita tidak direpotkan dengan rangkaian minimum sistem dan programmer karena sudah built in dalam satu board. Oleh sebab itu kita bisa fokus ke pengembangan sistem.

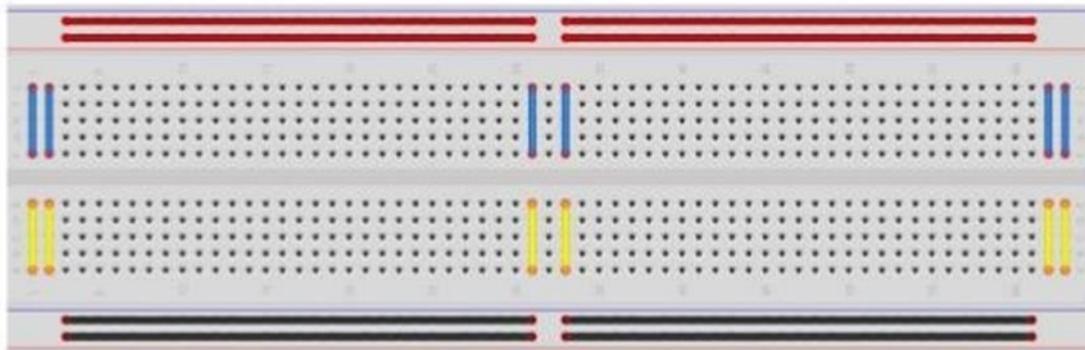
Untuk praktek, kita akan menggunakan project board (ada yang menyebutnya dengan istilah bread board) dan beberapa kabel jumper untuk menghubungkan antara komponen dan Arduino (Gambar 1.2). Dengan project board kita tidak perlu menyolder rangkaian sehingga relatif mudah dan cepat dalam merangkai. Project board memungkinkan kita untuk membangun dan membongkar rangkaian dengan cepat sehingga sangat cocok untuk eksperimen. Tapi jika kita ingin membuat rangkaian yang permanen, maka kita harus menggunakan PCB.



Gambar 1.2 Project Board dan Kabel Jumper

Yang terpenting adalah, kita harus memahami jalur-jalur pada project board. Project board yang akan diulas di sini terdiri dari jalur vertikal dan jalur horisontal. Jalur vertikal ada di bagian tengah yang terdiri dari 2 x 64 jalur. Masing-masing jalur terdiri dari 5 titik vertikal, misal jalur 1A1B-1C-1D-1E dan jalur 1F-1G-1H-1I-1J yang kedua tidak saling tersambung. Jalur horisontal

sebanyak 8 jalur, 4 jalur ada di bagian atas dan 4 jalur lagi di bagian bawah. Jalur ini bisa digunakan untuk power supply (VCC dan GND) untuk rangkaian. Untuk lebih jelasnya, silakan perhatikan Gambar 1.3. Garis-garis yang ada menunjukkan bahwa lubang tersebut terhubung secara fisik. Ada beberapa macam model project board, ada yang besar/panjang, ada yang pendek dan ada pula yang kecil. Semua model sama dalam penggunaannya dan cara pemasangan kabel jumper, prinsipnya seperti gambar 1.3 di bawah.



Gambar 1.3 Peta Jalur pada Project Board

Instalasi Arduino IDE Anda bisa mendownload Arduino IDE di website Arduino, yaitu di alamat : <https://www.arduino.cc/en/Main/Software>. Pada saat tulisan ini dibuat (12/01/2017), Arduino IDE sudah versi 1.8.1. Software Arduino ada yang versi installer (hanya untuk Windows) dan versi terkompres dalam zip. Jika memilih versi tanpa install (format .zip), maka Anda hanya perlu mengekstraknya di folder mana sajadan Anda bisa langsung menjalankannya. Jika Anda pengguna Linux, maka sedikit tantangan untuk Anda karena proses instalasi tidak semudah instalasi di Windows dan Mac. Panduan untuk menginstall di Linux bisa Anda pelajari di bagian instalasi Linux. Sedangkan untuk pengguna Windows dan Mac, Anda bisa menginstall dengan mengikuti instruksi dalam modul ini.

Instalasi di Windows

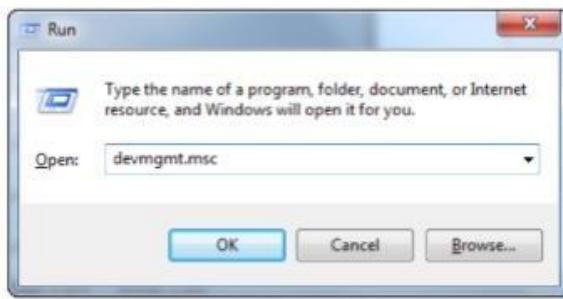
Praktik 1

1. Pasang board Arduino Anda ke port USB pada komputer atau laptop, kemudian tunggu hingga Windows mencoba untuk menginstall driver sendiri. Biasanya dia gagal menginstall driver jika belum memiliki driver tersebut. (Silakan lanjutkan ke step berikutnya)
2. Jika berhasil, berarti instalasi selesai. Tapi jika gagal, lanjutkan ke step selanjutnya.
3. Anda harus install dari device manager. Untuk masuk ke device manager, Anda bisa melakukannya dengan dua cara:



Gambar 1.4 Posisi tombol Windows

Tekan tombol ("Windows" + R) secara bersamaan. Tombol "Windows" adalah tombol pada keyboard dengan logo Windows (gambar logo windows, biasanya terletak di sebelah kiri atau kanan spasi, lihat Gambar 1.4). Setelah Anda menekan tombol "Windows" + R, maka akan muncul "Run", ketikkan "devmgmt.msc" (tanpa tanda petik), kemudian tekan tombol ENTER. Jika benar, maka akan muncul window Device Manager.



Gambar 1.5 Window yang muncul setelah menekan (Windows + R)

4. Jika Device Manager Anda sudah keluar, Anda bisa lanjut ke point 4, jika tidak, coba cara berikut untuk menampilkan device manager
5. Klik Start - pilih Control Panel. Di dalam Control Panel, pilih System and Security, lalu pilih System. Selanjutnya pilih Device Manager.



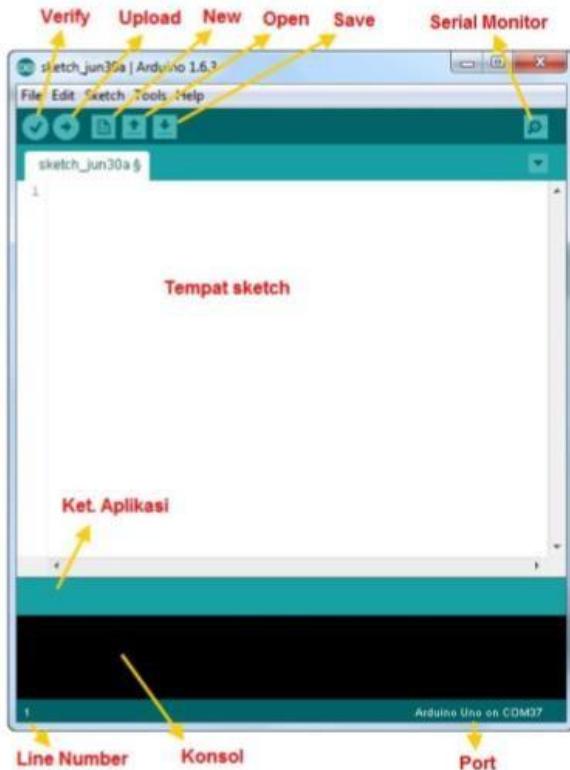
gambar 1.6 Tampilan Device Manager

6. Pada Device Manager, perhatikan bagian Ports (COM & LPT), akan muncul device baru dengan nama "Arduino UNO (COMxx)"
7. Klik kanan pada "Arduino UNO (COMxx)", kemudian pilih "Update Driver Software".
8. Selanjutnya pilih "Browse my computer for Driver software".

9. Cari folder software Arduino Anda, kemudian cari file arduino.inf (khusus untuk Arduino UNO REF.3) pada folder Drivers.
10. Jika Anda menggunakan versi IDE di bawah 1.0.3, Anda bisa memilih driver dengan nama file ArduinoUNO.inf
11. Jika berhasil, berarti instalasi driver sudah selesai. Jika belum, silakan Anda mencari caranya, bisa tanya ke teman-teman ataupun mencari jawabannya di internet.
12. Selanjut mari kita coba untuk mengupload sampel code yang ada pada software Arduino
13. Jalankan Aplikasi Arduino (arduino.exe), pada pojok kanan bawah akan ada tulisan "Arduino UNO on COMxx". Berarti port yang digunakan Arduino adalah COMxx, jika tulisan tersebut tidak muncul, berarti instalasi driver belum berhasil atau board Arduino belum disambungkan ke komputer. Selanjutnya, silakan buka sampel led flipflop dengan cara Klik menu File > Examples > 1.Basic > Blink
14. Setting board Arduino dengan cara : Klik menu Tools > Board > Arduino UNO
15. Pilih port yang digunakan Arduino dengan cara mengklik menu Tools > Ports > (pilih yang ada Arduino-nya)
16. Klik tombol upload (tombol denga panah ke kanan)
17. Setelah berhasil diupload, akan muncul tulisan "Done uploading" di bagian bawah.
18. Jika berhasil, maka LED dengan tulisan "L" pada board Arduino akan berkedip.

Arduino IDE

Untuk memprogram board Arduino, kita butuh aplikasi IDE (Integrated Development Environment) bawaan dari Arduino. Aplikasi ini berguna untuk membuat, membuka, dan mengedit source code Arduino (Sketches, para programmer menyebut source code arduino dengan istilah "sketches"). Selanjutnya, jika kita menyebut source code yang ditulis untuk Arduino, kita sebut "sketch" juga. Sketch merupakan source code yang berisi logika dan algoritma yang akan diupload ke dalam IC mikrokontroller (Arduino).



Gambar 1.7 Interface IDE Arduino

Interface Arduino IDE tampak seperti gambar 1.7. Dari kiri ke kanan dan atas ke bawah, bagian-bagian IDE Arduino terdiri dari:

1. Verify : pada versi sebelumnya dikenal dengan istilah Compile. Sebelum aplikasi diupload ke board Arduino, biasakan untuk memverifikasi terlebih dahulu sketch yang dibuat. Jika ada kesalahan pada sketch, nanti akan muncul error. Proses Verify/Compile mengubah sketch ke binary code untuk diupload ke mikrokontroller.

2. Upload : tombol ini berfungsi untuk mengupload sketch ke board Arduino. Walaupun kita tidak mengklik tombol verify, maka sketch akan di-compile, kemudian langsung diupload ke board. Berbeda dengan tombol verify yang hanya berfungsi untuk memverifikasi source code saja.
3. New Sketch : Membuka window dan membuat sketch baru
4. Open Sketch : Membuka sketch yang sudah pernah dibuat. Sketch yang dibuat dengan IDE Arduino akan disimpan dengan ekstensi file .ino
5. Save Sketch : menyimpan sketch, tapi tidak disertai mengcompile.
6. Serial Monitor : Membuka interface untuk komunikasi serial, nanti akan kita diskusikan lebih lanjut pada bagian selanjutnya
7. Keterangan Aplikasi : pesan-pesan yang dilakukan aplikasi akan muncul di sini, misal "Compiling" dan "Done Uploading" ketika kita mengcompile dan mengupload sketch ke board Arduino
8. Konsol : Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang sketch akan muncul pada bagian ini. Misal, ketika aplikasi mengcompile atau ketika ada kesalahan pada sketch yang kita buat, maka informasi error dan baris akan diinformasikan di bagian ini.
9. Baris Sketch : bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada sketch.
10. Informasi Port : bagian ini menginformasikan port yang dipakai oleh board Arduino.

Praktik 2

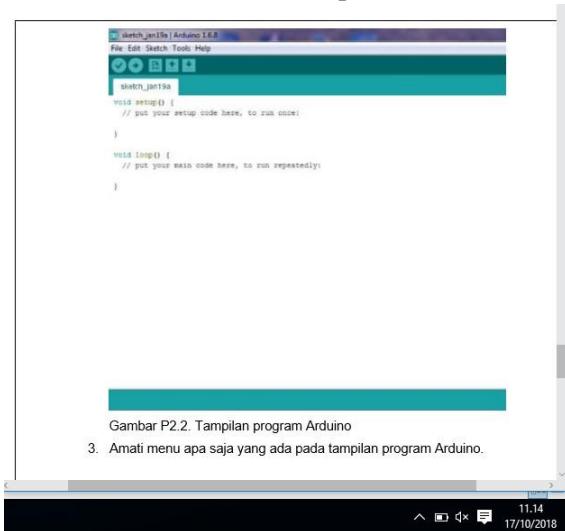
1. Buka program Arduino di komputer/laptop anda.

Ada 2 cara untuk membukanya, dapat lewat layar desktop ataupun lewat Start > Arduino. Pertama akan muncul gambar seperti di bawah. Ini menunjukkan program sedang dijalankan. Setelah selesai akan terbuka program Arduino seperti gambar P2.2



Gambar P2.1. Tampilan awal

Gambar P2.1. Tampilan awal



Gambar P2.2. Tampilan program Arduino
3. Amati menu apa saja yang ada pada tampilan program Arduino.

Gambar P2.2. Tampilan program Arduino 3. Amati menu apa saja yang ada pada tampilan program Arduino.

D. Aktifitas Pembelajaran

1. Lakukan pekerjaan install sesuai dengan langkah-langkah yang sudah tercantum di modul.
2. Setelah itu buka program dan pelajarilah aplikasi tersebut.

E. Latihan/Tugas Pertanyaan

Sebutkan tugas masing masing bagian sendiri

1. Verify !
2. Upload !
3. New Sketch !
4. Open Sketch !
5. Save Sketch !

6. Serial Monitor !
7. Keterangan Aplikasi !
8. Konsol !
9. Baris Sketch !
10. Informasi Port !

JOBSHEET 2

Program LED Berkedip

A. Tujuan Setelah mengikuti menyelesaikan materi

Setelah membaca Peserta dapat :

1. Menguji program Arduino untuk menyalakan lampu LED berkedip dengan benar.

B. Indikator Pencapaian Kompetensi

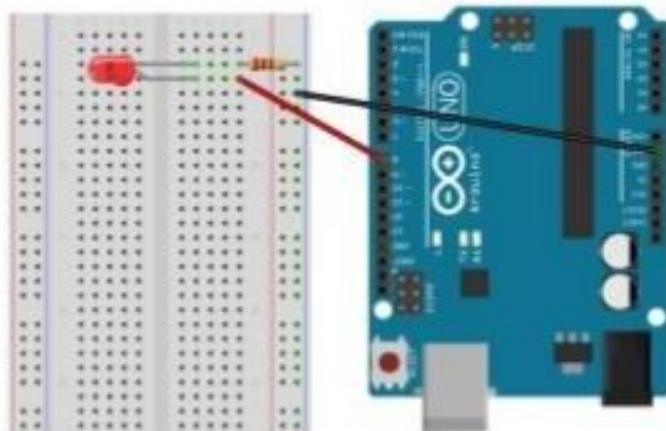
1. Membuat Sketch program lampu LED berkedip
2. Merangkai lampu LED berkedip dengan Arduino
3. Menjalankan Sketch program LED berkedip dengan Arduino

C. Uraian Materi

Rangkaian LED

Ketika belajar pemrograman, program pertama yang harus dicoba pertama kali adalah memunculkan pesan "Hello World!". Dalam belajar mikrokontroller ternyata juga ada, yang pertama kali harus dibuat adalah membuat lampu LED berkedip, LED berkedip maksudnya adalah flip-flop. LED merupakan kependekan dari Light Emitting Diode, yaitu diode yang mampu mengubah listrik menjadi cahaya. Sebagaimana sifat diode, lampu LED memiliki kaki positif dan negatif. Sehingga pemasangannya tidak boleh terbaik, jika dipasang terbalik maka tidak akan ada arus yang mengalir dan LED pun tidak akan menyala. Arduino bekerja pada tegangan 5-12 volt dengan arus yang relatif besar yang sanggup memutuskan LED. Sehingga jika kita ingin menyambungkan LED, maka kita butuh tahanan (resistor) untuk membatasi arus yang masuk ke LED. LED memiliki tegangan kerja yang disebut dengan forward voltage (f_v) yang mana tegangan ini adalah tegangan yang dibutuhkan LED untuk bisa menyala dengan baik dan aman. Ukuran resistor yang bisa dipakai adalah 100Ω hingga $1 K\Omega$ (Ω dibaca ohm, satuan dari resistansi/hambatan), makin besar nilai resistor maka nyala LED akan semakin redup. Pada Arduino, tegangan yang keluar dari pin-pinnya adalah 0-5 volt. Sementara catu daya untuk Arduino antara 5-12 volt. Oleh sebab itu, pemilihan resistor tergantung tegangan mana yang akan kita gunakan.

PERCOBAAN 1



Gambar 1. Rangkaian percobaan dengan menggunakan 1 LED dan 1 Resistor

1. Hubungkan kaki anoda (+) LED ke pin 8 di board Arduino dan kaki katoda (-) LED ke resistor 220 ohm lalu ke pin GND pada board Arduino. Hubungkan board Arduino ke komputer/laptop dengan kabel USB downloader.
2. Buka program Arduino, dan ketiklah sketch program berikut!

```
1 // Teknik Pemrograman
2 // Arduino untuk Pemula
3 // coder NBS
4
5 const int pinLED = 8;
6
7 void setup() {
8   pinMode(pinLED, OUTPUT);
9 }
10
11 void loop() {
12   digitalWrite(pinLED, HIGH);
13   delay(500);
14   digitalWrite(pinLED, LOW);
15   delay(500);
16 }
```

Gambar 2. Sketch Percobaan 1

Setelah selesai membuat Sketch maka akan tampak seperti gambar di bawah. Selanjutnya tekan tombol upload untuk mengirim Sketch program ke board Arduino untuk dijalankan. Tombol upload adalah menu panah arah ke kanan di bawahnya menu Edit

```
sketch_jan19a | Arduino 1.6.8
File Edit Sketch Tools Help
Upload
sketch_jan19a
// Teknik Pemrograman
// Arduino untuk Perempuan
// coder NBS

const int pinLED = 8;

void setup() {
pinMode(pinLED, OUTPUT);
}

void loop() {
digitalWrite(pinLED, HIGH);
delay(500);
digitalWrite(pinLED, LOW);
delay(500);
}
```

Gambar 3. Interface IDE Arduino dan Sketch Percobaan 1

Tunggu beberapa saat untuk proses mengirimkan sketch program ke board Arduino. Ditandai tulisan “Compiling sketch” pada pojok kiri bawah layar program Arduino. Setelah selesai tulisan menjadi “Done uploading”. Lihat apa yang terjadi pada rangkaian Arduino dan jelaskan apa yang Anda dapat dari pengamatan tersebut. Tuliskan hasil pengamatan anda.

PERCOBAAN 2

Memodifikasi Time Delay menggunakan command IF 1.

1. Buatlah rangkaian seperti gambar 1. Hubungkan kaki anoda (+) LED ke pin 8 di board arduino dan kaki katoda (-) LED ke resistor 220 ohm lalu ke pin GND pada board arduino. Hubungkan board Arduino ke komputer/laptop dengan kabel USB downloader.
2. Buka program Arduino, dan ketiklah sketch program berikut!

```
1 // Teknik Pemrograman
2 // Arduino untuk Pemula
3 // coder NBS
4
5 // Pin 8 untuk LED
6 const int pinLED = 8;
7
8 void setup() {
9 // pin LED sebagai output
10 pinMode(pinLED, OUTPUT);
11 }
12
13 // awal time delay 1000 | 1 detik

14 int timeDelay = 1000;
15
16 void loop() {
17 // Setiap looping, nilai timeDelay dikurangi 100
18 timeDelay = timeDelay - 100;
19
20 /* Jika timeDelay bernilai 0 atau negatif
21 maka nilai timeDelay direset ke 1000
22 */
23 if(timeDelay <= 0){
24 timeDelay = 1000;
25 }
26
27 //Nyalakan dan matikan LED selama timeDelay
28 digitalWrite(pinLED, HIGH);
29 delay(timeDelay);
30 digitalWrite(pinLED, LOW);
31 delay(timeDelay);
32 }
```

Gambar 4. Sketch Percobaan 2

Setelah selesai membuat Sketch maka selanjutnya tekan tombol upload untuk mengirim Sketch program ke board Arduino untuk dijalankan. Tombol upload adalah menu panah arah ke kanan di bawahnya menu Edit. Kalau tidak ada kesalahan pasti Sketch bisa dijalankan di Arduino. Jika ada kesalahan (error), maka carilah apa penyebabnya dan temukan pemecahannya. Tunggu beberapa saat untuk proses mengirimkan sketch program ke board Arduino. Ditandai tulisan “Compailing sketch” pada pojok kiri bawah layar program Arduino. Setelah selesai tulisan menjadi “Done uploading”. Lihat apa yang terjadi pada rangkaian Arduino dan jelaskan apa yang Anda dapat dari pengamatan tersebut. Tuliskan hasil pengamatan anda dan bandingkan dengan hasil pengamatan percobaan 1.

D. Aktifitas Pembelajaran

1. Selama proses pembelajaran, Peserta hendaknya mengidentifikasi dan mengamati cara menggunakan modul arduino Uno sesuai dengan manual book.

E. Tugas

1. Apa yang dimaksud lampu berkedip !
2. Apakah kepanjangan dari LED !
3. Apakah LED juga Dioda, dan bagaimana prinsip kerjanya !
4. Apakah pemasangan kaki dioda boleh terbalik!
5. Tuliskan bahasa pemrograman lampu berkedip

JOBSHEET 3

Program LED Berderet

A. Tujuan Setelah mengikuti menyelesaikan materi

Setelah membaca modul diharapkan siswa dapat :

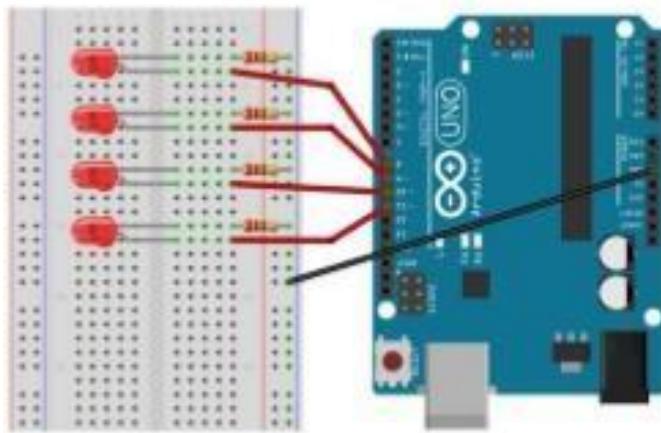
1. menguji program Arduino untuk menyalaikan lampu LED berderet dengan benar.

B. Indikator Pencapaian Kompetensi

1. Membuat Sketch program lampu LED berderet
2. Merangkai lampu LED berderet dengan Arduino
3. Menjalankan Sketch program LED berderet dengan Arduino

C. Uraian Materi

Jobsheet ini akan mempraktikkan pemrograman LED berderet sebagai pengembangan dari jobsheet sebelumnya



Gambar 1. Percobaan 1

PERCOBAAN 1

1. Siapkan 4 buah resistor dan 4 buah LED. Siapkan kabel jumper untuk menyuplai GDN pada project board.
2. Masing-masing kaki negatif LED dihubungkan ke GND dengan resistor. Sedangkan keempat LED tersebut dihubungkan berturut-turut dengan pin 8, 9, 10, dan 11 pada board Arduino.
3. Buka program Arduino, dan ketiklah sketch program berikut!

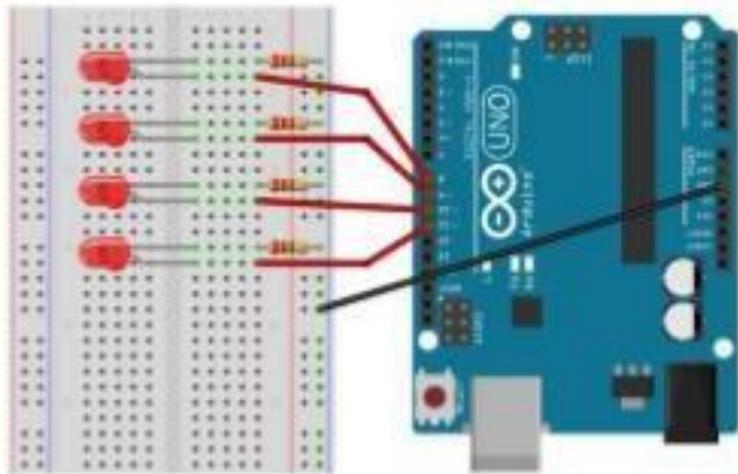
Buka program Arduino, dan ketiklah sketch program berikut!

```
1 // Teknik Pemrograman
2 // Arduino untuk Pemula
3 // coder NBS
4
5 // Inisialisasi Pin LED
6 const int pinLED1 = 8;
7 const int pinLED2 = 9;
8 const int pinLED3 = 10;
9 const int pinLED4 = 11;
10
11 void setup() {
12 // pin LED sebagai output
13 pinMode(pinLED1, OUTPUT);
14 pinMode(pinLED2, OUTPUT);
15 pinMode(pinLED3, OUTPUT);
16 pinMode(pinLED4, OUTPUT);
17 }
18
19 void loop() {
20 // perulangan sebanyak 5 kali
21 // dari i=0 hingga i=4 atau (i < 5)
22 for(int i=0; i<5; i++){
23 if(i==1){
24 // jika i=1, hidupkan led 1, led yang lain mati
25 digitalWrite(pinLED1, HIGH);
26 digitalWrite(pinLED2, LOW);
27 digitalWrite(pinLED3, LOW);
28 digitalWrite(pinLED4, LOW);
29 }else if(i==2){
30 // jika i=2, hidupkan led 1 & 2, led 3 & 4 mati
31 digitalWrite(pinLED1, HIGH);
32 digitalWrite(pinLED2, HIGH);
33 digitalWrite(pinLED3, LOW);
34 digitalWrite(pinLED4, LOW);
35 }else if(i==3){
36 // jika i=3, hidupkan led 1, 2, & 3, led 4 mati
37 digitalWrite(pinLED1, HIGH);
38 digitalWrite(pinLED2, HIGH);
39 digitalWrite(pinLED3, HIGH);
40 digitalWrite(pinLED4, LOW);
41 }else if(i==4){
42 // jika i=4, hidupkan semua led
43 digitalWrite(pinLED1, HIGH);
44 digitalWrite(pinLED2, HIGH);
45 digitalWrite(pinLED3, HIGH);
46 digitalWrite(pinLED4, HIGH);
47 }else{
48 // jika tidak, matikan semua led
49 digitalWrite(pinLED1, LOW);
50 digitalWrite(pinLED2, LOW);
51 digitalWrite(pinLED3, LOW);
52 digitalWrite(pinLED4, LOW);
53 }
54 // delay selama 5 detik
55 delay(1000);
56 }
```

4. Setelah selesai membuat Sketch, lanjutnya tekan tombol upload untuk mengirim Sketch program ke board Arduino untuk dijalankan. Tombol upload adalah menu panah arah ke kanan di bawahnya menu Edit.
5. Tunggu beberapa saat untuk proses mengirimkan sketch program ke board Arduino. Ditandai tulisan “Compiling sketch” pada pojok kiri bawah layar program Arduino. Setelah selesai tulisan menjadi “Done uploading”.
6. Lihat apa yang terjadi pada rangkaian Arduino dan jelaskan apa yang Anda dapat dari pengamatan tersebut. Tuliskan hasil pengamatan anda.

PERCOBAAN 2

Memodifikasi Time Delay menggunakan IF 1. Buatlah rangkaian seperti gambar di bawah!



Gambar 2. Percobaan 2

1. Siapkan 4 buah resistor dan 4 buah LED. Siapkan kabel jumper untuk menyulplai GDN pada project board.
2. Masing-masing kaki negatif LED dihubungkan ke GND dengan resistor. Sedangkan keempat LED tersebut dihubungkan berturut-turut dengan pin 8, 9, 10, dan 11 pada board Arduino.
3. Buka program Arduino, dan ketiklah sketch program berikut!

```
1 // Teknik Pemrograman
2 // Arduino untuk Pemula
3 // coder NBS
4
5 // Inisialisasi Jumlah LED
6 const int numLED = 4;
7 // LED 1,2,3,&4 jadi 1 varibel
8 // dengan alamat index 0,1,2,3
9 const int pinLED[numLED] = {8,9,10,11};

10
11 void setup() {
12 // Inisialisasi semua pin LED sebagai OUTPUT
13 for(int i=0; i<4; i++){
14 pinMode(pinLED[i], OUTPUT);
15 }
16 }
17
18 void loop() {
19 // hidupkan led indeks 0 hingga 2 satu-persatu
20 for(int i=0; i<3; i++){
21 digitalWrite(pinLED[i], HIGH);
22 delay(200);
23 digitalWrite(pinLED[i], LOW);
24 }
25 // hidupkan led indeks 3 hingga 1 satu-persatu
26 for(int i=3; i>0; i--){
27 digitalWrite(pinLED[i], HIGH);
28 delay(200);
29 digitalWrite(pinLED[i], LOW);
30 }
31 }
```

4. Setelah selesai membuat Sketch maka selanjutnya tekan tombol upload untuk mengirim

Sketch program ke board Arduino untuk dijalankan. Tombol upload adalah menu panah arah ke kanan di bawahnya menu Edit. Kalau tidak ada kesalahan pasti Sketch bisa dijalankan di Arduino. Jika ada kesalahan (error), maka carilah apa penyebabnya dan temukan pemecahannya.

5. Tunggu beberapa saat untuk proses mengirimkan sketch program ke board Arduino. Ditandai tulisan “Compiling sketch” pada pojok kiri bawah layar program Arduino. Setelah selesai tulisan menjadi “Done uploading”.
6. Lihat apa yang terjadi pada rangkaian Arduino dan jelaskan apa yang Anda dapat dari pengamatan tersebut. Tuliskan hasil pengamatan anda kemudian bandingkan dengan hasil pengamatan percobaan 1.

D. Tugas

Buatlah program yang lain dengan jumlah LED lebih banyak (Minimal 8) dengan nyala berganti-gantian selang 1 LED.

JOBSCHEET 4

Program Traffict Light

A. Tujuan Setelah mengikuti menyelesaikan materi

Dapat membuat program Traffict Light menggunakan Arduino dengan benar sesuai rancangannya

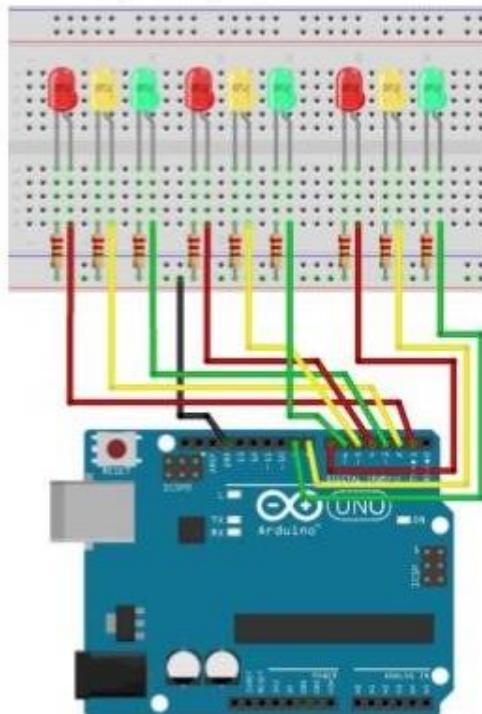
B. Indikator Pencapaian Kompetensi

Siswa dapat :

1. Membuat Sketch program Traffict Light menggunakan Arduino
2. Merangkai lampu LED sebagai Traffict Light menggunakan Arduino
3. Menjalankan Sketch program Traffict Light

C. Uraian Materi

Rangkaian LED Traffict Light Jobsheet ini akan mempraktikkan pemrograman LED berderet seolah sebagai Traffict Light, pengembangan dari jobsheet sebelumnya



Gambar 1. Percobaan 1

1. Siapkan 9 buah resistor 220 ohm, 3 buah LED merah, 3 buah LED kuning, dan 3 buah LED Hijau.
2. Siapkan kabel jumper untuk menyulplai GDN pada project board.

3. Buatlah rangkaian seperti gambar percobaan 1 di atas.
4. Buat sketch seperti gambar 2.

```

1 //Praktikum Sistem Digital dan Mikroprosesor
2 //Jobsheet 4
3 //Nama
4 //NIM
5 //Percobaan 1
6 //TRAFFIK LIGHT PERTIGAAN JALAN
7
8 //DEFINISIKAN
9
10 int UM=1;    //Utara Merah
11 int UK=2;    //Utara Kuning
12 int UH=3;    //Utara Hijau
13 int TM=4;    //Timur Merah
14 int TK=5;    //Timur Kuning
15 int TH=6;    //Timur Hijau
16 int SM=7;    //Selatan Merah
17 int SK=8;    //Selatan Kuning
18 int SH=9;    //Selatan Hijau
19
20 void setup(){
21   pinMode[UM,OUTPUT];
22   pinMode[UK,OUTPUT];
23   pinMode[UH,OUTPUT];
24   pinMode(TM,OUTPUT);
25   pinMode(TK,OUTPUT);
26   pinMode(TH,OUTPUT);
27   pinMode(SM,OUTPUT);
28   pinMode(SK,OUTPUT);
29   pinMode(SH,OUTPUT);
30 }
31
32 void loop(){
33
34 //Arah Utara Jalan
35 digitalWrite(UM,LOW);
36 digitalWrite(UK,LOW);
37 digitalWrite(UH,HIGH);
38 digitalWrite(TM,HIGH);
39 digitalWrite(TK,LOW);
40 digitalWrite(TH,LOW);
41 digitalWrite(SM,HIGH);
42 digitalWrite(SK,LOW);
43 digitalWrite(SH,LOW);
44 delay(6000);
45
46 digitalWrite(UM,LOW);
47 digitalWrite(UK,HIGH);
48 digitalWrite(UH,LOW);
49 digitalWrite(TM,HIGH);
50 digitalWrite(TK,LOW);
51 digitalWrite(TH,LOW);
52 digitalWrite(SM,HIGH);
53 digitalWrite(SK,LOW);
54 digitalWrite(SH,LOW);
55 delay(1000);
56
57 //Arah Timur Jalan
58 digitalWrite(UM,HIGH);
59 digitalWrite(UK,LOW);
60 digitalWrite(UH,LOW);
61 digitalWrite(TM,LOW);
62 digitalWrite(TK,LOW);
63 digitalWrite(TH,HIGH);
64 digitalWrite(SM,HIGH);
65 digitalWrite(SK,LOW);
66 digitalWrite(SH,LOW);
67 delay(6000);
}

```

Ln:33 Col:1 Sel:0|0 Windows (CR LF) UTF-8 INS

```
68 digitalWrite(UM,HIGH);
69 digitalWrite(UK,LOW);
70 digitalWrite(UH,LOW);
71 digitalWrite(TM,LOW);
72 digitalWrite(TK,HIGH);
73 digitalWrite(TH,LOW);
74 digitalWrite(SM,HIGH);
75 digitalWrite(SK,LOW);
76 digitalWrite(SH,LOW);
77 digitalWrite(SH,HIGH);
78 delay(1000);
79
80 //Arah Selatan Jalan
81 digitalWrite(UM,HIGH);
82 digitalWrite(UK,LOW);
83 digitalWrite(UH,LOW);
84 digitalWrite(TM,HIGH);
85 digitalWrite(TK,LOW);
86 digitalWrite(TH,LOW);
87 digitalWrite(SM,LOW);
88 digitalWrite(SK,LOW);
89 digitalWrite(SH,HIGH);
90 delay[6000];
91
92 digitalWrite(UM,HIGH);
93 digitalWrite(UK,LOW);
94 digitalWrite(UH,LOW);
95 digitalWrite(TM,HIGH);
96 digitalWrite(TK,LOW);
97 digitalWrite(TH,LOW);
98 digitalWrite(SM,LOW);
99 digitalWrite(SK,HIGH);
100 digitalWrite(SH,LOW);
101 delay(1000);
```

Ln:33 Col:1 Sel:0|0 Windows (CR LF) UTF-8 INS ..

Gambar 2. Sketch *traffic light* simpang 3

5. Setelah selesai membuat Sketch, lanjutnya tekan tombol upload untuk mengirim Sketch program ke board Arduino untuk dijalankan. Tombol upload adalah menu panah arah ke kanan di bawahnya menu Edit.
6. Tunggu beberapa saat untuk proses mengirimkan sketch program ke board Arduino. Ditandai tulisan “Compailing sketch” pada pojok kiri bawah layar program Arduino. Setelah selesai tulisan menjadi “Done uploading”.
7. Lihat apa yang terjadi pada rangkaian Arduino dan jelaskan apa yang Anda dapat dari pengamatan tersebut. Tuliskan hasil pengamatan anda.

D. Tugas

Buatlah program dengan traffic light simpang 4 dengan kreasi rancangan anda sendiri.

JOBSHEET 5

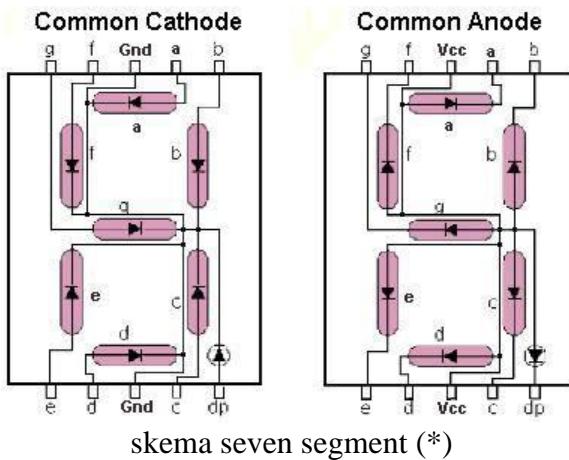
Output Seven Segment Pada Arduino

1. TUJUAN

- Mahasiswa dapat mengetahui bagaimana mengoutputkan karakter angka dan huruf pada seven segment menggunakan arduino.

2. DASAR TEORI

Pada umumnya seven segment terdiri 7 buah led yang disusun membentuk angka 8, dimana setiap segmentnya terdiri dari LED yang salah satu kaki terminal lednya di jadikan satu atau yang disebut dengan common. Kaki yang dijadikan satu / common sendiri dibagi menjadi 2 macam yaitu kaki common anoda dan kaki common cathoda. Skema dari 7 batang led tadi biasanya ditandai dengan huruf a - g, sebagai berikut:



Common Anoda

Common Anoda merupakan bagian kaki dari anoda (+) yang dijadikan satu dan dihubungkan dengan arus positif tegangan. sedangkan untuk mengaktifkan kaki yang lainnya harus di beri tegangan negatif. atau led akan menyala jika dalam kondisi aktif low (diberi logika 0). Misalkan ingin menampilkan angka 1, maka yang harus di lakukan adalah. kaki common di beri tegangan +, sedangkan kaki b dan c di beri tegangan -.

Common Katoda

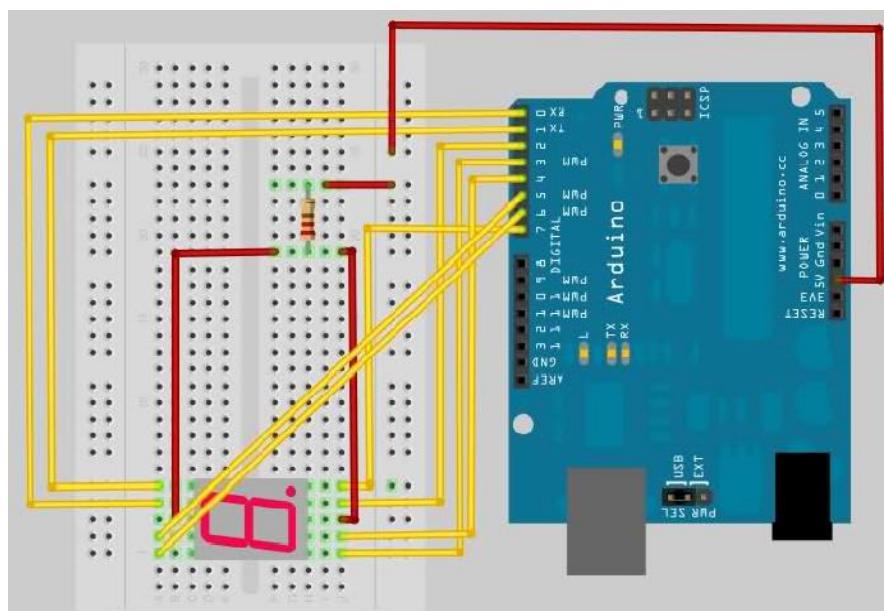
Common katoda ini kebalikannya dari common anoda, jadi kaki common yang disatukan adalah kaki katoda (-), sehingga untuk mengaktifkan kaki yang lain di beri tegangan (+) atau diberi logik high (1).

3. PERALATAN YANG DIBUTUHKAN

Projectboard	1 buah
Arduino uno	1 buah
Seven segment (boleh anoda maupun katoda)	1 buah
Resistor 1 KΩ	1 buah
Kabel jumper	secukupnya

Arduino Pin	7 Segment Pin Connection
2	7 (A)
3	6 (B)
4	4 (C)
5	2 (D)
6	1 (E)
7	9 (F)
8	10 (G)
9	5 (DP)

4. RANGKAIAN PERCOBAAN



5. LANGKAH PERCOBAAN

1. Susunlah komponen-komponen praktikum seperti gambar di atas, caranya :
2. Hubungkan kaki + Anoda Seven Segment ke Pin VCC Arduino menggunakan kabel jumper.
3. Hubungkan pin-pin seven segment ke pin Arduino menggunakan kabel jumper.
4. Hubungkan board Arduino Uno dengan Komputer menggunakan kabel USB.
5. Bukalah IDE Arduino, kemudian ketikkan kode program/sketch.
6. Compile menggunakan verify button (tanda ceklist pada IDE arduino) untuk mengecek ada atau tidaknya error/kesalahan dalam pengetikan.
7. Upload program ke arduino dengan cara, pilih File > Upload to I/O board, atau tekan tombol tanda panah pada jendela IDE arduino.

6. PROGRAM PERCOBAAN

A. Membuat Angka “1” Pada seven Segment

```
1 //Praktikum Sistem Digital dan Mikroprosesor
2 //Jobsheet 5
3 //Nama
4 //NIM
5 //Percobaan 1
6
7 int A=6;
8 int B=7;
9 int C=1;
10 int D=2;
11 int E=3;
12 int F=5;
13 int G=4;
14 void setup()
15 {
16 //Setup our pins
17 pinMode(A, OUTPUT);
18 pinMode(B, OUTPUT);
19 pinMode(C, OUTPUT);
20 pinMode(D, OUTPUT);
21 pinMode(E, OUTPUT);
22 pinMode(F, OUTPUT);
23 pinMode(G, OUTPUT);
24 }
25 void loop()
26 {
27 digitalWrite(A, HIGH);
28 digitalWrite(B, LOW);
29 digitalWrite(C, LOW);
30 digitalWrite(D, HIGH);
31 digitalWrite(E, HIGH);
32 digitalWrite(F, HIGH);
33 digitalWrite(G, HIGH);
34 }
```

B. Membuat Rotasi Huruf dan Angka pada Seven Segment

```
1 //Praktikum Sistem Digital dan Mikroprosesor
2 //Jobsheet 5
3 //Nama
4 //NIM
5 //Percobaan 2
6
7 int A=6;
8 int B=7;
9 int C=1;
10 int D=2;
11 int E=3;
12 int F=5;
13 int G=4;
14 void setup()
15 {
16 //Setup our pins
17 pinMode(A, OUTPUT);
18 pinMode(B, OUTPUT);
19 pinMode(C, OUTPUT);
20 pinMode(D, OUTPUT);
21 pinMode(E, OUTPUT);
22 pinMode(F, OUTPUT);
23 pinMode(G, OUTPUT);
24 }
25 void clr()
26 {
27 //Clears the LED
28 digitalWrite(A, HIGH);
29 digitalWrite(B, HIGH);
30 digitalWrite(C, HIGH);
31 digitalWrite(D, HIGH);
32 digitalWrite(E, HIGH);
33 digitalWrite(F, HIGH);
34 digitalWrite(G, HIGH);
35 }
36 void char_A()
37 {
38 digitalWrite(D, HIGH);
39 digitalWrite(E, LOW);
40 digitalWrite(F, LOW);
41 }
42 void char_B()
43 {
44 //Displays B
45 digitalWrite(D, LOW);
46 digitalWrite(E, LOW);
47 digitalWrite(F, LOW);
48 digitalWrite(G, LOW);
49 digitalWrite(A, HIGH);
50 digitalWrite(B, HIGH);
51 digitalWrite(C, LOW);
52 }
53 void char_C()
54 {
55 //Displays C
56 digitalWrite(D, LOW);
57 digitalWrite(E, LOW);
58 digitalWrite(F, LOW);
59 digitalWrite(G, HIGH);
60 digitalWrite(A, LOW);
61 digitalWrite(B, HIGH);
62 digitalWrite(C, HIGH);
63 }
64 void char_D()
65 {
66 //Displays D
67 digitalWrite(D, LOW);
```

```
66 //Displays D
67 digitalWrite(D, LOW);
68 digitalWrite(E, LOW);
69 digitalWrite(F, HIGH);
70 digitalWrite(G, LOW);
71 digitalWrite(A, HIGH);
72 digitalWrite(B, LOW);
73 digitalWrite(C, LOW);
74 }
75 void char_E()
76 {
77 //Displays E
78 digitalWrite(D, LOW);
79 digitalWrite(E, LOW);
80 digitalWrite(F, LOW);
81 digitalWrite(G, LOW);
82 digitalWrite(A, LOW);
83 digitalWrite(B, HIGH);
84 digitalWrite(C, HIGH);
85
86 void char_F()
87 {
88 //Displays F
89 digitalWrite(D, HIGH);
90 digitalWrite(E, LOW);
91 digitalWrite(F, LOW);
92 digitalWrite(G, LOW);
93 digitalWrite(A, LOW);
94 digitalWrite(B, HIGH);
95 digitalWrite(C, HIGH);
96 }
97 void one()
98 {
99 //Displays 1
100 digitalWrite(D, HIGH);
101
102 //Displays 1
103 digitalWrite(D, HIGH);
104 digitalWrite(E, LOW);
105 digitalWrite(F, LOW);
106 digitalWrite(G, HIGH);
107 }
108 void two()
109 {
110 //Displays 2
111 digitalWrite(D, LOW);
112 digitalWrite(E, LOW);
113 digitalWrite(F, HIGH);
114 digitalWrite(G, LOW);
115 digitalWrite(A, LOW);
116 digitalWrite(B, LOW);
117 digitalWrite(C, HIGH);
118 }
119 void three()
120 {
121 //Displays 3
122 digitalWrite(D, LOW);
123 digitalWrite(E, HIGH);
124 digitalWrite(F, HIGH);
125
126 void four()
127 {
128 //Displays 4
129 digitalWrite(D, HIGH);
130 digitalWrite(E, HIGH);
131 digitalWrite(F, LOW);
132 digitalWrite(G, LOW);
133 digitalWrite(B, HIGH);
```

```
131 digitalWrite(F, LOW);
132 digitalWrite(G, LOW);
133 digitalWrite(A, HIGH);
134 digitalWrite(B, LOW);
135 digitalWrite(C, LOW);
136 }
137 void five()
138 {
139 //Displays 5
140 digitalWrite(D, LOW);
141 digitalWrite(E, HIGH);
142 digitalWrite(F, LOW);
143 digitalWrite(G, LOW);
144 digitalWrite(A, LOW);
145 digitalWrite(B, HIGH);
146 digitalWrite(C, LOW);
147 }
148 void six()
149 {
150 //Displays 6
151 digitalWrite(D, LOW);
152 digitalWrite(E, LOW);
153 digitalWrite(F, LOW);
154 digitalWrite(G, LOW);
155 digitalWrite(A, LOW);
156 digitalWrite(B, HIGH);
157 digitalWrite(C, LOW);
158 }
159 void seven()
160 {
161 //Displays 7
162 digitalWrite(D, HIGH);
163 digitalWrite(E, HIGH);
164 digitalWrite(F, HIGH);
165 digitalWrite(G, HIGH);
166 digitalWrite(A, LOW);
167 digitalWrite(B, LOW);
168 digitalWrite(C, LOW);
169 }
170 void eight()
171 {
172 //Displays 8
173 digitalWrite(D, LOW);
174 digitalWrite(E, LOW);
175 digitalWrite(F, LOW);
176 digitalWrite(G, LOW);
177 digitalWrite(A, LOW);
178 digitalWrite(B, LOW);
179 digitalWrite(C, LOW);
180 }
181 void nine()
182 {
183 //Displays 9
184 digitalWrite(D, LOW);
185 digitalWrite(E, HIGH);
186 digitalWrite(F, LOW);
187 digitalWrite(G, LOW);
188 digitalWrite(A, LOW);
189 digitalWrite(B, LOW);
190 digitalWrite(C, LOW);
191 }
192 void zero()
193 {
194 //Displays 0
195 digitalWrite(D, LOW);
196 digitalWrite(E, LOW);
197 digitalWrite(F, LOW);
198 digitalWrite(G, HIGH);
199 digitalWrite(A, LOW);
```

```
199 digitalWrite(A, LOW);
200 digitalWrite(B, LOW);
201 digitalWrite(C, LOW);
202 }
203 void LoopDisplay()
204 {
205 //Loop through all Chars and Numbers
206 char_A();
207 delay(1000);
208 char_B();
209 delay(1000);
210 char_C();
211 delay(1000);
212 char_D();
213 delay(1000);
214 char_E();
215 delay(1000);
216 char_F();
217 delay(1000);
218 one();
219 delay(1000);
220 two();
221 delay(1000);
222 three();
223 delay(1000);
224 four();
225 delay(1000);
226 five();
227 delay(1000);
228 six();
229 delay(1000);
230 seven();
231 delay(1000);
232 eight();
233 delay(1000);
234 nine();
235 delay(1000);
236 zero();
237 delay(1000);
238 }
239 void loop()
240 {
241 LoopDisplay();
242 }
243
```

7. TUGAS

1. Buatlah program untuk menampilkan nim anda sendiri dengan delay 2ms antar angkanya.
2. Buatlah program untuk menampilkan nama anda sendiri dengan delay 1.5ms antar hurufnya.

JOBSHEET 6

MOTOR STEPPER

A. TUJUAN

1. Praktikan dapat mengakses motor stepper menggunakan Arduino
2. Praktikan dapat mengontrol motor stepper menggunakan Arduino

B. TEORI DASAR

Motor Stepper

Motor stepper merupakan perangkat elektromekanis yang bekerja dengan mengubah pulsa elektronis menjadi gerakan mekanis diskrit. Penggunaan motor stepper memiliki beberapa keunggulan dibandingkan dengan penggunaan motor DC biasa. Keunggulannya antara lain adalah :

- Sudut rotasi motor proporsional sehingga lebih mudah diatur
- Pada saat mulai bergerak, motor dapat langsung memberikan torsi penuh
- Presisis dalam posisi dan pergerakan repetisinya
- Memiliki respon yang sangat baik untuk pergerakan
- Karena tidak adanya sikat yang bersentuhan dengan rotor seperti pada motor DC, sehingga dapat dikatakan motor stepper itu bersifat realibel
- Dikarenakan beban dapat dikopel langsung ke porosnya sehingga dapat menghasilkan perputaran yang lambat
- Frekuensi perputaran dapat ditentukan secara bebas

1. Perbedaan antara motor DC dengan Motor stepper

Secara basic motor stepper berbeda dengan motor dc. Pada motor DC, prinsip kerjanya yaitu kumparan bergerak tergantung pada arah arusnya terhadap dua keping magnet permanen. Sedangkan, pada motor stepper terdapat komponen yang disebut dengan rotor dan stator.

Stator merupakan kumparan yang mempengaruhi pergerak motoran yang dimana jumlahnya lebih dari satu sesuai dengan fasanya. Sedangkan rotor merupakan magnet permanen yang akan bergerak terhadap kumparan / stator.

2. Prinsip kerja

Prinsip kerjanya yaitu mengubah pulsa-pulsa input menjadi gerakan mekanis diskrit. Motor ini bergerak berdasarkan urutan pulsa yang diberikan. Sehingga untuk menggerakkan motor ini diperlukan pengendali motor stepper yang berfungsi untuk membangkitkan pulsa-pulsa periodik.

Pulsa keluaran dari pengendali motor yang berupa gelombang kotak dan penerapan pulsa tersebut untuk menghasilkan arah putaran yang bersesuaian dengan pulsa kendali.

3. Jenis Motor stepper

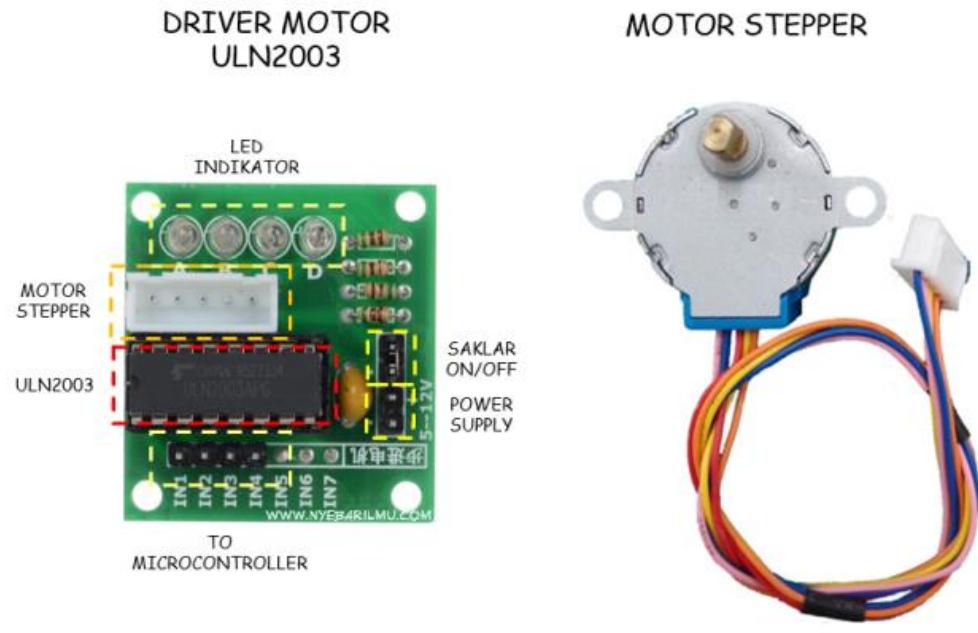
Berdasarkan metode perancangan rangkain pengendalinya, dapat dibagi dalam 2 jenis antara lain :

Motor Stepper Unipolar

Motor Stepper Bipolar

Pada pasaran telah banyak dijual modul motor stepper yang sudah diikutsertakan driver motornya menggunakan IC ULN2003. Dan modul tersebut akan dijadikan bahan yang

akan disiapkan untuk tutorial motor stepper menggunakan arduino uno. *Contoh Modul motor stepper dan driver motor ULN2003*



C. ALAT DAN BAHAN

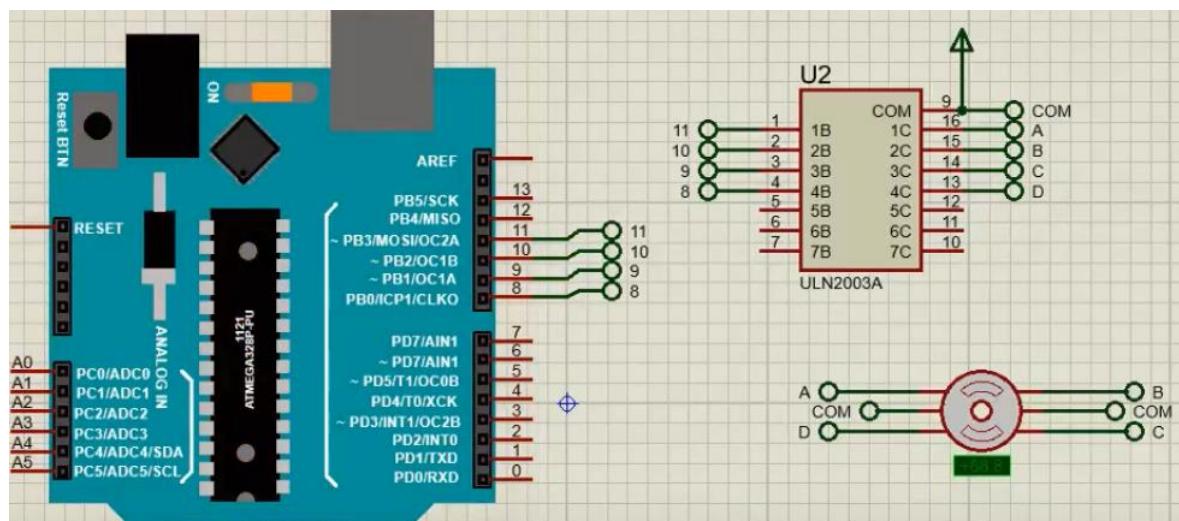
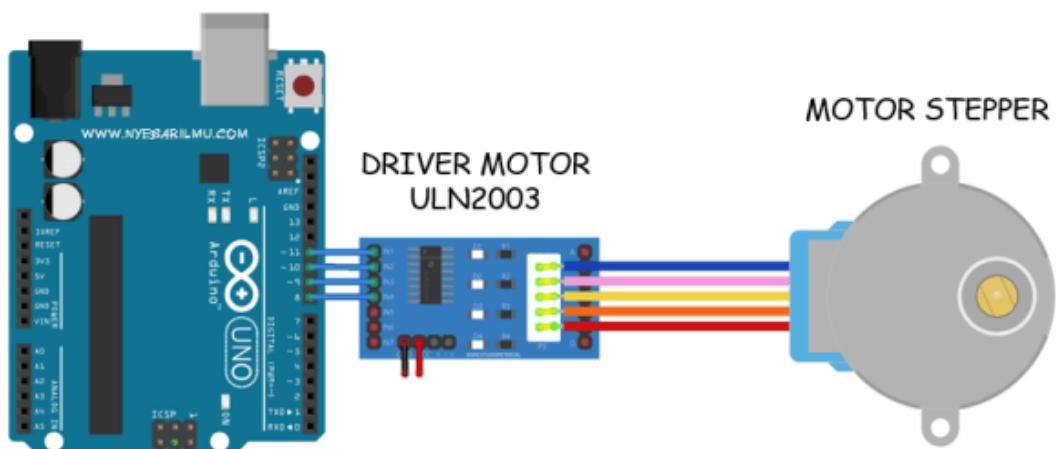
- Arduino Uno
- Komputer + Software IDE Arduino
- Module motor stepper
- Driver motor
- ULN2003
- Kabel Jumper

D. PERCOBAAN

Percobaan 1

1. Rangkai rangkaian dibawah ini.

Arduino UNO



2. Masukan sketch berikut pada IDE Arduino

```

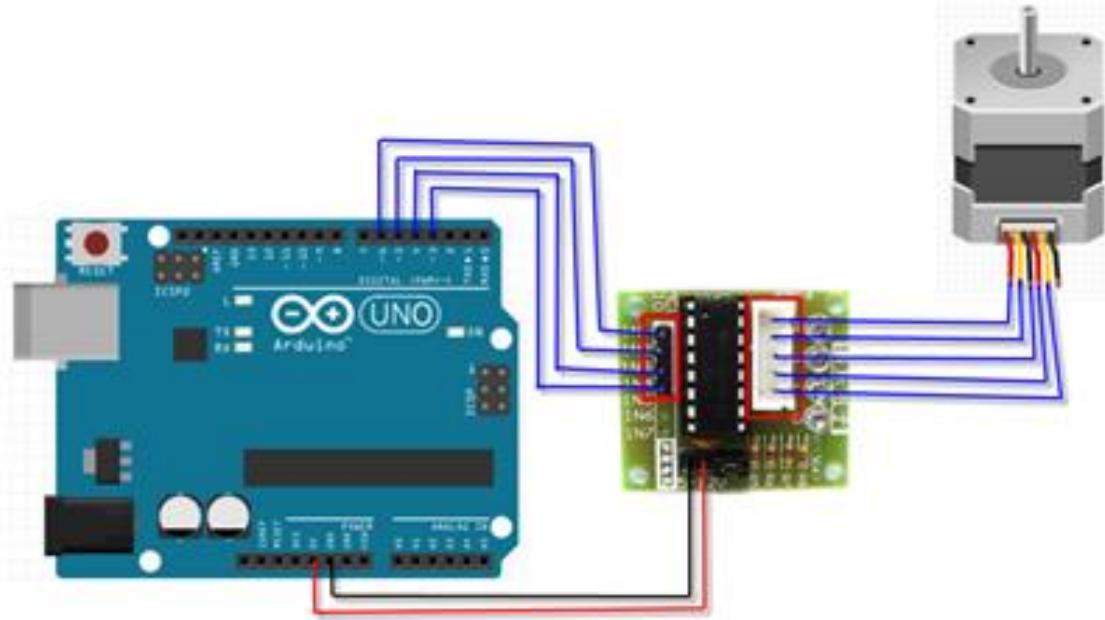
1 //Praktikum Sistem Digital dan Mikroprosesor
2 //Jobsheet 6
3 //Nama
4 //NIM
5 //Percobaan 1
6
7
8 //deklarasi pin yang digunakan
9 int A=8; //pin 8 - pin 11
10 int B=9;
11 int C=10;
12 int D=11;
13
14 void setup(){
15     //di deklarasikan sebagai output
16     pinMode(A,OUTPUT);
17     pinMode(B,OUTPUT);
18     pinMode(C,OUTPUT);
19     pinMode(D,OUTPUT);
20 }
21
22 void pergerakan_1(){
23     digitalWrite(A,00);
24     digitalWrite(D,1);
25     digitalWrite(B,00);
26     digitalWrite(C,1);
27 }
28
29 void pergerakan_2(){
30     digitalWrite(A,1);
31     digitalWrite(D,1);
32     digitalWrite(B,00);
33     digitalWrite(C,00);
34 }
35
36 void pergerakan_3(){
37     digitalWrite(A,1);
38     digitalWrite(D,00);
39     digitalWrite(B,1);
40     digitalWrite(C,00);
41 }
42
43 void pergerakan_4(){
44     digitalWrite(A,00);
45     digitalWrite(D,00);
46     digitalWrite(B,1);
47     digitalWrite(C,1);
48 }
49
50 void pergerakan(){
51     //pergerakan dari 00, 90', 180', 270', 360'
52     pergerakan_1(); delay(1000); //90'
53     pergerakan_2(); delay(1000); //180'
54     pergerakan_3(); delay(1000); //270'
55     pergerakan_4(); delay(1000); //360'
56 }
57
58 void loop(){
59     pergerakan();
60 }

```

3. Perhatikan apa yang terjadi pada motor stepper lalu analisis

Percobaan 2

1. Rangkai rangkaian dibawah ini.



2. Masukan sketch berikut pada IDE Arduino

```
1 //Praktikum Sistem Digital dan Mikroprosesor
2 //Jobsheet 6
3 //Nama
4 //NIM
5 //Percobaan 2
6
7
8 #include <AccelStepper.h> //library motor stepper
9
10 #define HALFSTEP 8           // definisi jumlah step
11
12 // definisi pin Arduino pada driver motor
13
14 #define motorPin1 6 // IN1 pada ULN2003 driver 1
15
16 #define motorPin2 5 // IN2 pada ULN2003 driver 1
17
18 #define motorPin3 4 // IN3 pada ULN2003 driver 1
19
20 #define motorPin4 3 // IN4 pada ULN2003 driver 1
21
22 // inisiasi urutan pin IN1-IN3-IN2-IN4 untuk library AccelStepper dengan motor 28BYJ-48
23
24 AccelStepper stepper1(HALFSTEP, motorPin1, motorPin3, motorPin2, motorPin4);
25
26 void setup()
27 {
28
29     stepper1.setMaxSpeed(1000.0);      //setting kecepatan maksimal motor
30
31     stepper1.setAcceleration(100.0); //setting akselerasi
32
33     stepper1.setSpeed(200);          //setting kecepatan
34
35     stepper1.moveTo(3000);         //setting untuk bergerak 3 putaran penuh
36
37 }
```

```
34     stepper1.moveTo(3000);      //setting untuk bergerak 3 putaran penuh
35
36
37 }
38
39 void loop()
40 {
41
42 //bila sampai posisi(3000) rubah ke posisi kebalik arah
43
44 if (stepper1.distanceToGo() == 0)
45 {
46
47 stepper1.moveTo(-stepper1.currentPosition());
48
49 Serial.println(stepper1.currentPosition());
50
51 }
52
53 stepper1.run(); // perintah menjalankan motor stepper
54
55 }
```

Normal text length:1.208 lines:55

Ln:5 Col:14 Sel:0|0

Windows (CR LF)

UTF-8

INS

3. Perhatikan apa yang terjadi pada motor stepper lalu analisis.

E. TUGAS

1. Buat program untuk mengontrol motor stepper sehingga motor bergerak 180° dan bergerak reverse – forward dengan kecepatan yang anda setting sendiri.
2. Buatlah program untuk mengontrol motor stepper dengan pergerakan searah jarum jam bergerak sejauh 90° lalu bergerak 180° ke arah sebaliknya.

JOBSHEET 7

KONTROL MOTOR SERVO MENGGUNAKAN ARDUINO

A. TUJUAN

1. Praktikan mengerti dan dapat memahami prinsip kerja motor servo
2. Praktikan dapat mengakses motor servo menggunakan arduino
3. Praktikan dapat mengontrol motor servo menggunakan arduino

B. TEORI DASAR

Motor servo adalah komponen elektronika yang berupa motor yang memiliki sistem *feedback* guna memberikan informasi posisi putaran motor aktual yang diteruskan pada rangkaian kontrol mikrokontroler. Pada dasarnya **motor servo** banyak digunakan sebagai aktuator yang membutuhkan posisi putaran motor yang presisi. Apabila pada motor DC biasa hanya dapat dikendalikan kecepatannya serta arah putaran, lain halnya pada motor servo yaitu penambahan besaran parameter yang dapat dikendalikan berdasarkan sudut/derajat. Adanya komponen potensiometer difungsikan sebagai *feedback* nilai yang akan diolah menjadi data posisi aktual. Sedangkan fungsi dari *controller* servo yaitu memberikan sinyal – sinyal PWM (*Pulse Width Modulator*) untuk menggerakan motor melalui kabel motor.

Macam tipe – tipe dari motor servo ini ada 2 yaitu **tipe standard** dan **tipe Continous**.

- Tipe standar berputarnya dibatasi sebesar 180° dan tipe ini sering banyak dipakai pada sistem robotika seperti *Arm Robot* / Robot Lengan.
- Tipe continuous mempunyai kriteria perputaran motornya sebesar 360° contoh aplikasinya pada mobil robot.

Pada setiap *body* servo terdapat informasi akan identitas tipe servo tersebut. Secara standar, motor servo terdiri atas 3 kabel yaitu kabel power / VCC, kabel GND serta kabel signal.

C. ALAT DAN BAHAN

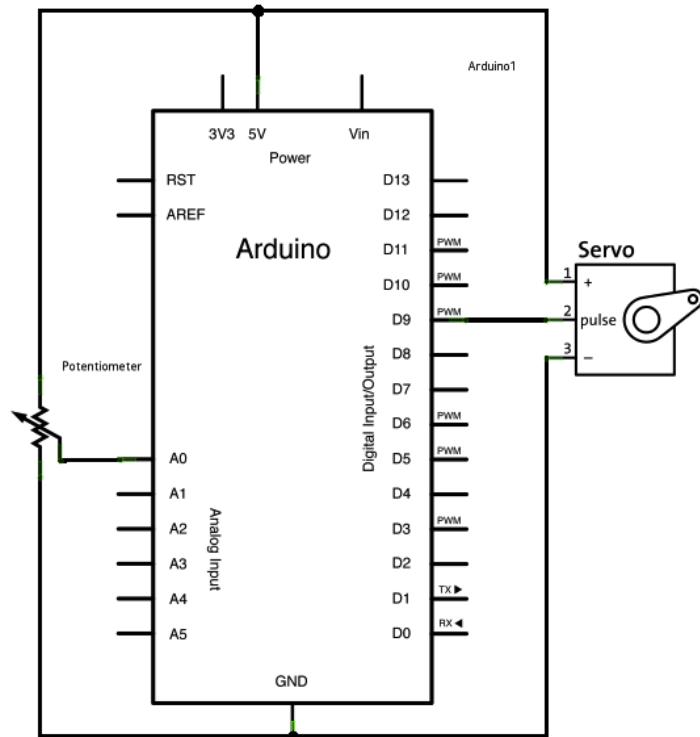
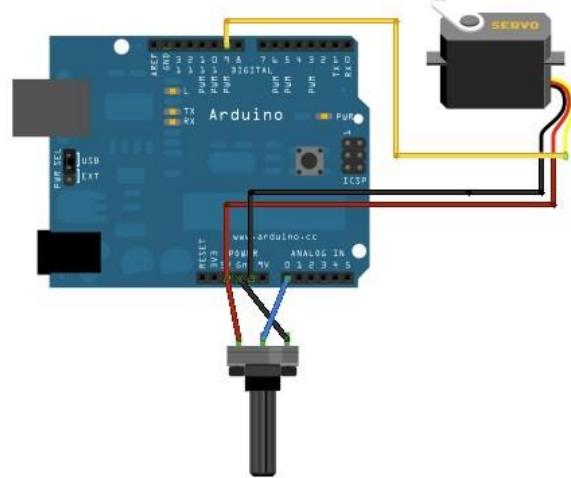
- Arduino Uno
- Komputer + Software IDE Arduino
- Micro Servo
- Kabel Jumper
- Potensiometer 10 KOhm

D. PERCOBAAN

Percobaan 1

Mengontrol motor servo menggunakan arduino dan potensiometer

Motor servo memiliki tiga kabel: daya, ground, dan sinyal. Kabel daya biasanya berwarna merah, dan harus dihubungkan ke pin 5V pada papan Arduino atau Genuino. Kabel arde biasanya berwarna hitam atau cokelat dan harus dihubungkan ke pin arde di papan tulis. Pin sinyal biasanya berwarna kuning atau oranye dan harus terhubung ke pin 9 pada arduino. Potensiometer harus dihubungkan dengan kabel sehingga dua pin luarnya terhubung ke daya (+ 5V) dan arde, dan pin tengahnya terhubung ke input analog 0 pada papan.



Source code untuk percobaan 1

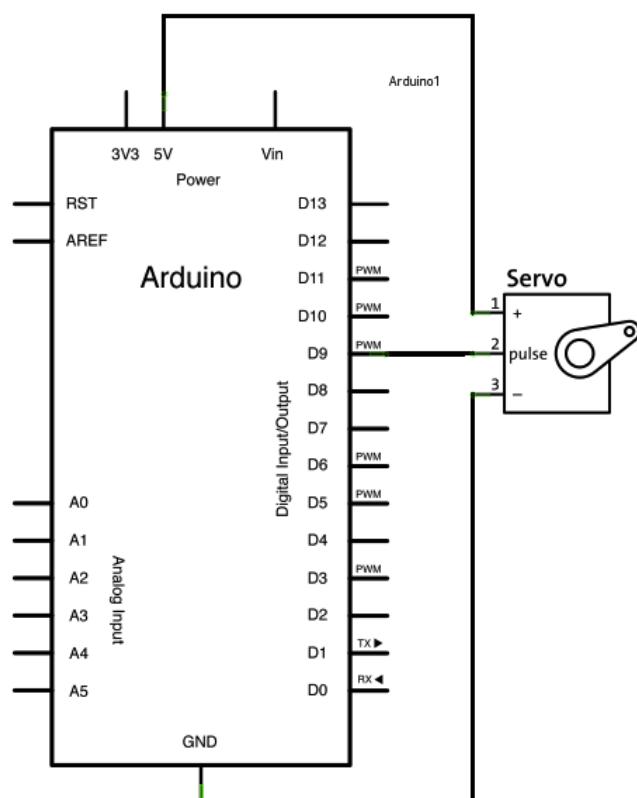
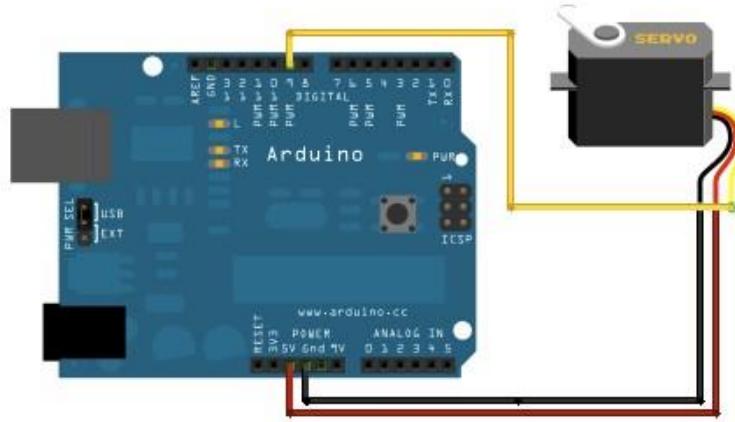
```
10 #include <Servo.h>
11
12 Servo myservo; // create servo object to control a servo
13
14 int potpin = 0; // analog pin used to connect the potentiometer
15 int val; // variable to read the value from the analog pin
16
17 void setup() {
18     myservo.attach(9); // attaches the servo on pin 9 to the servo object
19 }
20
21 void loop() {
22     val = analogRead(potpin); // reads the value of the potentiometer (value between 0 and 1023)
23     val = map(val, 0, 1023, 0, 180); // scale it to use it with the servo (value between 0 and 180)
24     myservo.write(val); // sets the servo position according to the scaled value
25     delay(15); // waits for the servo to get there
26 }
27
```

Perhatikan apa yang terjadi lalu analisis

Percobaan 2

Menyapu poros motor servo RC bolak-balik melintasi 180 derajat.

Motor servo memiliki tiga kabel: daya, ground, dan sinyal. Kabel daya biasanya berwarna merah, dan harus dihubungkan ke pin 5V pada papan Arduino atau Genuino. Kabel arde biasanya berwarna hitam atau cokelat dan harus dihubungkan ke pin arde di papan tulis. Pin sinyal biasanya berwarna kuning atau oranye dan harus terhubung ke pin 9 pada arduino.



Source code untuk percobaan 2

```
10 #include <Servo.h>
11
12 Servo myservo; // create servo object to control a servo
13 // twelve servo objects can be created on most boards
14
15 int pos = 0; // variable to store the servo position
16
17 void setup() {
18     myservo.attach(9); // attaches the servo on pin 9 to the servo object
19 }
20
21 void loop() {
22     for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
23         // in steps of 1 degree
24         myservo.write(pos); // tell servo to go to position in variable 'pos'
25         delay(15); // waits 15ms for the servo to reach the position
26     }
27     for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
28         myservo.write(pos); // tell servo to go to position in variable 'pos'
29         delay(15); // waits 15ms for the servo to reach the position
30     }
31 }
```

Amati dan analisis apa yang terjadi pada motor servo tersebut.

E. TUGAS

1. Jelaskan prinsip kerja motor servo
2. Buat program untuk mengontrol motor servo agar berputar searah jarum jam 45° dengan delay 500ms dan berbalik berputar berlawanan jarum jam 45° dengan delay 500ms.
3. Buat program untuk mengontrol motor servo agar bergerak searah jarum jam 45° dengan delay 1000ms lalu dilanjutkan sampai 0° dengan delay 1000ms kemudian putar kembali ke posisi center yaitu 90° dengan delay 1000 ms. Lakukan pada arah berlawanan jarum jam seperti tadi, putar ke arah 135° dengan delay 1000ms lalu diteruskan ke arah 180° dengan delay 100ms kemudian kembalikan ke center yaitu 90° dengan delay 1000ms.

JOBSHEET 8

ARDUINO SEBAGAI PENGONTROL LED DAN BUZZER

A. Tujuan

1. Mahasiswa dapat memahami konsep dasar mikrokontroler arduino.
2. Mahasiswa dapat memahami blok komponen arduino.
3. Mahasiswa dapat membuat program menggunakan arduino IDE.
4. Mahasiswa dapat membuat konsep dan aplikasi arduino.

B. Bahan

• Arduino Uno	x1
• <i>Buzzer</i>	x1
• Relay	x1
• Transistor 2N3904	x1
• Project Board	x1
• LED	x2
• Trimpot 50k	x1
• Push button	x2
• Kabel Jumper	secukupnya

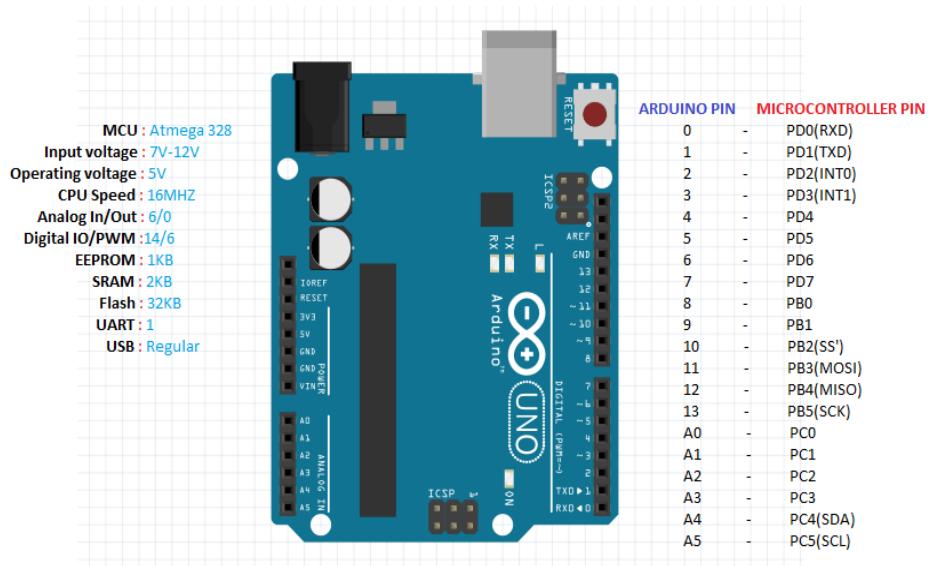
C. Alat

- Tang pemotong (opsional)
- Multimeter (opsional)
- Obeng minus
- Laptop dengan software Arduino IDE.

Ringkasan Teori

1. Arduino

Arduino adalah salah satu kit mikrokontroler yang berbasis pada Atmega328. Modul ini sudah dilengkapi dengan berbagai hal yang dibutuhkan untuk mendukung mikrokontroler untuk bekerja, hanya sambungkan ke power suply atau sambungkan melalui kabel USB ke PC selanjutnya Arduino Uno ini sudah siap digunakan. Arduino Uno ini memiliki 14 pin digital input/output, 6 analog input, sebuah resonator keramik 16MHz, koneksi USB, konektor power input, ICSP header, dan sebuah tombol reset.



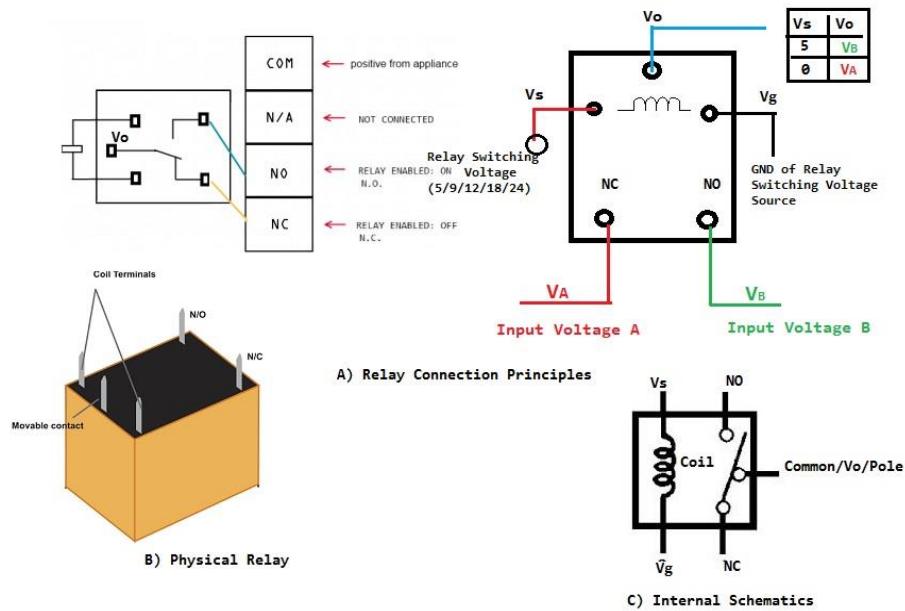
Gambar 1. Detail Arduino Uno tampak atas.

2. Relay

Relay terdiri dari *coil* dan *switch*(Pole, No, Nc), untuk mengaktifkan switch ini tergantung dari pencatuan coil, coil sendiri memiliki tegangan catu yang berbeda-beda, pada praktikum ini menggunakan relay dengan coil 5V DC, namun pada umumnya terdapat coil dengan tegangan catu 5V,12V,24V,48V. Ketika coil dicatuh maka Pole akan tersambung pada No (normaly open) pada kasus ini normaly open berarti ketika coil tidak di catuh maka titik pole dan No tidak terhubung. Berartu jika coil tidak di catuh maka Pole akan tersambung pada Nc (normaly close).



Gambar 2. Tampilan fisik relay 5V.



Gambar 3. Gambaran prinsip kerja Relay.

3. Buzzer Melody

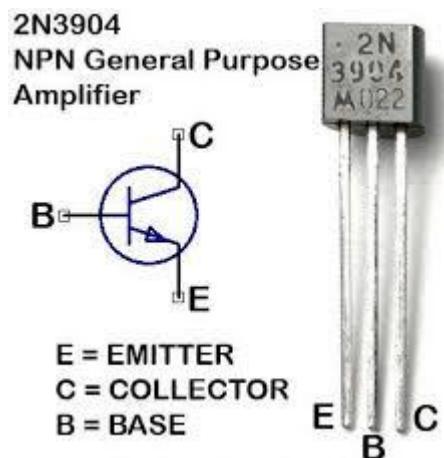
Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Perangkat elektronika ini terbuat dari *elemen piezoceramics* yang diletakkan pada suatu diafragma yang mengubah getaran/vibrasi menjadi gelombang suara. Buzzer menggunakan resonansi untuk memperkuat intensitas suara. Berbeda dengan active *buzzer*, *buzzer* melody akan bersuara lantang bila diberi frekuensi tertentu. Tegangan catu *buzzer* juga berbeda-beda, pada praktik ini menggunakan *buzzer* 5v.



Gambar 4. Tampilan fisik *Buzzer*.

4. Transistor 2N3904

Transistor ini adalah jenis NPN berarti transistor akan bekerja apabila *base emitter* diberi bias forward dan basis *colector* diberi bias reverse, berarti bahwa ketika ada arus yang mengalir dari *base* ke *emitter* maka ada arus yang mengalir dari *colector* ke *emitter*.



Gambar 5. Konfigurasi kaki transistor 2N3904.

5. LED

LED adalah diode yang mengemisikan cahaya ketika diberi bias forward, artinya *anode* diberi tegangan yang lebih tinggi dibanding tegangan pada *cathode*.



Gambar 6. Tampilan Fisik LED.

6. Trimpot

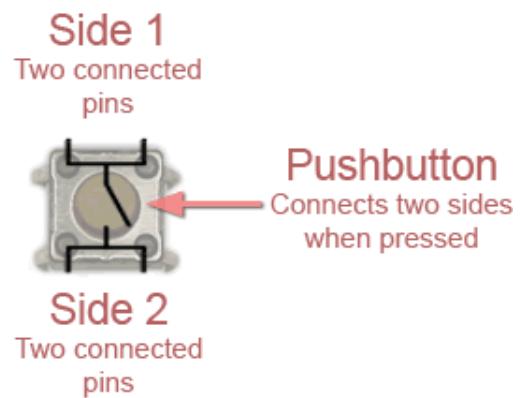
Trimpot adalah golongan *variable resistor*, dimana nilai resistansinya akan berubah-ubah ketika kita memutarnya. Prinsipnya adalah pembagi tegangan, trimpot terdiri dari 2 buah resistor yang diseri.



Gambar 7. Tampilan fisik Trimpot.

7. Push Button

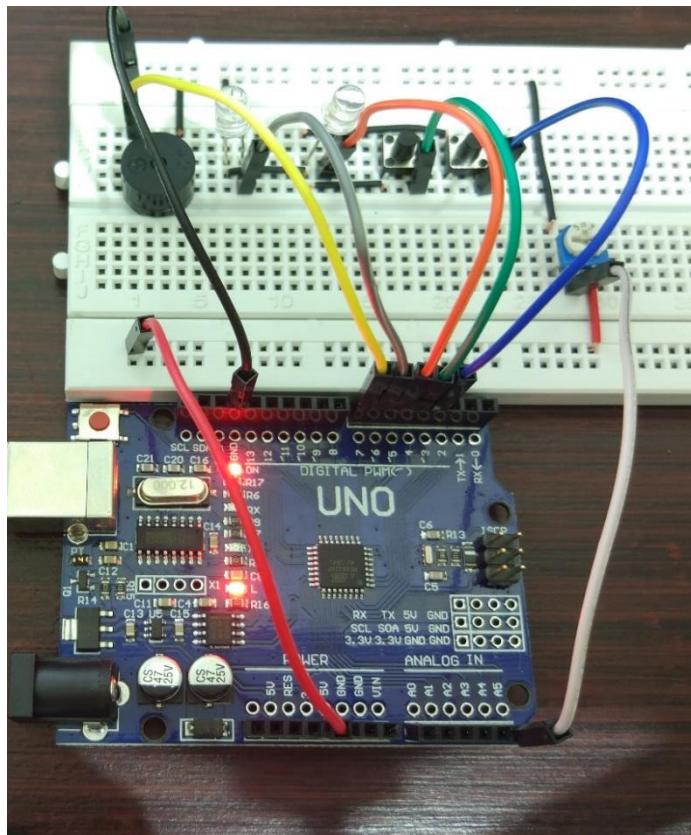
Push button memiliki jenis push off dan push on. Pada percobaan ini kita menggunakan push ON artinya ketika ditekan maka kedua sisi akan tersambung. Sebaliknya jika push off, ketika ditekan maka kedua sisi akan terputus.



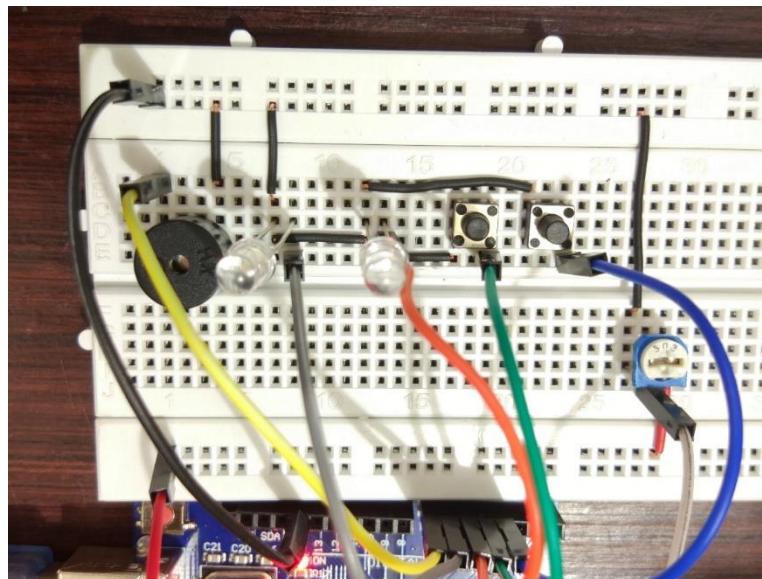
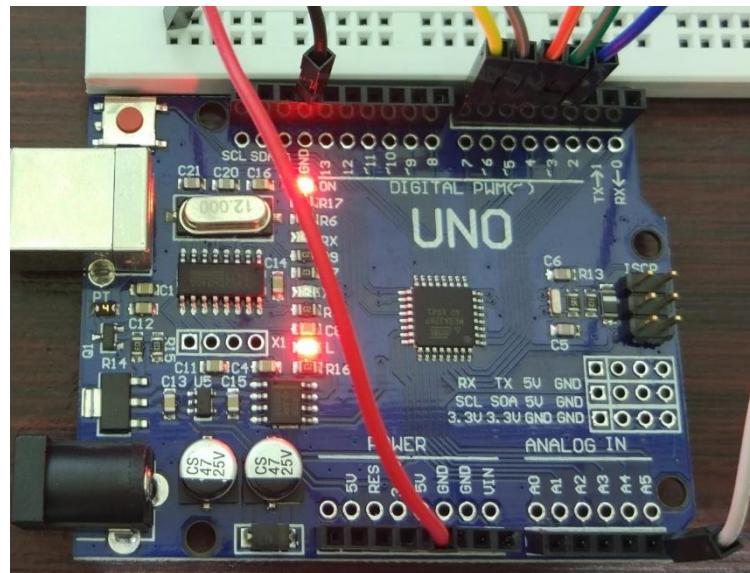
Gambar 8. Gambaran prinsip kerja pushbutton.

D. Langkah Kerja

1. Siapkan alat dan bahan, kemudian susun rangkaian dibawah pada project board, perhatikan warna kabel dan sambungannya seperti gambar 9 dan 10. Sambungkan usb terakhir setelah rangkaian tersusun.



Gambar 9. Sambungan rangkaian tampak atas.



Gambar 10. Sambungan rangkaian LED, *buzzer*, *pushbutton*.

Keterangan :

- Kabel Hitam adalah kabel ground terhubung ke ground arduino.
- Setiap kabel hitam terhubung dengan negative komponen, seperti katoda LED, negative buzzer, salah satu bagian kaki push button.
- Kabel merah adalah kabel positif 5V terhubung ke pin 5V arduino.
- Kabel biru tersambung antara pin nomor 3 arduino ke salah satu kaki push button 1, sisi lain dari kaki yang terhubung ke negatif.
- Kabel hijau tersambung antara pin nomor 4 arduino ke salah satu kaki push button 2 sisi lain dari kaki yang terhubung ke negatif.
- Kabel orange tersambung antara pin nomor 5 arduino ke anoda LED1.
- Kabel abu tersambung antara pin nomor 6 arduino ke anoda LED2.
- Kabel kuning tersambung antara pin nomor 7 arduino ke positif buzzer.
- Kabel putih tersambung antara kaki analog 5 (A5) arduino ke kaki tengah dari trimpot.

- Apabila sudah tersusun, sambungkan kabel usb dari pc ke arduino. Buka Arduino IDE pada laptop, kemudian ikuti script dibawah.

2.1 Menyalakan LED sintaks decision.

Part 1 (menggunakan sintaks IF else)

```
///////////////////////////////
// Menyalakan LED menggunakan push button //
// dengan IF          //
// part 1            //
/////////////////////////////
```

```
#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6

int pushbutton1,pushbutton2;

void setup()
{
    pinMode(PB1, INPUT);
    pinMode(PB2, INPUT);
    digitalWrite(PB1, HIGH);
    digitalWrite(PB2, HIGH);
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);

}

void loop()
{
    pushbutton1=digitalRead(PB1);
    pushbutton2=digitalRead(PB2);

    if(pushbutton1==0)
    {
        digitalWrite(LED1,HIGH);
    }
    else
    {
        digitalWrite(LED1,LOW);
    }

    if(pushbutton2==0)
    {
        digitalWrite(LED2,HIGH);
    }
    else
    {
        digitalWrite(LED2,LOW);
    }
}
```

```
}
```

Part 2 (menggunakan sintaks FOR)

```
//////////  
// Menyalakan LED berulang kali ketika //  
// tombol ditekan menggunakan for //  
// part 2 //  
//////////
```

```
#define PB1 3  
#define PB2 4  
#define LED1 5  
#define LED2 6
```

```
int pushbutton1,pushbutton2;
```

```
void setup()  
{  
    pinMode(PB1, INPUT);  
    pinMode(PB2, INPUT);  
    digitalWrite(PB1, HIGH);  
    digitalWrite(PB2, HIGH);  
    pinMode(LED1, OUTPUT);  
    pinMode(LED2, OUTPUT);  
}
```

```
void loop()  
{  
    pushbutton1=digitalRead(PB1);  
    pushbutton2=digitalRead(PB2);
```

```
    if(pushbutton1==0)  
    {  
        for(int i=0; i<5; i++)  
        {  
            digitalWrite(LED1,HIGH);  
            delay(500);  
            digitalWrite(LED1,LOW);  
            delay(500);  
        }  
    }
```

```
    if(pushbutton2==0)  
    {  
        for(int i=0; i<5; i++)  
        {  
            digitalWrite(LED2,HIGH);  
            delay(500);  
        }
```

```

        digitalWrite(LED2,LOW);
        delay(500);
    }
}
}

```

Part 3 (menggunakan sintaks while)

```

///////////////////////////////
// Menyalakan LED berkedip kemudian      //
// menyala berulang menggunakan while      //
// part 3                                //
/////////////////////////////

```

```

#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6

int pushbutton1,pushbutton2;
char count;

void setup()
{
    pinMode(PB1, INPUT);
    pinMode(PB2, INPUT);
    digitalWrite(PB1, HIGH);
    digitalWrite(PB2, HIGH);
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);
}

```

```

void loop()
{
    pushbutton1=digitalRead(PB1);
    pushbutton2=digitalRead(PB2);

    while(count<10)
    {
        count=count+1;
        delay(1000);
        digitalWrite(LED1,HIGH);
        delay(200);
        digitalWrite(LED1,LOW);
        digitalWrite(LED2,HIGH);
        delay(200);
        digitalWrite(LED2,LOW);
    }
    digitalWrite(LED1,HIGH);
    digitalWrite(LED2,HIGH);
}

```

```

delay(500);
digitalWrite(LED1,LOW);
digitalWrite(LED2,LOW);
delay(500);
}

```

Part 4 (menggunakan sintaks switch case)

```

///////////
// Menyalakan LED bergantian dengan tombol //
// menggunakan switch case          //
// part 4                      //
///////////

#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6

int pushbutton1,pushbutton2;
char count;

void setup()
{
    pinMode(PB1, INPUT);
    pinMode(PB2, INPUT);
    digitalWrite(PB1, HIGH);
    digitalWrite(PB2, HIGH);
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);

}

void loop()
{

    pushbutton1=digitalRead(PB1);
    pushbutton2=digitalRead(PB2);

    switch(pushbutton1)
    {
        case 1:
            digitalWrite(LED1,HIGH);
            digitalWrite(LED2,HIGH);
            break;
        case 0:
            for(int i=0;i<10;i++)
            {
                digitalWrite(LED1,HIGH);
                digitalWrite(LED2,HIGH);
                delay(100);
                digitalWrite(LED1,LOW);
            }
    }
}

```

```

digitalWrite(LED2,LOW);
delay(100);
}
break;
}

switch(pushbutton2)
{
case 1:
digitalWrite(LED1,HIGH);
digitalWrite(LED2,HIGH);
break;
case 0:
for(int i=0;i<10;i++)
{
digitalWrite(LED1,HIGH);
digitalWrite(LED2,LOW);
delay(100);
digitalWrite(LED1,LOW);
digitalWrite(LED2,HIGH);
delay(100);
}
break;
}
}

```

Analog input dan PWM

Part 1 (serial monitor analog input menggunakan trimpot)

```

///////////////////////////////
// Analog input dan PWM      //
// input Trimpot dan serial monitor   //
// part 1                      //
/////////////////////////////

```

```

#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6
#define trimpot A5
int variabelADC;

void setup()
{
  Serial.begin(9600);
  pinMode(trimpot,INPUT);
}

void loop() {
  variabelADC=analogRead(trimpot);
  Serial.print("Nilai Analog 10bit dalam desimal : ");
  Serial.println(variabelADC);
  delay(100);
}

```

```
}
```

Buka serial monitor dengan klik Tools > Serial Monitor

Putar-putar trimpot dan amati serta analisis hasilnya

Part 2 (mengatur intensitas LED menggunakan trimpot)

```
///////////////////////////////
```

```
// Analog input dan PWM atur LED dengan //  
// input Trimpot dan serial monitor //  
// part 2 //  
///////////////////////////////
```

```
#define PB1 3
```

```
#define PB2 4
```

```
#define LED1 5
```

```
#define LED2 6
```

```
#define trimpot A5
```

```
int variabelADC, ADCbaru;
```

```
float vo;
```

```
void setup()
```

```
{
```

```
  Serial.begin(9600);
```

```
  pinMode(trimpot,INPUT);
```

```
}
```

```
void loop()
```

```
{
```

```
  variabelADC=analogRead(trimpot);
```

```
  vo=(5*variabelADC)/1023;
```

```
  ADCbaru=map(variabelADC,0,1023,0,255);
```

```
  analogWrite(LED1,ADCbaru);
```

```
  analogWrite(LED2,ADCbaru);
```

```
  Serial.print("Nilai ADC baru : ");
```

```
  Serial.print(ADCbaru);
```

```
  Serial.print(" Nilai tegangan : ");
```

```
  Serial.print(vo);
```

```
  Serial.println(" V ");
```

```
  delay(100);
```

```
}
```

Buka serial monitor dengan klik Tools > Serial Monitor

Putar-putar trimpot dan amati serta analisis hasilnya

Part 3 (mengatur kecepatan perpindahan LED menggunakan trimpot)

```
///////////////////////////////
```

```
// Analog input dan PWM atur kecepatan LED//
```

```
// input Trimpot dan serialmonitor //
```

```
// part3 //
```

```
///////////////////////////////
```

```
#define PB1 3
```

```

#define PB2 4
#define LED1 5
#define LED2 6
#define trimpot A5
int variabelADC, ADCbaru;

void setup()
{
    Serial.begin(9600);
    pinMode(trimpot,INPUT);
    pinMode(LED1,OUTPUT);
    pinMode(LED2,OUTPUT);
}

void loop()
{
    variabelADC=analogRead(trimpot);
    ADCbaru=map(variabelADC,0,1023,0,2000);
    digitalWrite(LED1,HIGH);
    digitalWrite(LED2,LOW);
    delay(ADCbaru);
    digitalWrite(LED1,LOW);
    digitalWrite(LED2,HIGH);
    delay(ADCbaru);
    digitalWrite(LED1,HIGH);
    digitalWrite(LED2,HIGH);
    delay(ADCbaru);
    digitalWrite(LED1,LOW);
    digitalWrite(LED2,LOW);
    Serial.print("Nilai delay: ");
    Serial.print(ADCbaru);
    Serial.println(" ms");
    delay(100);
}

```

Buka serial monitor dengan klik Tools > Serial Monitor
Putar-putar trimpot dan amati serta analisis hasilnya

Buzzer

Part 1 (mengatur frekuensi buzzer)

```

///////////////////////////////
// Buzzer atur frekuensi      //
// sweep frekuensi           //
// part1                      //
///////////////////////////////

```

```

#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6
#define buzzer 7
int freq=0;

```

```

void setup() {
  pinMode(buzzer,OUTPUT);

}

void loop()
{
awal:
tone(buzzer,freq,100);
freq=freq+100;
delay(200);
if(freq>=3500)
{
  freq=3500;
  goto akhir;
}
goto awal;

akhir:
tone(buzzer,freq,100);
freq=freq-100;
delay(200);
if(freq<=0)
{
  goto awal;
}
goto akhir;

}

```

Part 2 (membuat melodi lagu menggunakan buzzer)

```

///////////////////////////////
// Buzzer atur frekuensi memnbuat lagu  //
// nada pitch.h manual          //
// part 2                      //
/////////////////////////////

```

```

#define C6 1047
#define D6 1175
#define E6 1319
#define F6 1397
#define G6 1568
#define A6 1760
#define C7 2093
#define B6 1976

```

```

#define buzzer 7

void setup()

```

```
{  
pinMode(buzzer,OUTPUT);  
  
}  
  
void loop()  
{  
tone(buzzer,C6,500);  
delay(1000);  
tone(buzzer,D6,200);  
delay(1000);  
tone(buzzer,E6,400);  
delay(1000);  
tone(buzzer,F6,400);  
delay(1000);  
tone(buzzer,G6,600);  
delay(1000);  
tone(buzzer,E6,200);  
delay(1000);  
tone(buzzer,C6,500);  
delay(1000);  
tone(buzzer,A6,500);  
delay(1000);  
tone(buzzer,C7,200);  
delay(1000);  
tone(buzzer,B6,200);  
delay(1000);  
tone(buzzer,A6,500);  
delay(1000);  
tone(buzzer,G6,500);  
delay(1000);  
noTone(buzzer);  
delay(1000);  
  
}
```

Part 3 (merubah frekuensi buzzer menggunakan potensio meter)

```
///////////////////////////////
// Buzzer      Atur frekuensi dengan      //
// input Trimpot dan serial monitor    //
// part3          //
///////////////////////////////
```

```
#define PB1 3
#define PB2 4
#define LED1 5
#define LED2 6
#define trimpot A5
#define buzzer 7
int variabelADC, freq;

void setup() {
  Serial.begin(9600);
  pinMode(trimpot,INPUT);
  pinMode(buzzer,OUTPUT);

}

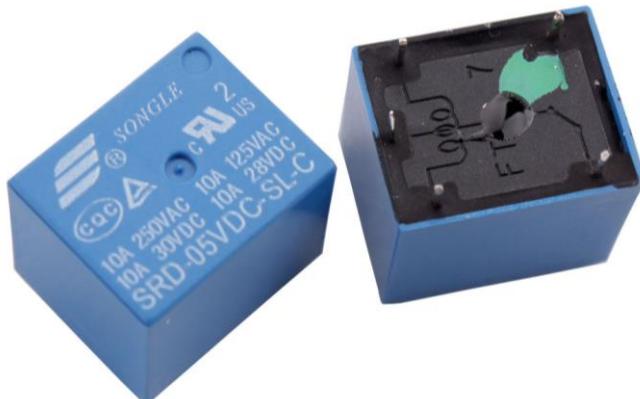
void loop()
{
  variabelADC=analogRead(trimpot);
  freq=map(variabelADC,0,1023,0,4000);
  Serial.print("frequensi : ");
  Serial.print(freq);
  Serial.println(" Hz");
  tone(buzzer,freq,200);
  delay(1000);
}
```

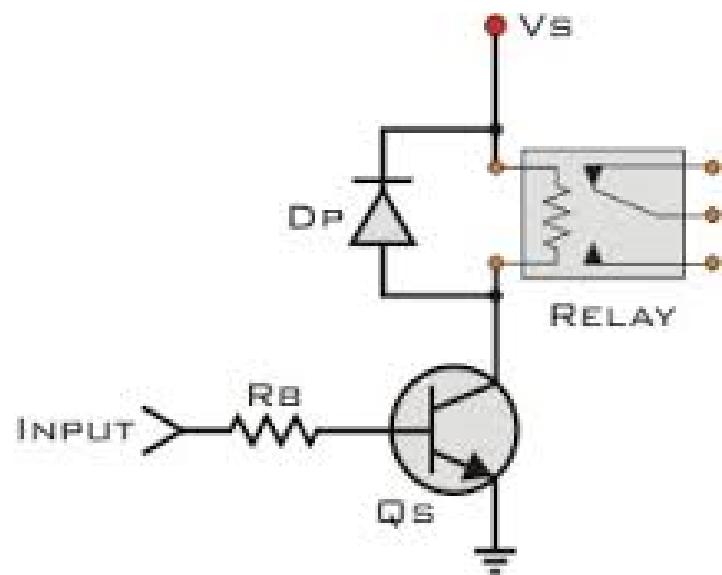
Buka serial monitor dengan klik Tools > Serial Monitor

Putar-putar trimpot dan amati serta analisis hasilnya

E. Tugas

Dari praktikum diatas, pikirkan sebuah konsep program berikut aplikasinya menggunakan bahan-bahan yang telah dicoba dan sebagai tambahannya gunakan relay 5V.





JOBSHEET 10

DISAIN SKEMATIK MENGGUNAKAN SOFTWARE XILINX ISE

Pada tahun 1980-an Departemen Pertahanan Amerika berinisiatif mengembangkan sistem pembuatan rangkaian elektronik terintegrasi secara cepat yang disebut sebagai very- high-speed integrated circuits (VHSIC) hardware description language atau lebih dikenal dengan nama HDL. Perangkat lunak ini ditujukan untuk mendeskripsikan perangkat keras komponen-komponen elektronika yang standar dan umum digunakan seperti gerbang logika, flip-flop rangkaian kombinasional, rangkaian serial, rangkaian memori, kontroler dan mikroprosesor. HDL dapat memodelkan sebuah sistem yang ingin dikembangkan dari berbagai sudut pandang, baik secara struktural maupun perilaku pada semua level deskripsi. Perangkat lunak HDL bersifat independen dan terpisah dari perangkat lunak yang digunakan untuk proses kompilasi, pemrograman untuk pembuatan komponen dan simulasi serta bersifat independen terhadap teknologi elektronik yang digunakan dalam bentuk implementasi riilnya seperti FPGA, CPLD, ASIC dan lainnya.

Penggunaan HDL akan memudahkan pengguna karena dapat menghindari penggunaan tabel kebenaran dalam menyederhanakan fungsi boolean yang ingin diimplementasi. Cukup dengan menyatakan secara eksplisit fungsi boolean tersebut dan saat proses kompilasi, compiler berfungsi untuk mengubah dan menyederhanakan code fungsi boolean tersebut menjadi gerbang-gerbang logika dan menyusunnya menjadi sebuah rangkaian elektronik. Pengujian fungsi logika disain rangkaian menggunakan model simulasi dalam bentuk interface diagram sinyal untuk menggambarkan logika kerja rangkaian tersebut. Model simulasi ini umumnya disebut testbench.

Hardware Description Language HDL yang umum dikenal adalah VHDL, Verilog dan SystemVerilog. Ketiganya memiliki sintaks yang berbeda, cara mendeskripsikan kekuatan dan waktu propagasi sinyal. hingga saat ini, penggunaan HDL telah memungkinkan merancang dan mengimplementasikan sebuah sistem yang kompleks kedalam chip (SoC: System on Chip). Dalam bab ini diuraikan tentang dasar-dasar pemrograman HDL dengan dua pendekatan yaitu disain rangkaian melalui pendekatan Schematic menggunakan pendekatan simulasi Verilog dan disain rangkaian dan simulasi menggunakan VHDL.

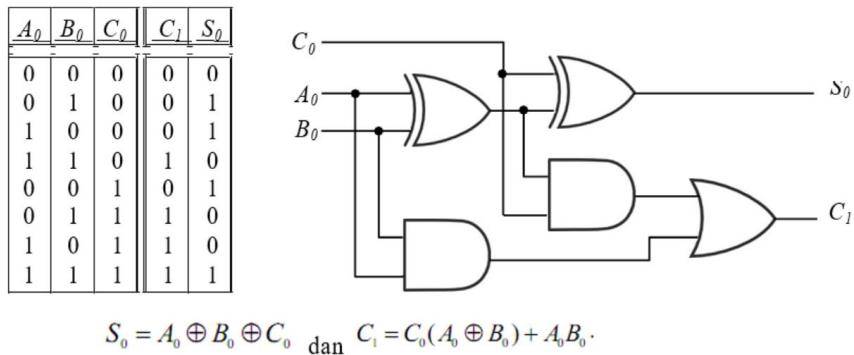
Perangkat lunak HDL yang digunakan dalam buku adalah dari Xilinx ISE WebPACK, ISE Design Suite 14.7. Pembaca dapat men-download secara gratis pada alamat web: www.xilinx.com, dan tutorial penggunaanya dapat di-download di alamat <http://www.xilinx.com/support/techsup/tutorials/>.

Implementasi Rangkaian FPGA Melalui Pendekatan Schematic

Sasaran dari sub bab ini adalah menguraikan tentang urutan proses penggunaan perangkat lunak Xilinx ISE Design Suite 14.7 dalam membuat project, mendisain rangkaian, mensimulasikan, sintesis dan mengimplementasikannya ke dalam IC FPGA melalui pendekatan schematic. Perangkat lunak ini dilengkapi dengan berbagai tools model dan jenis rangkaian digital sebagaimana IC rangkaian digital yang telah digunakan secara umum dalam dunia nyata. Ada empat bagian penting yang akan diuraikan yaitu : Bagaimana memasukkan disain rangkaian elektronika kedalam model simulasi rangkaian melalui pendekatan schematic; Bagaimana mengakses dan menggunakan simbol-simbol rangkaian elektronika dari library untuk dirangkai menjadi satu rangkaian elektronika baru; Bagaimana mensimulasikan operasi fungsi rangkaian yang telah dibuat;

Bagaimana mentransfer model simulasi rangkaian kedalam IC FPGA dan melakukan rekonfigurasi IC FPGA dengan disain model rangkaian yang telah dibuat.

Hal yang pertama harus disiapkan sebelum melakukan disain schematic adalah mempersiapkan persamaan boolean dan rangkaian elektronika yang ingin diimplementasi. Misalkan, mulai dari disain rangkaian yang paling sederhana seperti "full-adder" pada gambar 3.1. Turunan persamaan boolean dan bentuk rangkaian dari "full-adder" telah diuraikan dalam bab 2 (sub-bab 2.3.1.2). Setelah persamaan boolean dan atau bentuk rangkaian telah siap, maka sebuah projek disain rangkaian menggunakan schematic dapat dimulai.



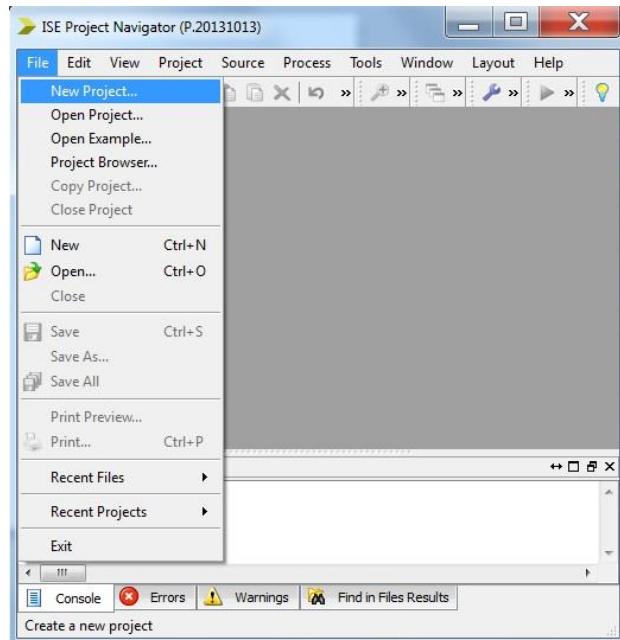
Gambar 3.1. Persamaan boolean dan bentuk rangkaian full-adder.

PROSEDUR PRAKTIKUM

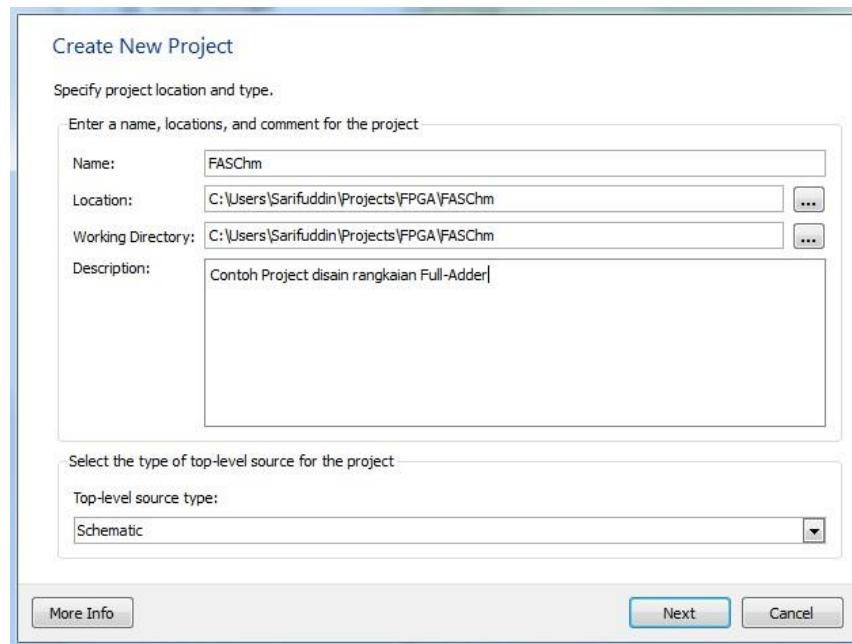
Langkah pertama adalah menjalankan perangkat lunak ISE Design Suite 14.7. Selanjutnya untuk membuat sebuah Project (rangkaian) baru ikuti tahapan berikut:

1. Pada layar Project Navigator pilih/click menu File → New Project (seperti pada gambar 3.1). Setelah menu New Project dipilih maka akan terbuka jendela baru Creat New Project seperti yang terlihat pada gambar 3.2.
2. Pada jendela ini tulis nama project pada panel Name: tulis nama project misalnya "FASchem". Saat pemberian nama project tersebut, pada panel location dan working directory akan ditambahkan secara otomatis ke lokasi dan ke directory mana project tersebut akan disimpan. Bila ingin mengganti nama location dan working directory, click tombol browse (...) sebelah kanan panel, kemudian pilih sesuai dengan location dan working directory yang diinginkan. Pada panel ke-4 terdapat bagian description, dapat ditambahkan teks penjelasan dari project yang dibuat (boleh juga dikosongkan). Pada panel bagian bawah terdapat pilihan top-level source type, pilih "Schematic" lalu tekan tombol "Next", maka akan tampil layar baru Project Settings seperti pada gambar 3.3.

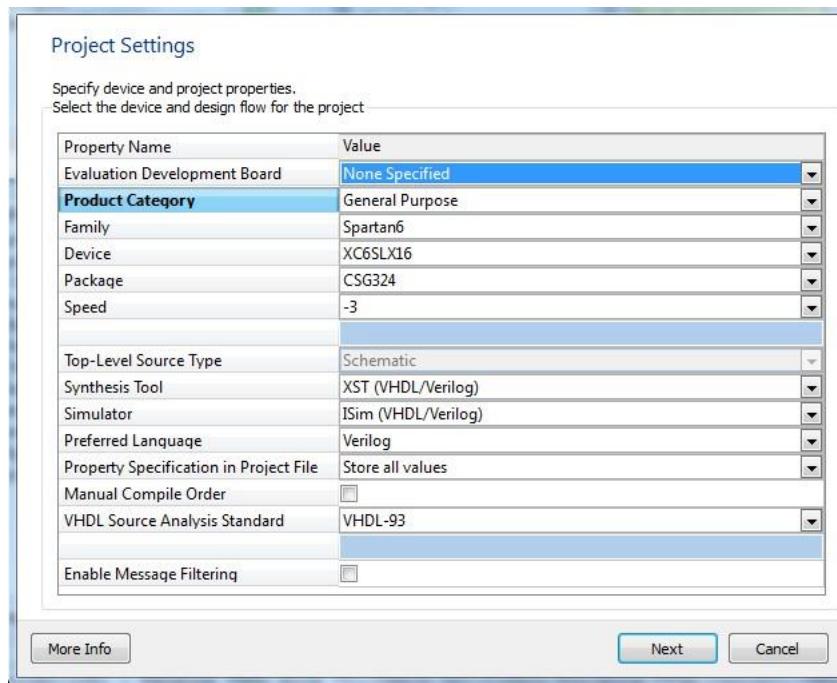
$$S_0 = A_0 \oplus B_0 \oplus C_0 \text{ dan } C_1 = C_0(A_0 \oplus B_0) + A_0B_0.$$



Gambar 3.1. Layar interface ISE Project Navigator

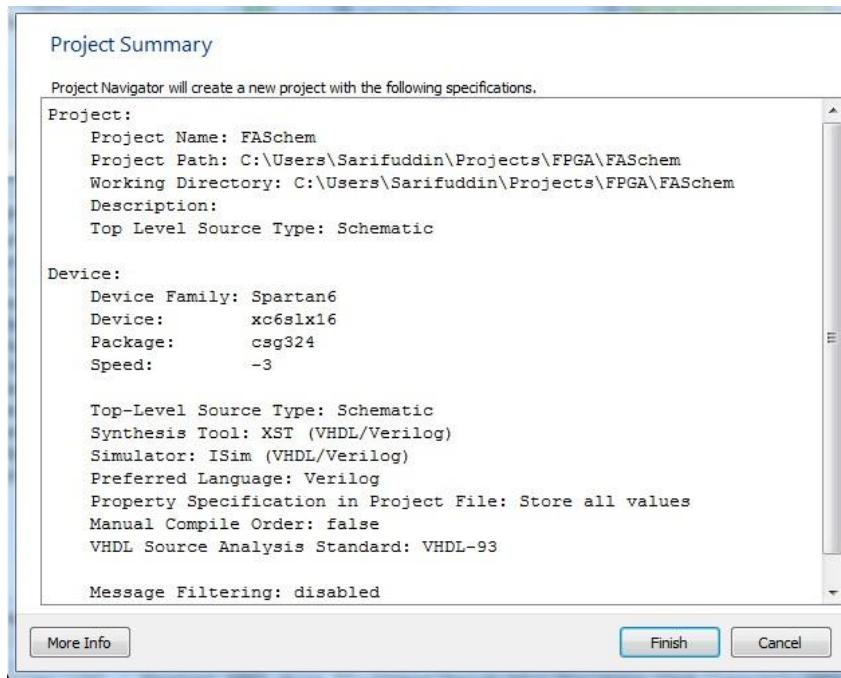


Gambar 3.2. Layar interface untuk menuliskan nama project dan lokasinya



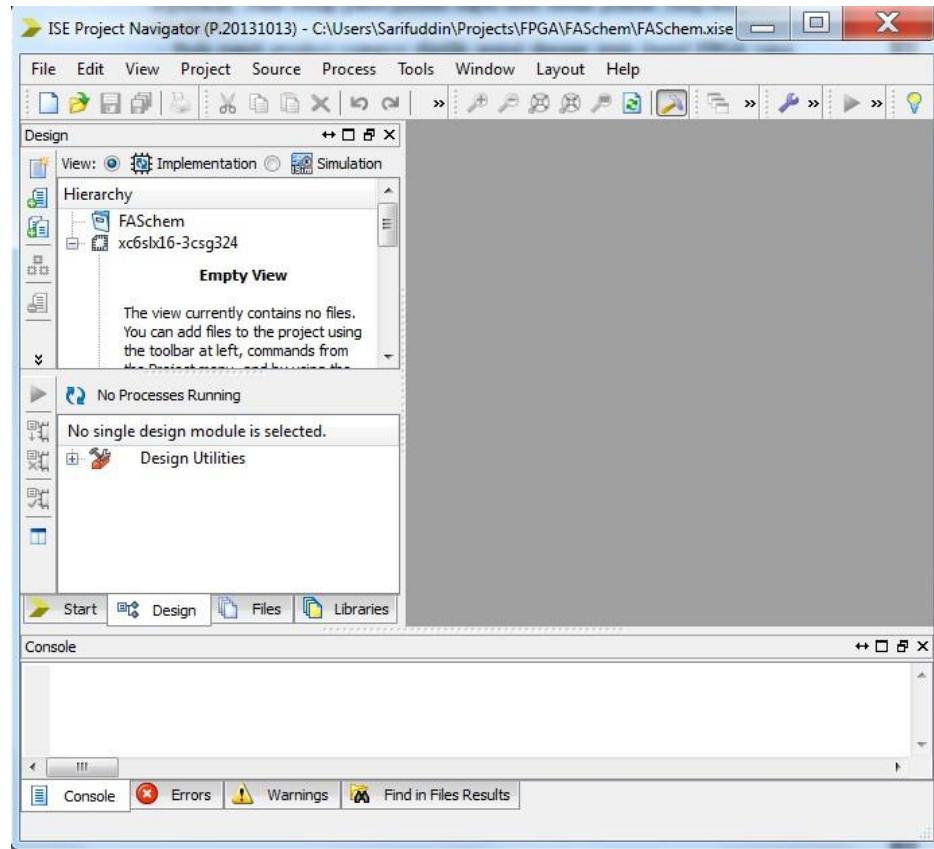
Gambar 3.3. Layar interface Project Settings untuk memilih spesifikasi project.

3. Pada layar Project Settings, ini dilakukan pemilihan spesifikasi perangkat lunak, synthesis, simulator dan perangkat keras (jenis board FPGA) yang akan digunakan. Pada setiap panel pilihan dapat diisi sesuai pilihan yang dinginkan, misalnya:
 - Pada panel "Product Category" dipilih sesuai dengan jenis board FPGA yang digunakan, pada contoh ini pilih "General Purpose". Kemudian, pada panel "Family" pilih "Spartan6", pada panel "Device" pilih "XC6SLX16", lalu pada panel "Package" pilih "FG320" dan pada panel "Speed" pilih "-3".
 - Pada bagian "Top-level source type": pada panel "Synthesis tool" pilih XST (VHDL/Verilog), pada panel "Simulator" pilih ISim (VHDL/Verilog), pada panel "Preferred language" pilih "Verilog", pada panel "Property specification in project file" pilih "Store all values", dan pada panel "VHDL source analysis standard" pilih "VHDL-93". Setelah selesai tekan tombol "Next", maka akan muncul layar Project Summary seperti pada gambar 3.4. Selanjutnya click tombol Finish untuk menyelesaikan pembuatan file Project baru, dan kembali pada layar ISE Project Navigator dengan tampilan seperti pada gambar 3.5.



Gambar 3.4. Layar interface resume sepesifikasi project sudah dibuat..

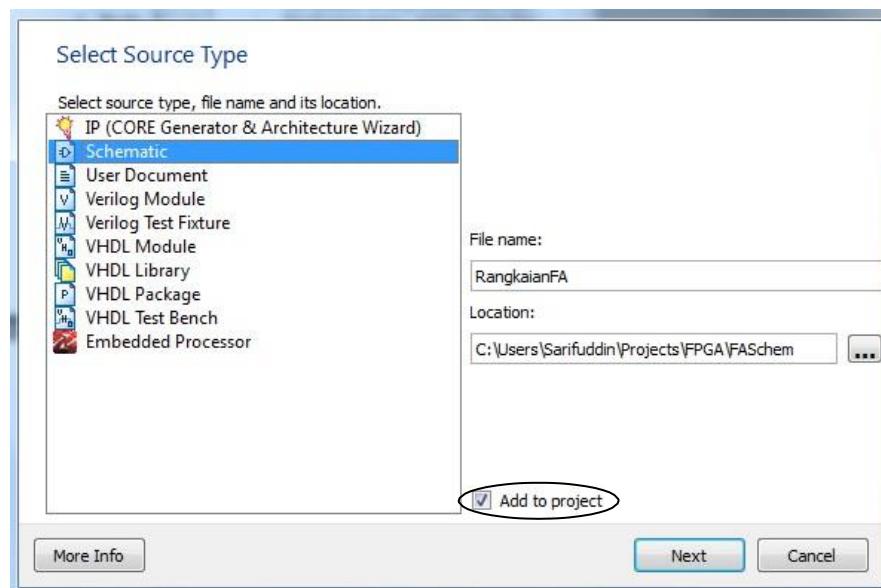
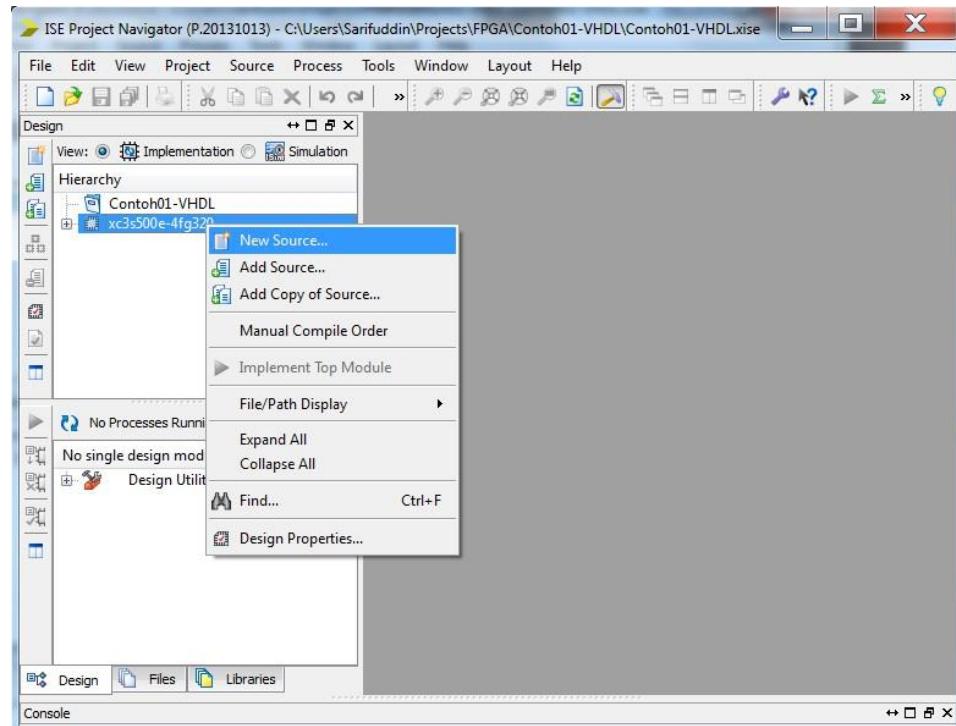
4. Pada layar interface ISE Project Navigator terdapat tiga panel yaitu Design, Console dan HDL Editor. Pada panel pertama Design terbagi dua yaitu jendela View dan jendela Process. Jendela View (1a) menampilkan semua source files yang dibuat dalam project yang sedang dijalankan, dimana setiap file bisa berisi komponen dasar atau rangkaian kompleks yang menggabungkan beberapa komponen dalam beberapa file pada project tersebut. Pada jendela View ini terdapat dua pilihan yaitu Implementation dan Simulation. Jendela Process (1b) berfungsi menampilkan semua jenis proses yang dapat dijalankan oleh setiap file sesuai yang dipilih pada jendela View. Panel Console (2) menampilkan informasi error atau kesalahan pada coding atau kesalahan proses dan warning. Panel HDL editor (3) akan menampilkan ruang edit rangkaian atau menampilkan source code dari file yang dipilih atau informasi lainnya seperti Design Summary dan lainnya.

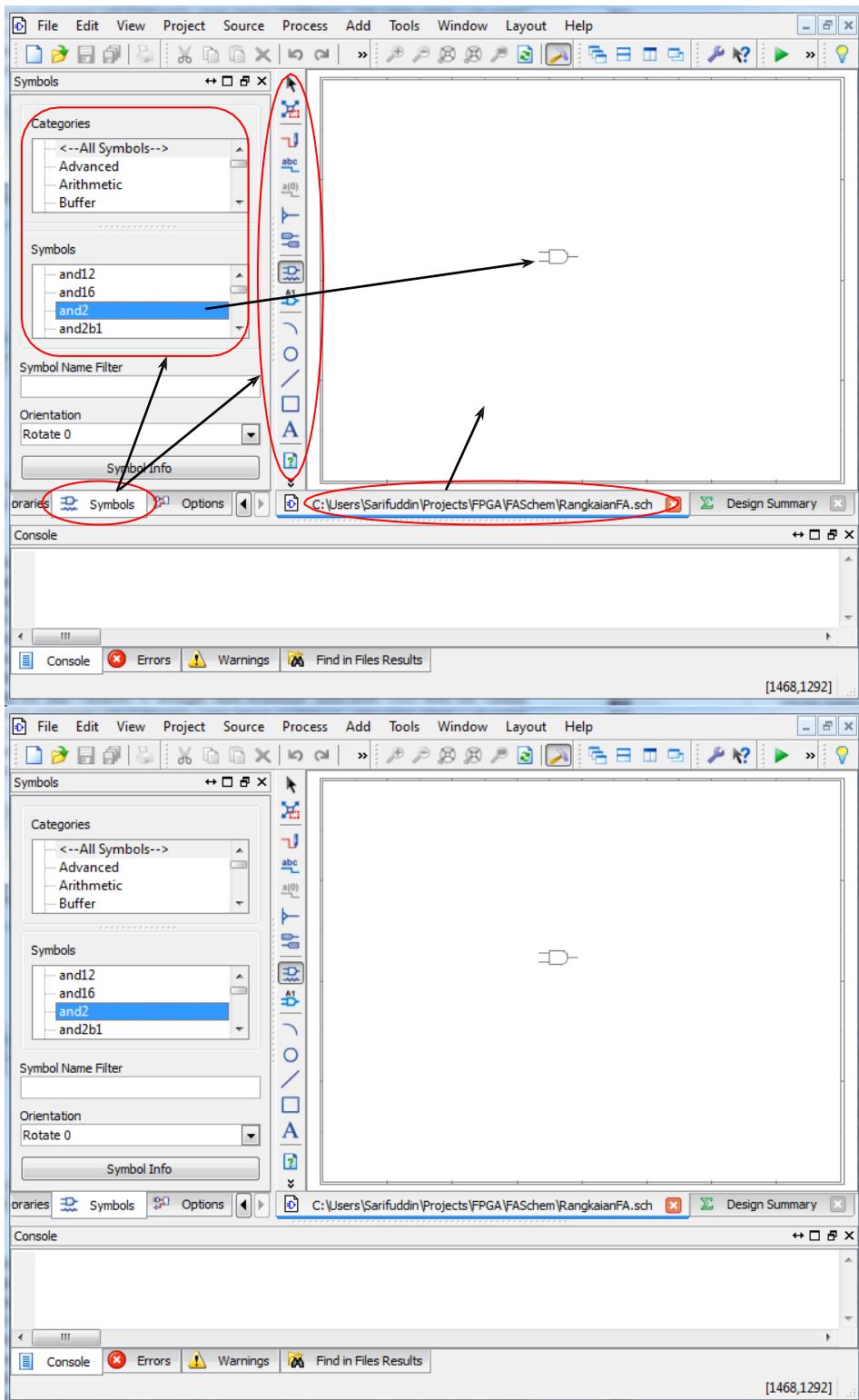


3.1.2 Disain Rangkaian Schematic

Setelah nama project serta spesifikasi device sudah didefinisikan, langkah berikutnya adalah menambahkan file schematic untuk disain rangkaian kedalam project.

1. Click-kiri pada project sehingga muncul pilihan seperti pada gambar 3.6. Ada tiga pilihan yaitu membuat file baru (New source), mengambil file yang sudah ada (Add source) atau mengkopi coding dari sumber file yang sudah ada dan membuat file baru (Add copy of source). Selain terdapat dalam menu "Project", tiga pilihan tersebut juga ada pada bagian kiri jendela View (lihat anak panah) yang dapat dipilih langsung melalui menu icon tersebut. Selanjutnya pilih New Source, maka akan tampil layar Select source type seperti pada gambar 3.7.
2. Untuk membuat file schematic pilih menu "Schematic", lalu pada panel "File name" isi nama file sesuai dengan yang dinginkan, misalnya RangkaianFA. Pastikan checklist "Add to project" (lingkaran berwarna hitam) dipilih, lalu click tombol "Next", maka akan tampil layar Define Module seperti pada gambar 3.8.



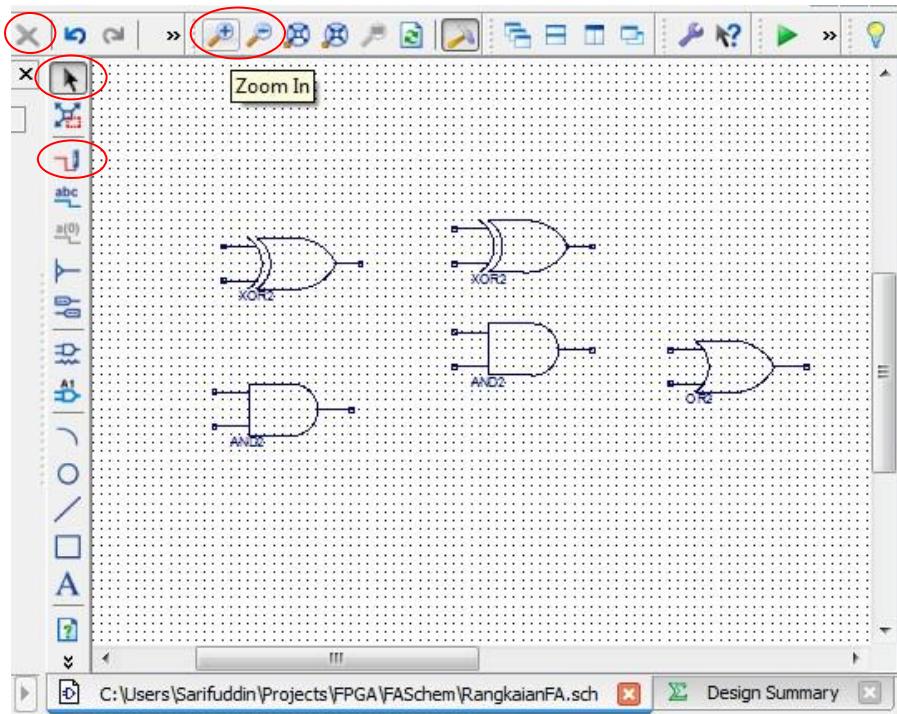


komponen atau jalur hubungan antar komponen, klik pada komponen/jalur tersebut (akan bertanda merah) lalu pilih tombol menu bersimbol kali " " atau gunakan tombol Delete pada keyboard. Operasi juga dapat dilakukan pada setiap komponen dengan cara klik kiri pada komponen yang akan diedit lalu pilih menu sesuai operasi yang diinginkan. Kemudian bila ingin memperbesar/memperkecil tampilan layar editor, gunakan simbol zoomin/zoomout " ".

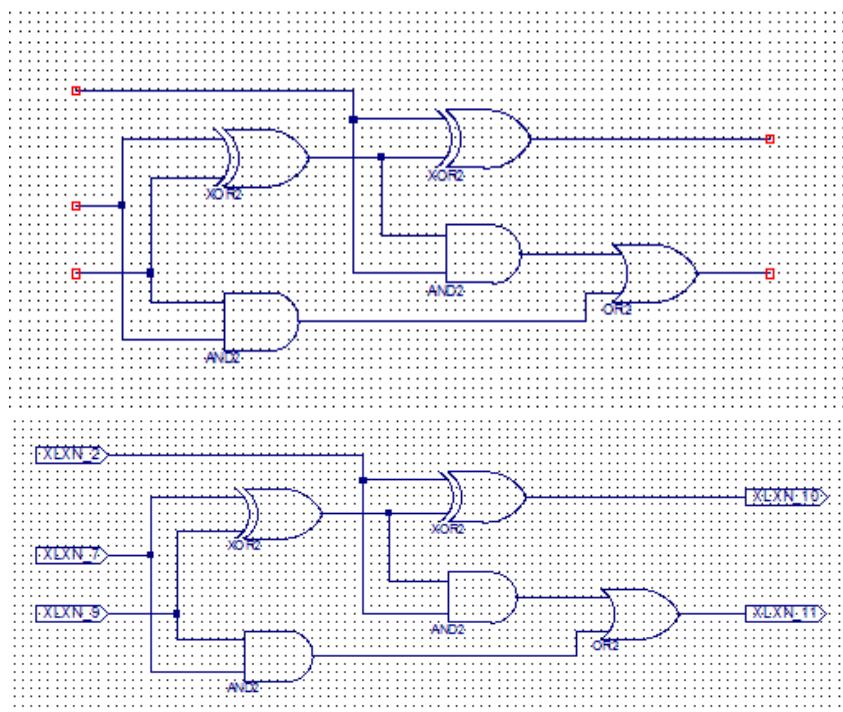
4. Tahap selanjutnya adalah menghubungkan jalur masukan, keluaran dan keterhubungan

antar komponen. Klik tombol menu Add wire " ", lalu tambahkan jalur masukkan, anatar komponen atau keluaran hingga semua terhubung seperti pada gambar 3.10-a.

5. Tahap terakhir pembuatan rangkaian adalah menambahkan Pin I/O. Pilih tombol menu dengan simbol " ", lalu pada panel sebelah kiri, pilih "Add an automatic marker". Selanjutnya, pada titik jalur masukkan, klik kanan sambil menarik sedikit ke kiri lalu lepas, maka secara otomatis akan ditambahkan Pin Input. Kemudian pada titik jalur keluaran, klik kanan sambil menarik sedikit ke kanan lalu lepas. Lakukan hal yang sama untuk jalur masukkan dan keluaran yang lainnya. Setelah penambahan Pin I/O, maka rangkaian Full Adder (FA) akan terlihat seperti pada gambar 3.10-b.



Gambar 3.9. Layar editor schematic disain rangkaian.



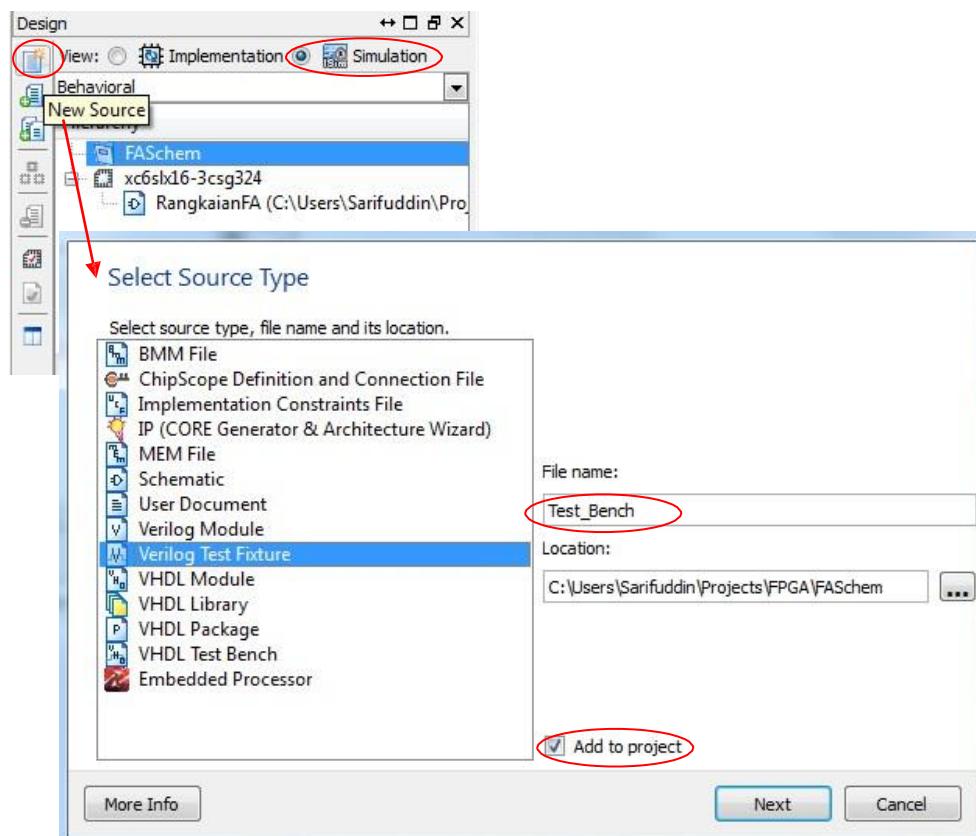
Gambar 3.10. Disain schematic rangkaian Full-Adder.

3.1.3 Simulasi Rangkaian Pada Level Disain Schematic

Setelah selesai proses pembuatan skema rangkaian FA, maka masuk pada tahap simulasi untuk mensimulasikan fungsi logika kerja dari rangkaian tersebut. Untuk melakukannya, kembali lihat gambar 3.5 yang diberi tanda lingkaran ada pilihan "Implementation" dan "Simulation".

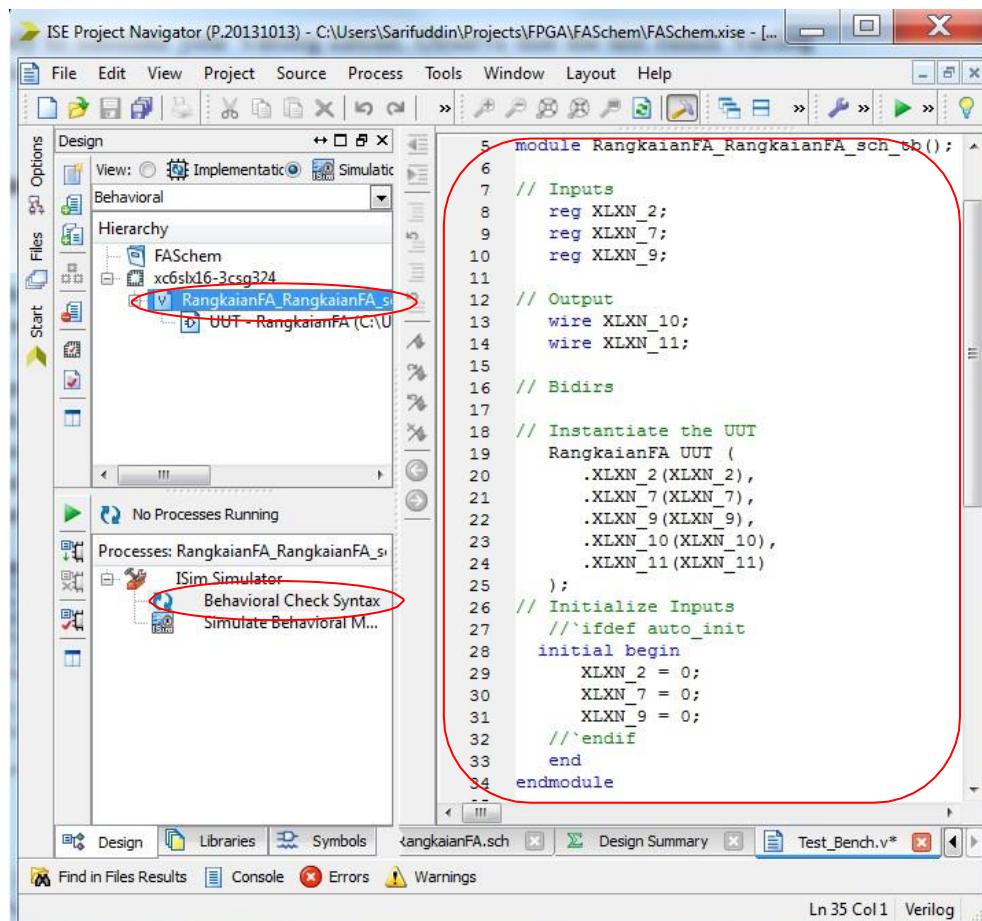
1. Pada tampilan tersebut, klik pada pilihan "Simulation". Kemudian pada panel Hierarchy, pilih nama project yang akan dijalankan yaitu FASchem. Lalu pilih menu "New source" tombol simbol "", maka akan tampak layar "Select Source Type" seperti pada gambar 3.11. Sebagai catatan bahwa saat membuat project "Preferred Language" yang dipilih adalah Verilog, maka untuk simulasi pilih "Verilog Test Fixture". Tulis nama file untuk menyimpan source code simulasi Verilog, misalnya Test_Bench. Kemudian klik "Next", lalu pilih nama file rangkaian yang akan disimulasikan. Pilih nama file yang telah dibuat di atas yaitu "RangkaianFA" lalu "Next", maka akan tampil layar seperti pada gambar 3.12. Pada bagian kanan akan tampil code simulasi yang terdiri dari empat blok: pertama adalah Input (ada 3 jalur masukkan), kedua adalah Output (ada 2 jalur keluaran), ketiga adalah modul rangkaian "Unit UnderTest" (UUT) dan terakhir adalah bagian Initialize Inputs untuk inisialisasi dan settingan variasi kombinasi nilai logika masukkan. Pada bagian terakhir ini di awal semua masukkan diset ke nilai 0 (nol) dan dalam bagian inilah yang harus diset nilai-nilai kombinasi masukkannya sesuai dengan yang diinginkan untuk proses test.

Misalnya seperti yang diperlihatkan pada gambar 3.13. Setiap perubahan kombinasi masukkan diberi jedah waktu clock selama 50 ticks.



Gambar 3.11. Penambahan file simulasi Verilog.

2. Proses verifikasi sintaks, pada panel "Hierarchy" pilih file "Test_Bench.v" (lihat lingkaran pada sebelah kiri atas gambar 3.12). Lalu jalankan proses verifikasi sintaks pada coding dengan cara mengklik dua kali menu "Behavioral Check Syntax" pada panel Processes: "ISim Simulator". Selama proses verifikasi akan ditampilkan informasi proses yang sedang dijalankan pada panel "Console". Jika informasi "Process 'Behavioral Check Syntax' completed successfully" berarti berhasil tanpa error.
3. Jalankan simulasi dengan cara klik dua kali menu "Simulate Behavioral Model" pada panel Processes: "ISim Simulator". Setelah proses simulasi selesai maka tampil interface ISim yang menampilkan sinyal hasil simulasi seperti pada bagian kanan atas dari gambar 3.14. Sinyal pertama adalah keluaran S0, sinyal kedua adalah keluaran Carry-out C1 dan tiga sinyal terakhir adalah masukkan A0, B0, dan C0.



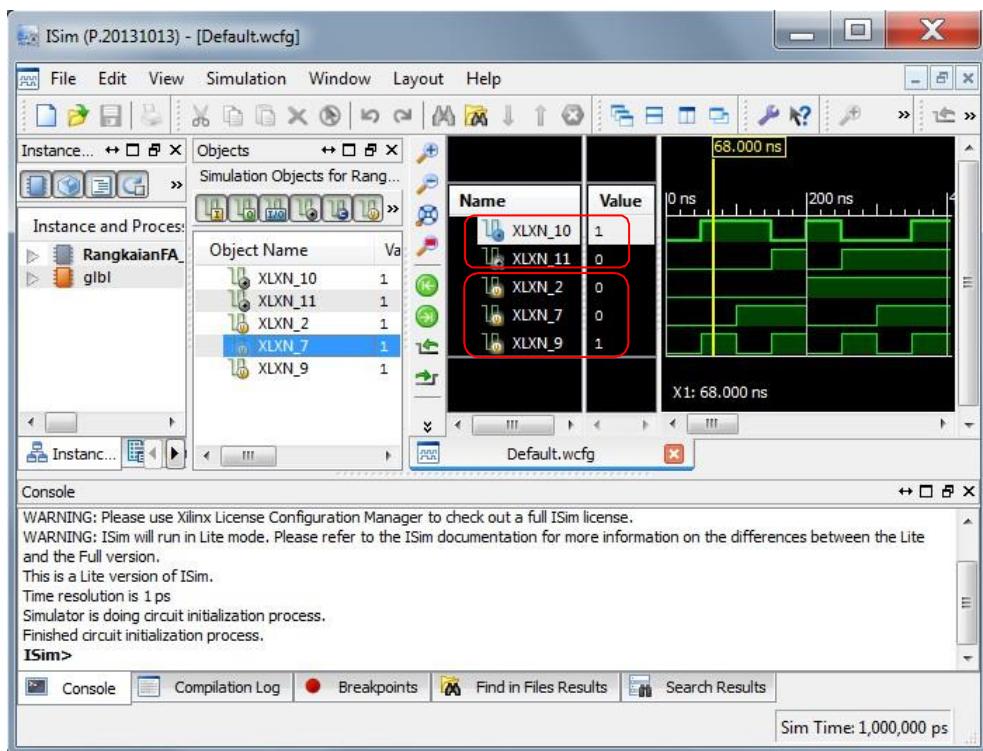
Gambar 3.12. Interface proses simulasi.

```

26 // Initialize Inputs
27 //`ifndef auto_init
28 initial begin
29     XLXN_2 = 0;
30     XLXN_7 = 0;
31     XLXN_9 = 0;
32 //`endif
33
34 #50;
35 XLXN_2 = 0; XLXN_7 = 0; XLXN_9 = 1; //001
36 #50;
37 XLXN_2 = 0; XLXN_7 = 1; XLXN_9 = 0; //010
38 #50;
39 XLXN_2 = 0; XLXN_7 = 1; XLXN_9 = 1; //011
40 #50;
41 XLXN_2 = 1; XLXN_7 = 0; XLXN_9 = 0; //100
42 #50;
43 XLXN_2 = 1; XLXN_7 = 0; XLXN_9 = 1; //101
44 #50;
45 XLXN_2 = 1; XLXN_7 = 1; XLXN_9 = 0; //110
46 #50;
47 XLXN_2 = 1; XLXN_7 = 1; XLXN_9 = 1; //111
48 #50;
49 end
50 endmodule
51

```

Gambar 3.13. Penambahan code simulasi variasi nilai variabel sinyal masukkan



Gambar. 3.14. Interface ISim, hasil simulasi disain schematic rangkaian.

Catatan: selain model coding variasi nilai variabel masukkan yang telah digunakan di atas, ada dua model lainnya seperti yang ditunjukkan pada gambar 3.5. Lakukan modifikasi code simulasi pada gambar 3.13 menjadi seperti pada gambar 3.15-a atau 3.15-b. Lakukan kembali tahap ke-2 dan ke-3, bagaimana hasilnya?

```

integer i, j, k;
initial begin
    XLXN_2 = 0; XLXN_7 = 0; XLXN_9 = 0;
#50;
    for (i = 0; i < 2; i = i + 1)
        for (j = 0; j < 2; j = j + 1)
            for (k = 0; k < 2; k = k + 1)
                begin
                    XLXN_2 = i;
                    XLXN_7 = j;
                    XLXN_9 = k;
#25;
                end
            end
        endmodule

```

(a)

```

integer i;
initial begin
    XLXN_2 = 0; XLXN_7 = 0; XLXN_9 = 0;
#50;
    for (i = 0; i < 8; i = i + 1)
        begin
            {XLXN_2, XLXN_7, XLXN_9} = i;
#25;
        end
    end
endmodule

```

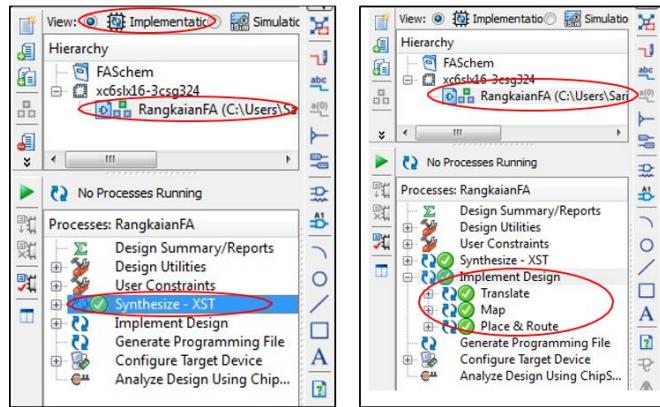
(b)

Gambar 3.15. Dua contoh coding cara lain mengkombinasikan nilai masukkan.

3.1.4 Sintesis dan Implementasi Schematic Rangkaian

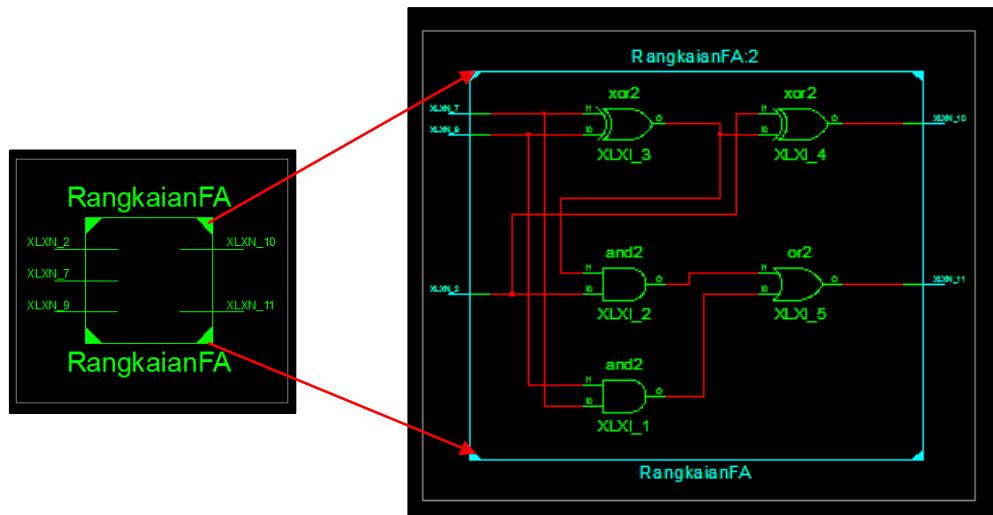
Sebelum tahap sintesis (Synthesize) ada tahap Constraints yang harus dilakukan bila rangkaian akan diimplementasikan menggunakan FPGA Board tertentu. Tahap ini bertujuan untuk mendefinisikan Pin-Pin I/O yang digunakan sesuai dengan spesifikasi dari Pin-Pin IC FPGA pada Board tersebut. Untuk sementara, tahap ini dapat dilewatkan.

1. Tahap sintesis adalah proses translasi dari level deskripsi disain schematic ke level struktural (sama halnya seperti meng-compile coding bahasa C ke bahasa assembler). Untuk menjalankan proses sintesis : kembali ke interface ISE Project Navigator gambar 3.12. Pada panel "View" klik kembali pada pilihan menu "Implementation". Kemudian pada panel "Hierarchy" pilih file schematic yang akan dijalankan "RangkaianFA.sch". Pada panel "Processes", klik kanan dua kali pada menu "Synthesize-XST" dan tunggu hingga tanda checklist berwarna hijau muncul seperti yang terlihat pada gambar 3.16-a. Pada panel Console akan muncul pesan "Process "Synthesize - XST" completed successfully" yang menunjukkan bahwa proses sintesis telah berhasil.
2. Tahap implementasi (Implement) bertujuan untuk mendefinisikan konfigurasi perangkat keras. Ada tiga proses yang dilakukan yaitu Translate, Mapping dan PAR (Place and Route). Pada panel "Processes", klik kanan dua kali pada menu "Implement Disain" dan tunggu hingga tanda checklist berwarna hijau muncul seperti yang terlihat pada gambar 3.16-b. Pada panel Console akan muncul pesan " Process "Generate Post- Place & Route Static Timing" completed successfully". Selanjutnya untuk melihat berapa jumlah sumber daya yang akan digunakan dalam IC FPGA, klik "Desing Summery (Implemented)" pada panel sebelah kanan bawah. Telihat pada tabel resume tersebut, setelah proses palce & route, rangkaian FA yang dibuat menempati atau menggunakan 1 slice dan 5 pin IOB.

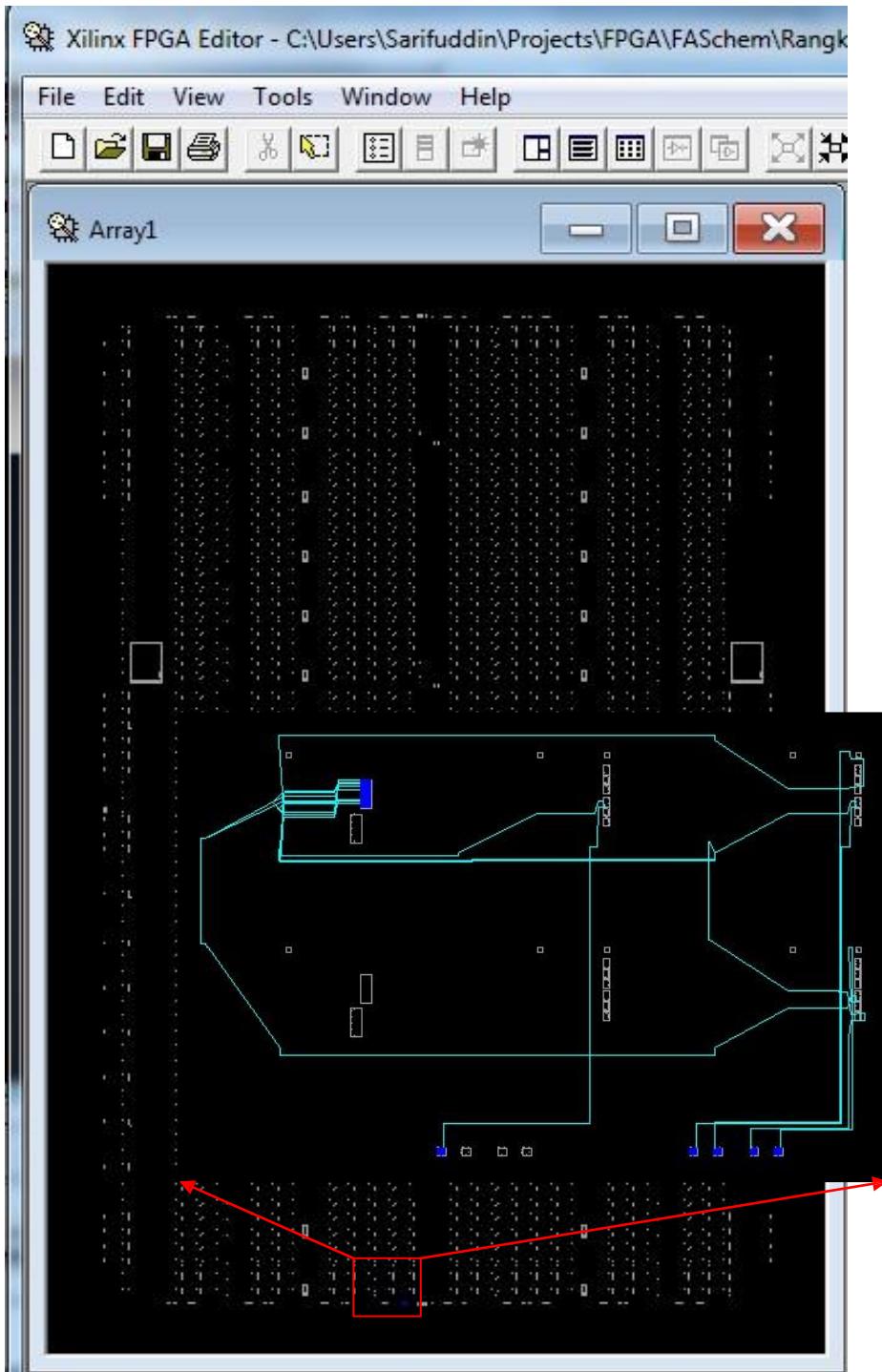


Gambar 3. 16. (a) proses Synthsize, (b) proses Implement

3. Untuk melihat blok rangkaian yang telah terbentuk setelah proses Synthsize dan Implement, klik dua kali menu "View RTL Schematic" yang terletak dalam menu "Synthesize-XST" maka akan muncul blok "RangkaianAF" seperti pada gambar 3.17-a. Untuk melihat rangkaian FA yang terdapat dalam blok tersebut: klik kiri pada blok dan pada panel menu yang muncul pilih "Show block Contents" maka akan terlihat RangkaianFA. Rangkaian tersebut sesuai dengan yang telah dibuat pada tahap disain schematic. Kemudian, untuk kembali ke bentuk blok, klik kiri pada rangkaian dan pilih "Hide block Contents".



4. Selanjutnya, untuk melihat hasil implementasi rangkaian di dalam IC FPGA, klik dua kali pada menu "View/Edit Routed Design (FPGA Editor)" yang terletak dalam menu "Place & Route", maka akan tampil interface Xilinx FPGA Editor dimana berisi blok-blok sumberdaya yang ada dalam IC FPGA seperti pada gambar 3.18 (sesuai dengan tipe yang dipilih diawal pembuatan file rangkaian). Blok Slice dan I/O yang telah digunakan berwarna biru dan yang masih kosong berwarna putih. Untuk RangkaianFA yang dibuat terletak pada bagian paling bawah (dalam kotak merah). Pilih menu symbol dan pilih area RangkaianFA yang akan dizoom lalu tekan menu simbol ZoomIn, maka akan tampak pembesaran dan dapat terlihat Slice dan I/O mana saja yang digunakan serta jalur-jalur kabel penghubungnya. Pada interface ini dapat dilakukan modifikasi (mengedit) posisi Sclice dan/atau posisi Pin I/O serta jalur-jalur penghubung antar Slice dan I/O. Untuk melakukan hal tersebut dan mengetahui lebih dalam tentang Xilinx FPGA Editor, silahkan pelajari lebih mendalam dokumentasi yang tersedia melalui menu "Help"-nya.



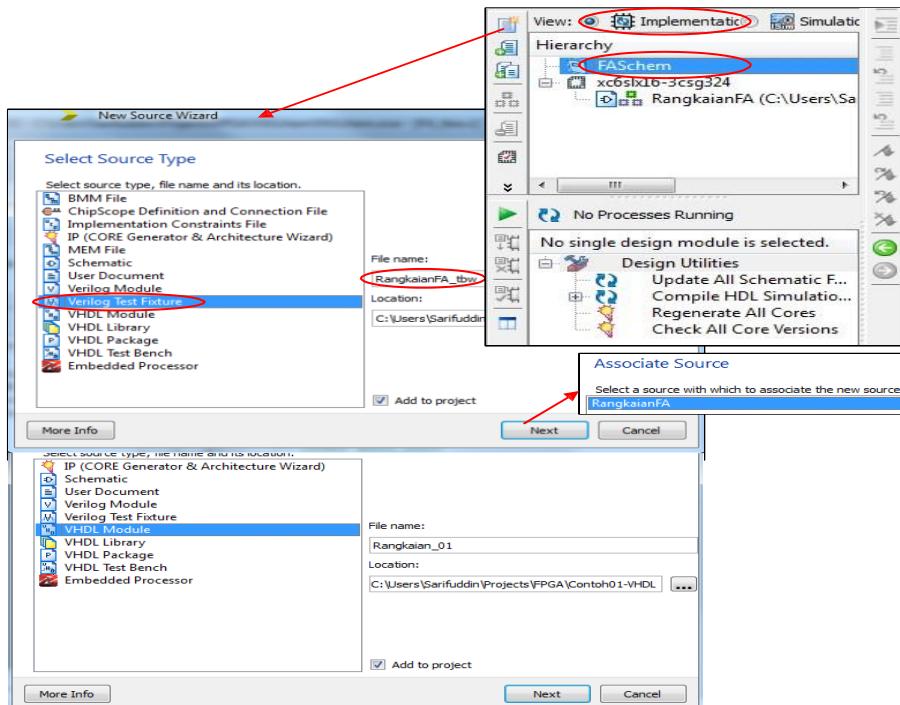
Gambar 3.18. Bentuk implementasi rangkaian FA pada level Place & Route.

3.1.5. Simulasi Pada level Place & Route

Setelah proses Route perlu dilakukan kembali simulasi sinyal untuk memastikan bahwa rangkaian yang telah diimplementasi hingga pada level Route tetap berfungsi sesuai dengan disain semula. Berikut tahapan proses simulasi:

1. Dari interface ISE Project Navigator, pada panel "View" pilih "Implementation" dan pada panel "Hierarchy" pilih project "FAShem", lalu pilih "New Source" (lihat gambar 3.19-a), maka akan tampak jendela baru seperti pada gambar 3.19-b. Buat file program Verilog untuk simulasi level Route, misalnya pada panel File name tulis nama file

RangkaianFA_tbw lalu pilih menu "Verilog Test Fixture" dan klik "Next", akan muncul jendela baru gambar 3.19-c. Pada jendela tersebut akan muncul pilihan rangkaian yang akan digabungkan dan disimulasikan melalui file RangkaianFA_tbw.v. Pilih RangkaianFA lalu tekan menu "Next". Kembali ke interface ISE Project Navigator dan pada layar sebelah kanan akan secara otomatis muncul coding Verilog dari file RangkaianFA_tbw.v.



Gambar 3.19. Pembuatan file simulasi level Route

2. Coding file simulasi level Route identik dengan coding simulasi leavel desain Schematic. Bagian blok "Initialize Input" harus ditambahkan misalnya seperti yang diberi tanda lingkaran (lihat gambar 3.20).
3. Pada panel View sebelah kiri pilih: "Simulation", "Post-Route" dan file RangkaianFA.tbw.v, seperti yang dilingkari pada gambar 3.20. Selanjutnya pada panel Processes: ISim Simulator, klik dua kali pada "Post-Place & Route Check Syntax" hingga tanda checklist hijau muncul. Kemudian dua klik pada panel Console untuk memastikan muncul "Process 'Simulate Post-Place & Route Model' completed successfully", yang menunjukkan codingnya sudah benar. Bila terdapat error, betulkan kembali codingnya sesuai dengan posisi yang ditunjuk, kemudian jalankan kembali "Post-Place & Route Check Syntax". Selanjutnya, klik dua kali pada "Simulate Post- Place & Route Model". Bila proses simulasinya sukses maka akan muncul interface ISim yang menampilkan sinyal masukkan dan keluaran hasil proses simulasi seperti yang diperlihatkan pada gambar 3.21.

The screenshot shows the ISE Project Navigator interface. In the top left, under 'Design' view, 'Implementation' is selected. Below it, 'Post-Route' is highlighted. In the center, the 'Processes' panel shows 'RangkaianFA_RangkaianFA_sch_tb' expanded, with 'Isim Simulator' selected. Underneath, 'Post-Place & Route Check Syntax' and 'Simulate Post-Place & Route Model' are listed. A red circle highlights the 'Isim Simulator' entry. On the right, the Verilog code for 'RangkaianFA_timesim.v' is displayed:

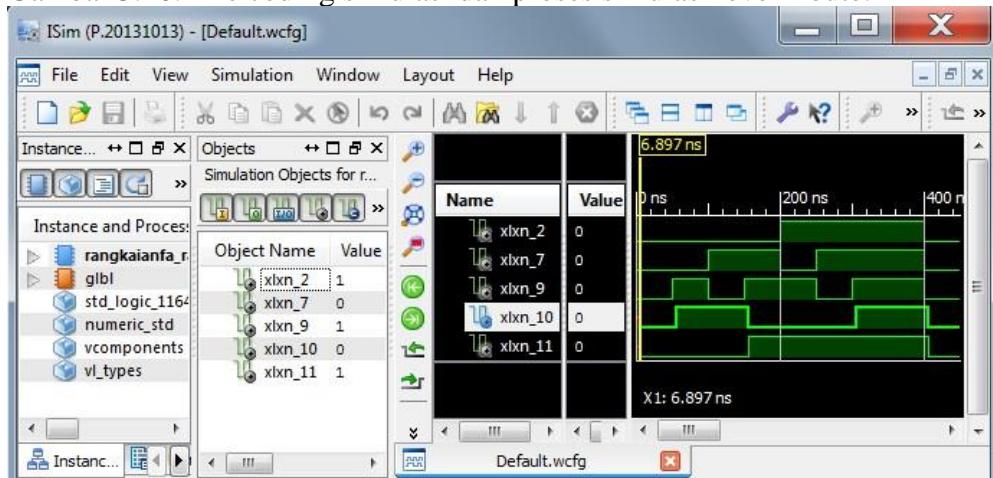
```

1 // Verilog test fixture created from schematic
2
3 `timescale 1ns / 1ps
4
5 module RangkaianFA_RangkaianFA_sch_tb();
6
7 // Inputs
8 reg XLXN_2;
9 reg XLXN_7;
10 reg XLXN_9;
11
12 // Output
13 wire XLXN_10;
14 wire XLXN_11;
15
16 // Bidirs
17
18 // Instantiate the UUT
19 RangkaianFA_UUT (
20     .XLXN_2 (XLXN_2),
21     .XLXN_7 (XLXN_7),
22     .XLXN_9 (XLXN_9),
23     .XLXN_10 (XLXN_10),
24     .XLXN_11 (XLXN_11)
25 );
26
27 // Initialize Inputs
28 integer i;
29 initial begin
30     XLXN_2 = 0; XLXN_7 = 0; XLXN_9 = 0;
31     #50;
32     for (i = 0; i < 8; i = i + 1)
33         begin
34             { XLXN_2, XLXN_7, XLXN_9 } = i;
35             #25;
36         end
37     end
38 endmodule

```

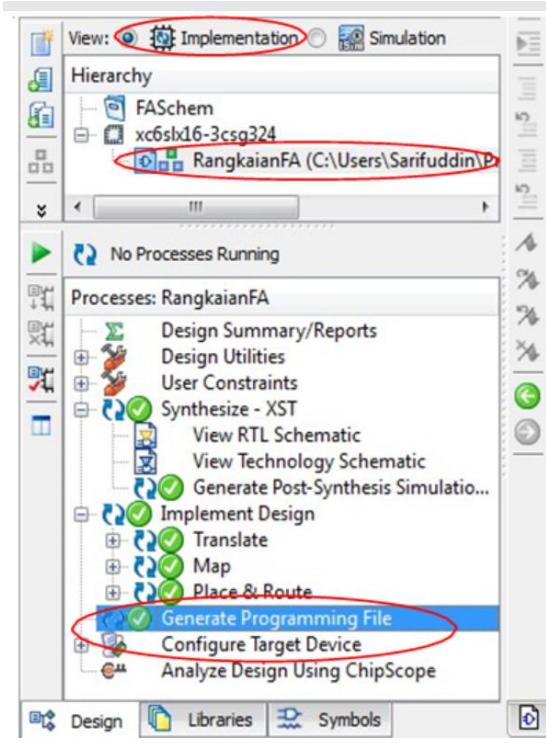
A red box highlights the initialization loop from line 29 to line 37. At the bottom, the status bar shows 'Ln 26 Col 21 Verilog'.

Gambar 3.20. File coding simulasi dan proses simulasi level Route.



Gambar 3.21. Sinyal hasil simulasi level Route.

4. Tahap selanjutnya adalah proses kompilasi oleh Bitstream Generator (BitGen) untuk membangkitkan Bit file. Pada ISE Navigator dan panel "View" pilih "Implementation" dan file RangkaianFA.sch (lihat gambar 2.22). Kemudian pada panel "Processes" klik dua kali pada "GenerateProgramming File", tunggu hingga tanda checklist hijau muncul dan pastikan pada panel "Console" muncul "Process "Generate Programming File" completed successfully".



Gambar 2.22. Proses Pembangkitan Bit file.

Tahap terakhir adalah proses transfer Bit file ke FPGA Board. Sebelum melanjutkan ke tahap ini pastikan:

1. Pastikan bahwa proses "Synthesize", "Implement" dan "GenerateProgramming File" dilakukan secara berurutan karena proses-proses tersebut bergantung satu terhadap yang lainnya.
2. Pastikan hubungan kabel USB antara PC yang digunakan dan FPGA Board tersambung, power listrik menyala dan pastikan semua koneksi ke board berfungsi dengan baik.

Catatan: guide komponen elektronika dan karakteristiknya untuk kebutuhan disain schematic dapat di download pada alamat berikut:

https://gab.wallawalla.edu/~larry.aamodt/engr433/xilinx_14/ug616_spartan6_lib_guide_for_schem.pdf

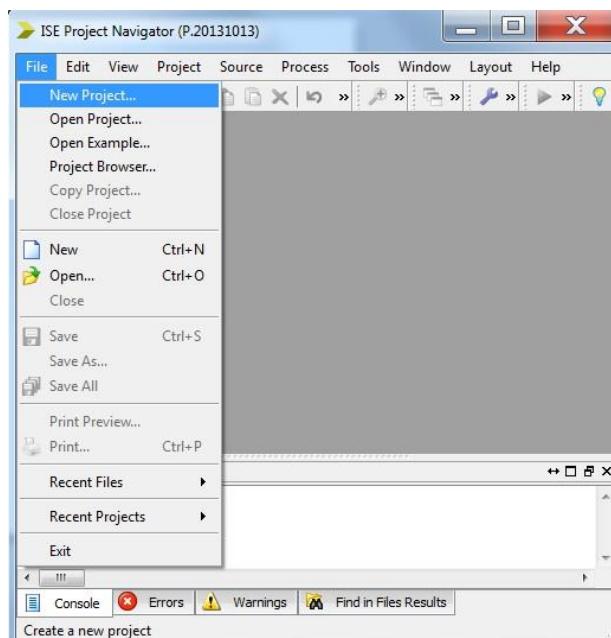
JOBSHEET 11

PEMROGRAMAN VHDL MENGGUNAKAN ISE DESIGN

1. Membuat Project VHDL

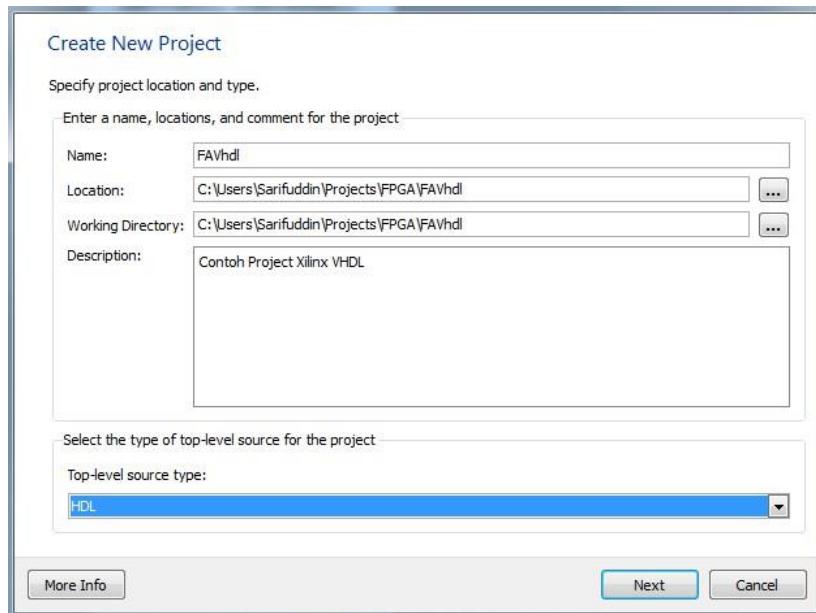
Sebelum memulai membuat sebuah projek pastikan terlebih dulu fungsi logika dari rangkaian yang ingin dibuat. Misal dalam tutorial akan diimplementasi rangkaian Full Adder yang memiliki fungsi logika: $S_0 = A_0 \oplus B_0 \oplus C_0$ dan $C_1 = C_0(A_1 \oplus B_1) + A_1B_1$. Fungsi ini memiliki tiga variabel masukkan (A_0, B_0, C_0) dan dua variabel keluaran (S_0, C_1). Setelah itu menjalankan perangkat lunak ISE Design Suite 14.7. Untuk membuat sebuah Projek (rangkaian) baru ikuti tahapan berikut:

1. Pada layar Project Navigator pilih/click menu File → New Project (seperti pada gambar 3.23). Setelah menu New Project dipilih maka akan terbuka jendela baru Creat New Project seperti yang terlihat pada gambar 3.24.

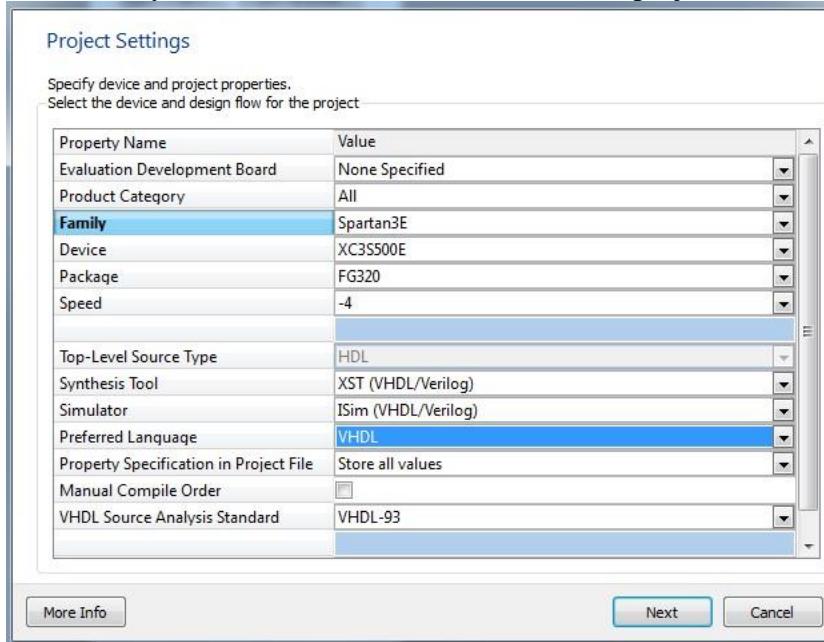


Gambar 3.23. Layar interface ISE Project Navigator

2. Pada jendela ini tulis nama project pada panel Name: tulis nama project misalnya FAVhdl. Saat pemberian nama project tersebut, pada panel Location dan Working Directory akan ditambahkan secara otomatis ke lokasi dan ke directory mana project tersebut akan disimpan. Bila ingin mengganti nama Location dan Working Directory, click tombol browse (...) sebelah kanan panel, kemudian pilih sesuai dengan lokasi dan directory yang diinginkan. Pada panel ke-4 terdapat bagian Description, dapat ditambahkan teks penjelasan dari project yang dibuat (boleh juga dikosongkan). Pada panel bagian bawah terdapat pilihan top-level source type, pilih "HDL" lalu tekan tombol "Next", maka akan tampil layar baru Project Settings seperti pada gambar 3.25.



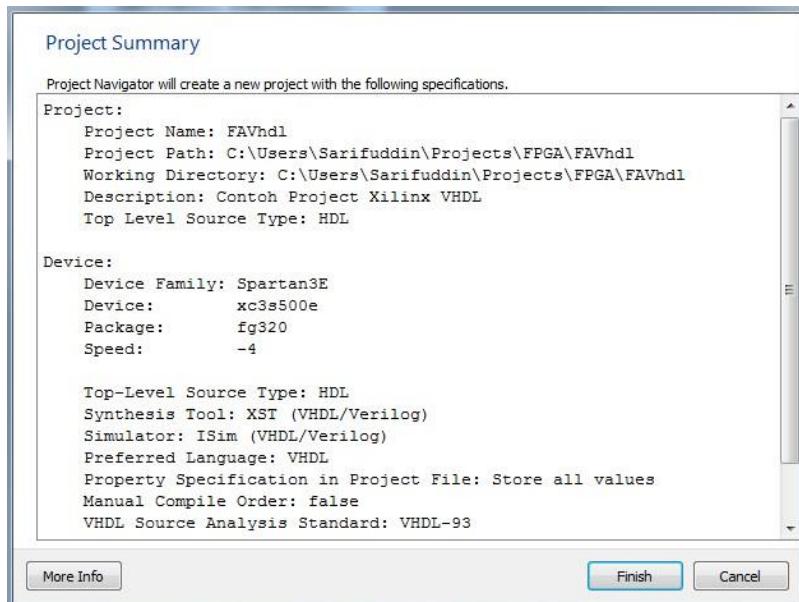
Gambar 3.24. Layar interface untuk menuliskan nama project dan lokasinya.



Gambar 3.25. Layar interface Project Settings untuk memilih spesifikasi project.

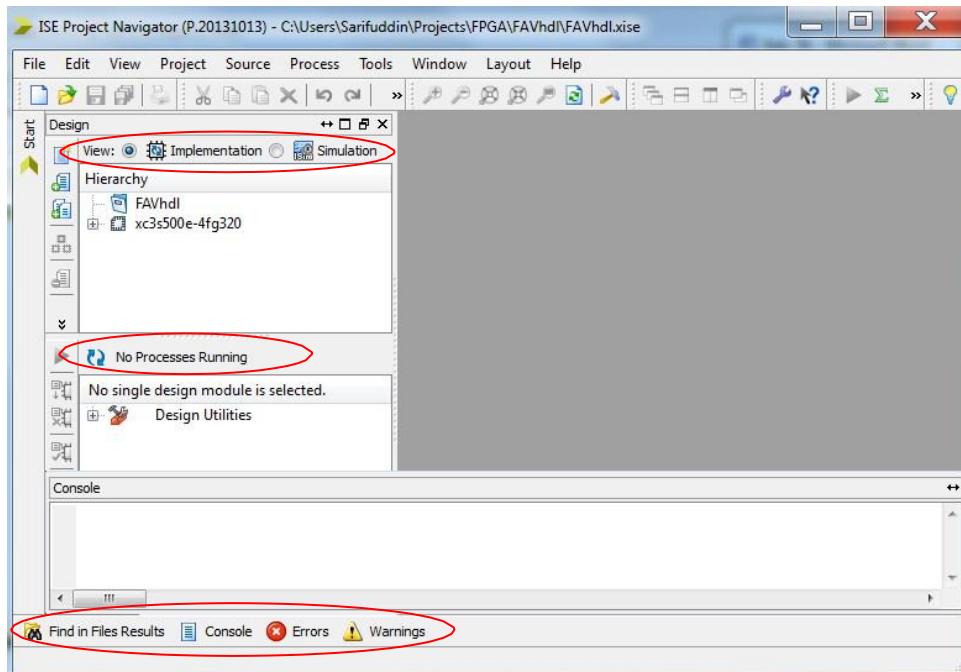
3. Pada layar Project Settings ini, lakukan pemilihan spesifikasi perangkat lunak, synthesis, simulator dan perangkat keras (jenis FPGA) yang akan digunakan. Pada setiap panel pilihan dapat diisi sesuai pilihan yang dinginkan, misalnya:
 - Pada panel "Product Category" pilih "All" (bila Evaluation Development board belum diketahui), pada panel "Family" pilih "Spartan3E", pada panel "Device" pilih "XC3S500E", pada panel "Package" pilih "FG320" dan pada panel "Speed" pilih "-4".
 - Pada bagian Top-level source type, karena yang akan digunakan adalah VHDL maka dipilih: pada panel "Synthesis Tool" pilih "XST (VHDL/Verilog)", pada panel "Simulator" pilih "ISim (VHDL/Verilog)", pada panel "Preferred Language" pilih VHDL, pada panel "Property Specification in Project File" pilih "Store all values", dan pada panel "VHDL Source Analysis Standard" pilih "VHDL-93". Setelah selesai tekan tombol "Next", maka akan muncul layar Project Summary seperti pada gambar 3.26. Click tombol "Finish" untuk menyelesaikan pembuatan file Project baru, dan

kembali pada layar ISE Project Navigator dengan tampilan seperti pada gambar 3.27.



Gambar 3.26. Layar interface resume sepesifikasi project sudah dibuat..

4. Pada layar interface ISE Project Navigator terdapat tiga panel yaitu Design, Console dan HDL Editor. Panel Design terbagi dua yaitu jendela View dan jendela Processes. Jendela View (1a) menampilkan semua source files yang dibuat dalam project yang sedang dijalankan, dimana setiap file bisa berisi komponen dasar atau rangkaian kompleks yang menggabungkan beberapa komponen dalam beberapa file pada project tersebut serta file simulasi. Pada jendela View ini terdapat dua pilihan yaitu Implementation dan Simulation. Pada Panel View ini klik pada checklist Implementation, dibawahnya akan ada panel Hierarchy yang berisikan project FAVhdl dan komponen atau device FPGA yang telah dipilih pada tahap sebelumnya. Jendela Processes (1b) berfungsi menampilkan semua jenis proses yang dapat dijalankan oleh setiap file sesuai yang dipilih pada jendela View. Panel Console (2) menampilkan informasi proses-proses yang dijangan, suksesnya proses, error bila terdapat kesalahan pada coding atau kesalahan proses dan warning. Panel HDL editor (3) menampilkan source code dari file yang dipilih.

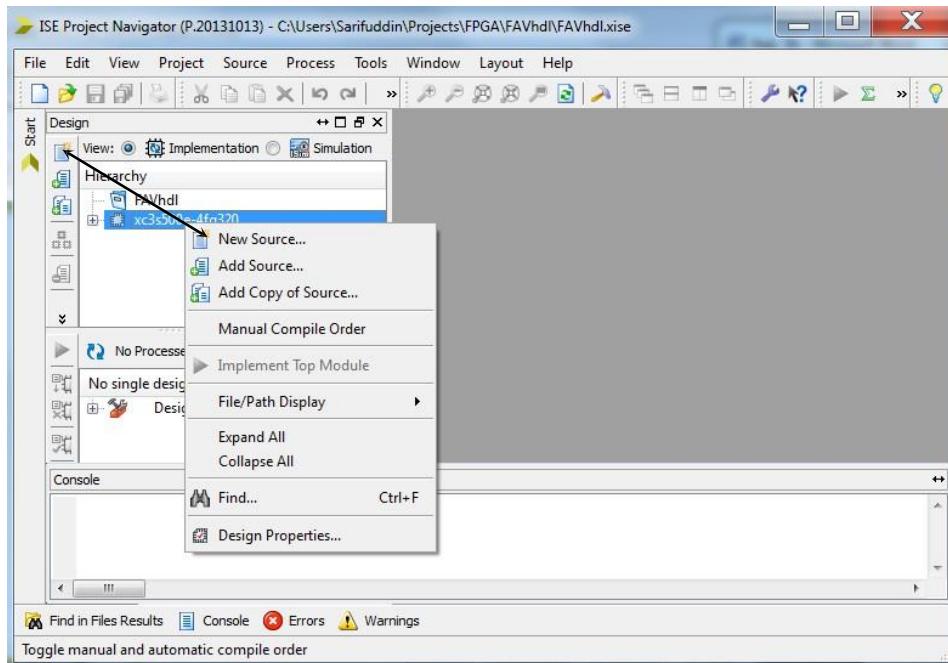


Gambar 3.27. Layar interface ISE Project Navigator setelah file Project dibuat.

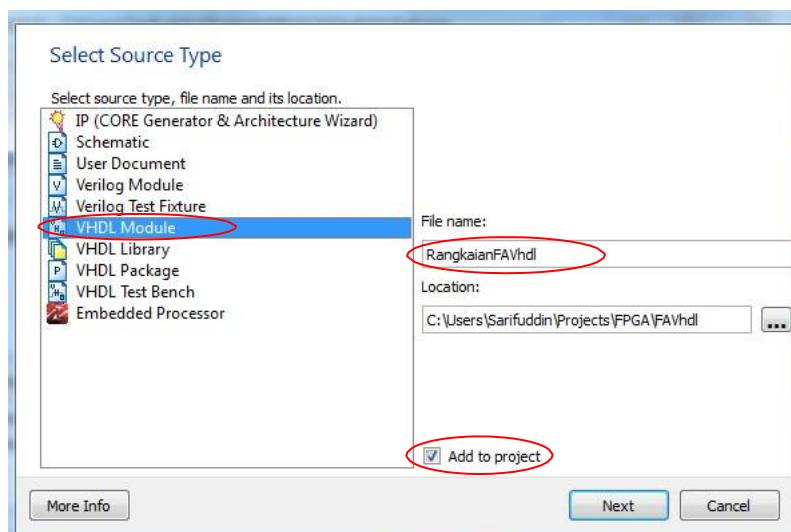
Membuat File Modul Rangkaian VHDL

Setelah project sudah dibuat, langkah berikutnya adalah menambahkan file disain modul rangkaian menggunakan bahasa VHDL kedalam project :

1. Click-kiri pada menu "Project" sehingga muncul pilihan seperti pada gambar 3.28. Ada tiga pilihan yaitu membuat file baru (New source), mengambil file yang sudah ada (Add source) atau mengkopi coding dari sumber file yang sudah ada dan membuat file baru (Add copy of source). Tiga pilihan ada pada bagian kiri panel "View" (lihat anak panah) dan dapat juga dipilih langsung melalui menu icon tersebut. Selanjutnya pilih "New Source" maka akan tampil layar Select source type seperti pada gambar 3.29.
2. Untuk membuat file program VHDL pilih menu "VHDL Module", lalu pada panel "File name" isi nama file sesuai dengan nama rangkaian yang dinginkan, misalnya RangkaianFAVhdl. Pastikan checklist "Add to project" dipilih, lalu click tombol "Next", maka akan tampil layar Define Module seperti pada gambar 3.30-a. Sebelum mengisi tabel yang diminta pada layar Design Modul, pastikan terlebih dahulu variabel-variabel masukkan (in), keluaran (out) dan dua arah masukan- keluaran (inout) apa saja yang dibutuhkan. Misalnya dapat dilihat pada gambar 3.30-b: bila variabel A berupa data masukkan 8 bit, tulis variabel A pada kolom "Port Name", pada kolom "Direction" diset "in". dan pilih checklist pada kolom "Bus" lalu set bit ke-0 sebagai LSB dan bit ke-7 sebagai MSB. Selanjutnya, misalkan untuk variabel B berupa jalur bus data masukkan dan juga sebagai jalur keluaran 8 bit, maka pada kolom "Direction" diset "inout". Contoh lain untuk variabel C sebagai satu bit data masukan: pada kolom "Direction" diset "in", sedang untuk variabel Y sebagai satu bit data keluaran: pada kolom "Direction" diset "out".

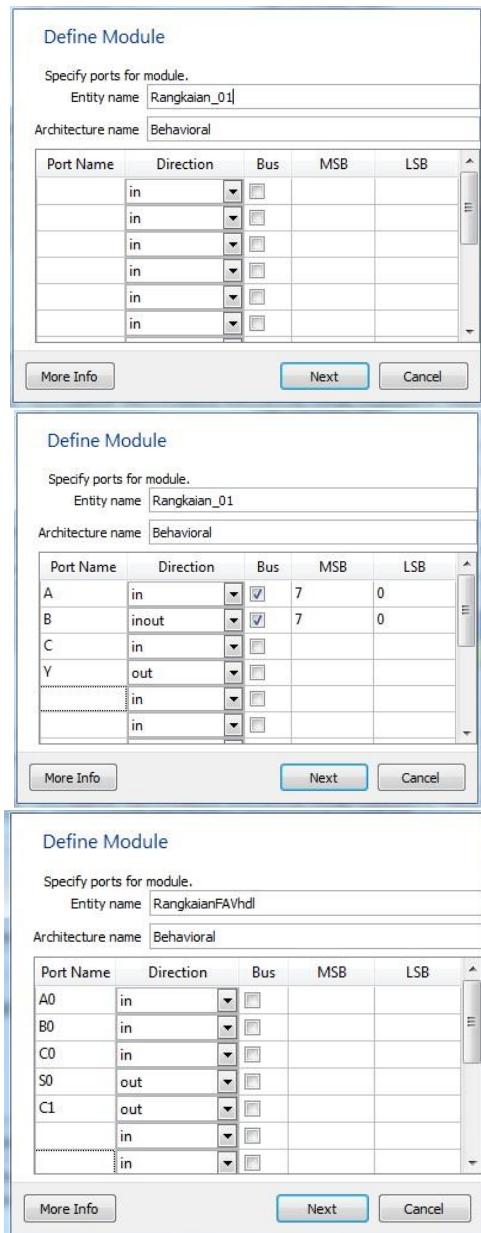


Gambar 3.28. Layar interface menambahkan file pada Project



Gambar 3.29. Layar interface membuat file rangkaia dan pilihan bahasa VHDL.

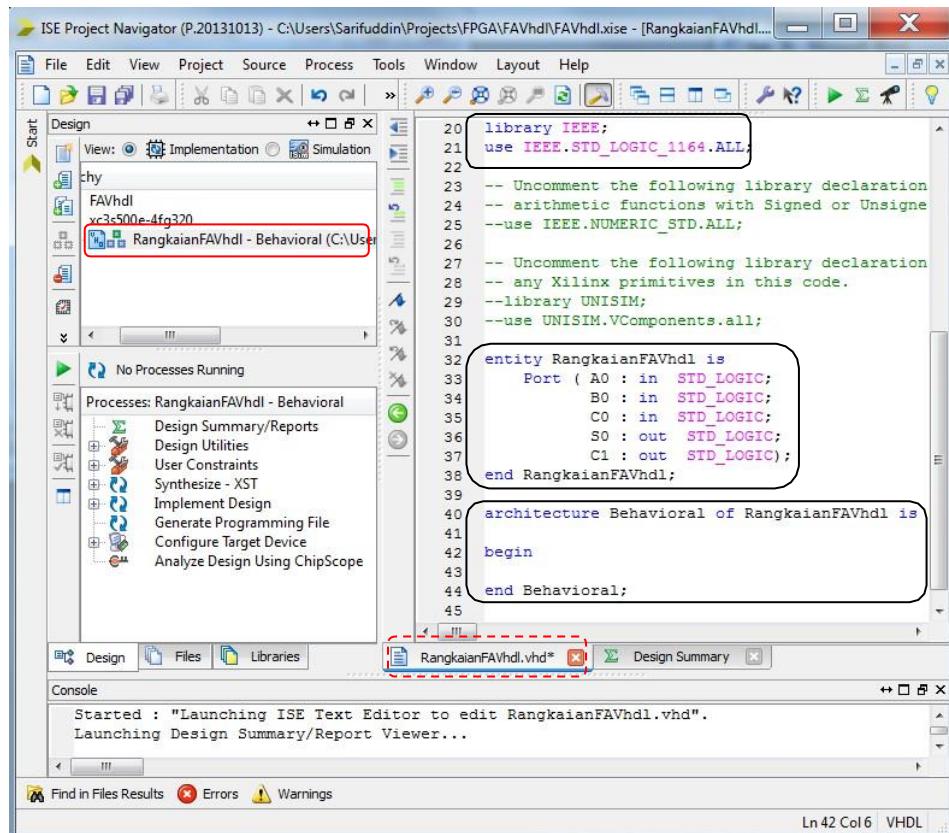
2. Sesuai dengan fungsi rangkaian FA yang memiliki tiga variabel masukkan (A0, B0, C0) dan dua variabel keluaran (S0, C1), maka defenisi masukkan dan keluaran modul rangkaianya diset seperti pada gambar 3.30-c. Selanjutnya click tombol "Next", maka akan muncul layar summary lalu click "Finish" dan kembali ke layar utama seperti pada gambar 3.31.



Gambar 3.30. Layar interface Define Module.

3. Pada bagian bawah panel editor HDL (tanda lingkaran putus-putus) menunjukkan file RangkaianFAVhdl.vhd telah dibuat yang isinya terlihat pada panel editor sebelah kanan. Isi file terbagi dalam tiga bagian yaitu: library IEEE, entity dan architecture behavioral. Bagian entity adalah bagian deklarasi variabel-variabel eksternal yang digunakan. Variabel eksternal adalah variabel yang berhubungan dengan data masukan dan data keluaran dari rangkaian yang akan dibuat. Variabel internal adalah variabel yang digunakan dalam rangkaian misalnya jalur data/sinyal yang menghubungkan antara satu komponen ke komponen lainnya dalam rangkaian tersebut. Variabel internal ini dapat dideklarasikan pada bagian awal dari architecture Behavioral. Pada bagian architecture Behavioral merupakan tempat untuk mendeskripsikan struktur rangkaian yang ingin dibuat dan diletakkan antara stetmen begin dan end Behavioral. Selain itu, pada panel "Processes" ditampilkan sejumlah compilers yang akan digunakan: Design Utilities, User Constraints, Synthesize-XST, Implement Design, Generate Programming File, Cofigure Target Device dan Analyse Design

Using ChipScope.



Gambar 3.31. Layar interface Project Navigator setelah penambahan file VHDL.

4. Sesuai dengan persamaan logika rangkaian FA yang telah disiapkan sebelumnya, maka persamaan logika tersebut dapat langsung ditulis di antara stetmen begin dan end Behavioral, seperti dua contoh pada gambar 3.32-a dan 3.32-b. Sintaks program yang ditambahkan pada gambar 3.32-a adalah langsung menuliskan fungsi logika (tanpa menggunakan variabel internal), sedang pada gambar 3.32-b menggunakan variabel internal. Dalam contoh ini, variabel internal diberi nama w_WIRE-1, w_WIRE-, dan w_WIRE-3 yang diletakkan sebelum stetmen begin. Dua contoh tersebut akan menghasilkan rangkaian yang sama. Setelah fungsi logika rangkaian telah dimasukkan, click tombol menu "Save". Dengan demikian proses pembuatan project dan file modul rangkaian pada level bahasa VHDL telah selesai.

```

23 -- Uncomment the following library declaration
24 -- arithmetic functions with Signed or Unsigned
25 --use IEEE.NUMERIC_STD.ALL;
26
27 -- Uncomment the following library declaration
28 -- any Xilinx primitives in this code.
29 --library UNISIM;
30 --use UNISIM.VComponents.all;
31
32 entity RangkaianFAVhdl is
33     Port ( A0 : in STD_LOGIC;
34             B0 : in STD_LOGIC;
35             C0 : in STD_LOGIC;
36             S0 : out STD_LOGIC;
37             C1 : out STD_LOGIC);
38 end RangkaianFAVhdl;
39
40 architecture Behavioral of RangkaianFAVhdl is
41 begin
42
43     S0 <= A0 xor B0 xor C0;
44     C1 <= (C0 and (A0 xor B0)) or (A0 and B0);
45
46 end Behavioral;
47
48

```



```

31
32 entity RangkaianFAVhdl is
33     Port ( A0 : in STD_LOGIC;
34             B0 : in STD_LOGIC;
35             C0 : in STD_LOGIC;
36             S0 : out STD_LOGIC;
37             C1 : out STD_LOGIC);
38 end RangkaianFAVhdl;
39
40 architecture Behavioral of RangkaianFAVhdl is
41
42     signal w_WIRE_1 : std_logic;
43     signal w_WIRE_2 : std_logic;
44     signal w_WIRE_3 : std_logic;
45
46 begin
47     w_WIRE_1 <= A0 xor B0;
48     w_WIRE_2 <= w_WIRE_1 and C0;
49     w_WIRE_3 <= A0 and B0;
50
51     S0 <= w_WIRE_1 xor C0;
52     C1 <= w_WIRE_2 or w_WIRE_3;
53
54 end Behavioral;
55
56

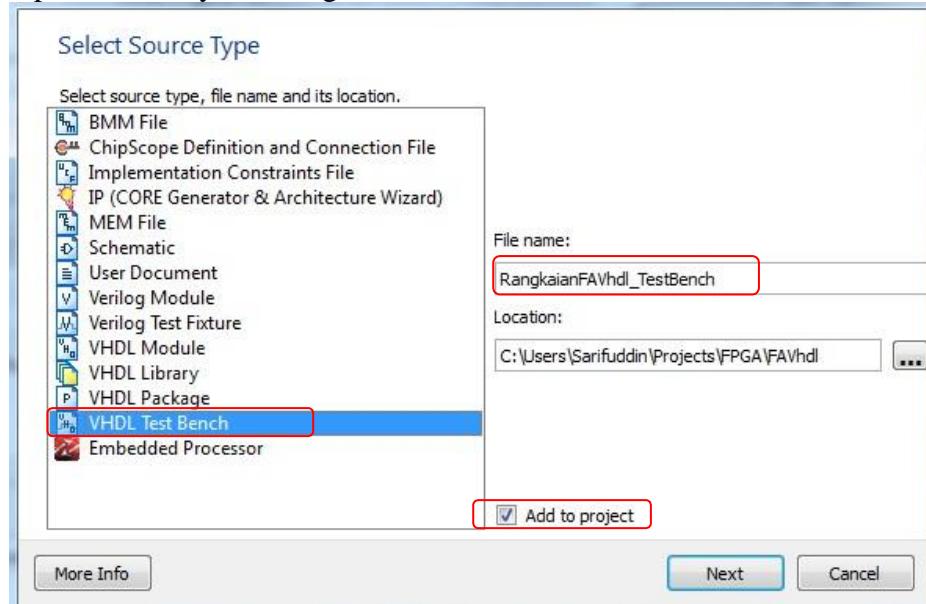
```

Gambar 3.32. Coding simulasi file modul rangkaian FA VHDL
Proses Simulasi Level Modul VHDL

Proses simulasi pada level modul VHDL dapat dilakukan untuk memastikan bahwa coding pada file RangkaianFAVhdl.vhd sudah benar dan berfungsi dengan baik. Berikut urutan proses untuk menjalankan simulasi level modul VHDL.

- Untuk melakukannya, tambahkan file simulasi kedalam projek dengan melakukan seperti pada gambar 3.28. Setelah pilihan "New Source" akan muncul jendela seperti pada gambar 3.33. Tambahkan nama file simulasi pada panel "File name" (misalnya RangkaianFVhdl_TestBench), pada panel sebelah kiri pilih model simulasi "VHDL Test Bench", kemudian "Next", lalu pada jendela Associate

Source yang muncul pilih "RangkaianFAVhdl" dan "Finish", maka secara otomatis akan dibangkitkan source code awal dari file RangkaianFAVhdl_TestBench.vhd dengan mengacu pada file rangkaian yang telah dipilih semula yaitu RangkaianFAVhdl.vhd.



Gambar 3.33. Penambahan file simulasi

2. Pada gambar 3.34 terlihat pada area panel editor source code file RangkaianFAVhdl_TestBench.vhd. Dalam file ini terdapat tiga bagian utama yaitu: LIBRARY, ENTITY dan ARCHITECTURE. Pada bagian awal ARCHITECTURE ini didefinisikan Port dari komponen rangkaian, parameter-parameter (Inputs, Output dan Clock). Bagian dalam ARCHITECTURE antara BEGIN dan END terdapat 3 bagian: Unit Under Test (UUT) Port Map, Penentuan periode proses sinyal clock (bila rangkaian menggunakan clock) dan Stimulus Process (lihat source code yang diberi lingkaran).
3. Karena untuk contoh rangkaian yang dibuat tidak menggunakan sinyal clock, maka semua sintaks pembangkit sinyal clock bisa dihapus. Selanjutnya, Sintaks-sintaks dalam bagian Stimulus Process diganti dengan sintaks-sintaks baru yang menyatakan variasi kombinasi logika dari semua variabel masukkan A0, B0 dan C0, dimana setiap perubahan variasi nilai diselingi waktu tunda sesuai yang diinginkan seperti yang diperlihatkan pada gambar 3.35.

The screenshot shows the ISE Project Navigator interface. The 'View' dropdown is set to 'Simulation'. In the 'Hierarchy' pane, 'RangkaianFAVhdl_TestBench - beh' is selected. The 'Processes' pane shows 'ISim Simulator' and 'Behavioral Check Syntax' listed under 'RangkaianFAVhdl_TestBench - beh'. The main code editor displays VHDL code for a testbench, with several sections highlighted by red dashed boxes:

```

63 constant <clock>_period : time := 10 ns;
64
65 BEGIN
66   -- Instantiate the Unit Under Test (UUT)
67   uut: RangkaianFAVhdl PORT MAP (
68     A0 => A0,
69     B0 => B0,
70     C0 => C0,
71     S0 => S0,
72     C1 => C1
73   );
74
75   -- Clock process definitions
76   <clock>_process :process
77   begin
78     <clock> <= '0';
79     wait for <clock>_period/2;
80     <clock> <= '1';
81     wait for <clock>_period/2;
82   end process;
83
84   -- Stimulus process
85   stim_proc: process
86   begin
87     -- hold reset state for 100 ns.
88     wait for 100 ns;
89     wait for <clock>_period*10;
90     -- insert stimulus here
91     wait;
92   end process;
93
94 END;
95

```

The 'Console' tab shows an error message: 'ERROR:ProjectMgmt - 4 error(s) found while parsing design hierarchy.' The status bar at the bottom right indicates 'Ln 66 Col 6 | VHDL'.

Gambar 3.34. Bentuk struktur file simulasi

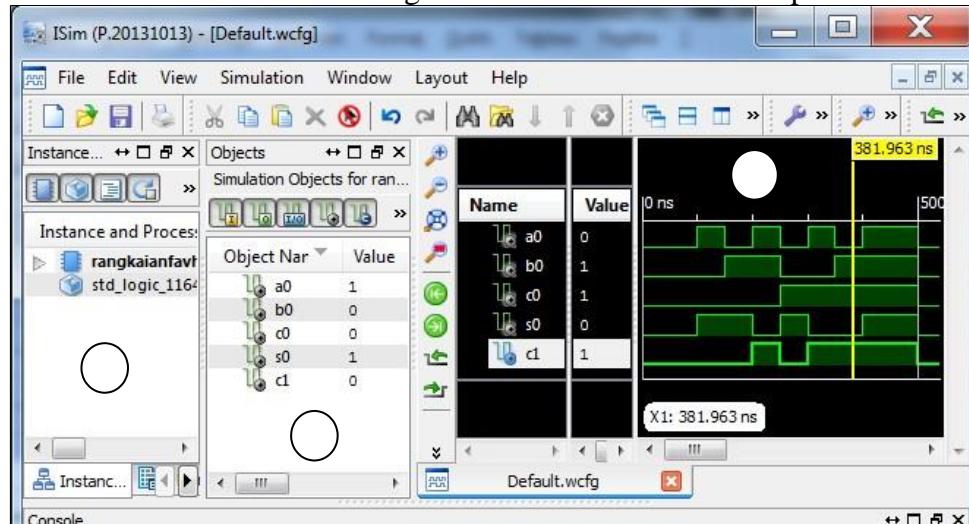
4. Setelah penambahanata revisi coding selesesai, "Save" file. Kemudian, Pada panel "View" pilih checklist "Simulation", "Behavioral" lalu pada "Hierarchy" pilih "RangkaianFAVhdl_TestBench". Jalankan pengecekan sintaks dengan cara klik dua kali simulator "Behavioral Check Syntax" yang terletak pada panel "Processes": "ISim Simultor", tunggu hingga checklist berwarna hijau muncul dan pada panel Console muncul inromasi: " Process "Simulate Behavioral Model" completed successfully". Bila terdapat error, cek kembali codingnya pada posisi yang ditunjuk, betulkan, lalu save dan jalankan kembali "Behavioral Check Syntax".
5. Setelah proses cek sintaks selesai, jalankan proses simulasi. Klik dua kali "Simulate Behavioral Model", tunggu hingga intarface simulasi ISim ditampilkan. Hasilnya diperlihatkan seperti pada gambar 3.36. Pada layar ini terdapat empat area: pertama adalah list nama file Tets Bench yang sedang di-running (pada contoh ini hanya ada RangkaianFAVhdl), kedua adalah nama- nama variabel (Object Name) yang digunakan dan nilai logikanya (Value), ketiga adalah diagram sinyal setiap variabel sesuai dengan yang telah dideklarasikan, serta keempat adalah area Console.

```

67 -- Instantiate the Unit Under Test (UUT)
68 uut: RangkaianFAVhdl PORT MAP (
69     A0 => A0,
70     B0 => B0,
71     C0 => C0,
72     S0 => S0,
73     C1 => C1
74 );
75
76 -- Stimulus process
77 Process
78 Begin
79     wait for 50ns;
80     A0<='0'; B0<='0'; C0<='0';
81     wait for 50ns;
82     A0<='1'; B0<='0'; C0<='0';
83     wait for 50ns;
84     A0<='0'; B0<='1'; C0<='0';
85     wait for 50ns;
86     A0<='1'; B0<='1'; C0<='0';
87     wait for 50ns;
88     A0<='0'; B0<='0'; C0<='1';
89     wait for 50ns;
90     A0<='1'; B0<='0'; C0<='1';
91     wait for 50ns;
92     A0<='0'; B0<='1'; C0<='1';
93     wait for 50ns;
94     A0<='1'; B0<='1'; C0<='1';
95     wait for 50ns;
96 end process;
97
98 END;

```

Gambar 3.35. Edit coding varisi nilai masukkan untuk proses simulasi.



Gambar 3.36. Interface ISim menampilkan hasil simulasi.

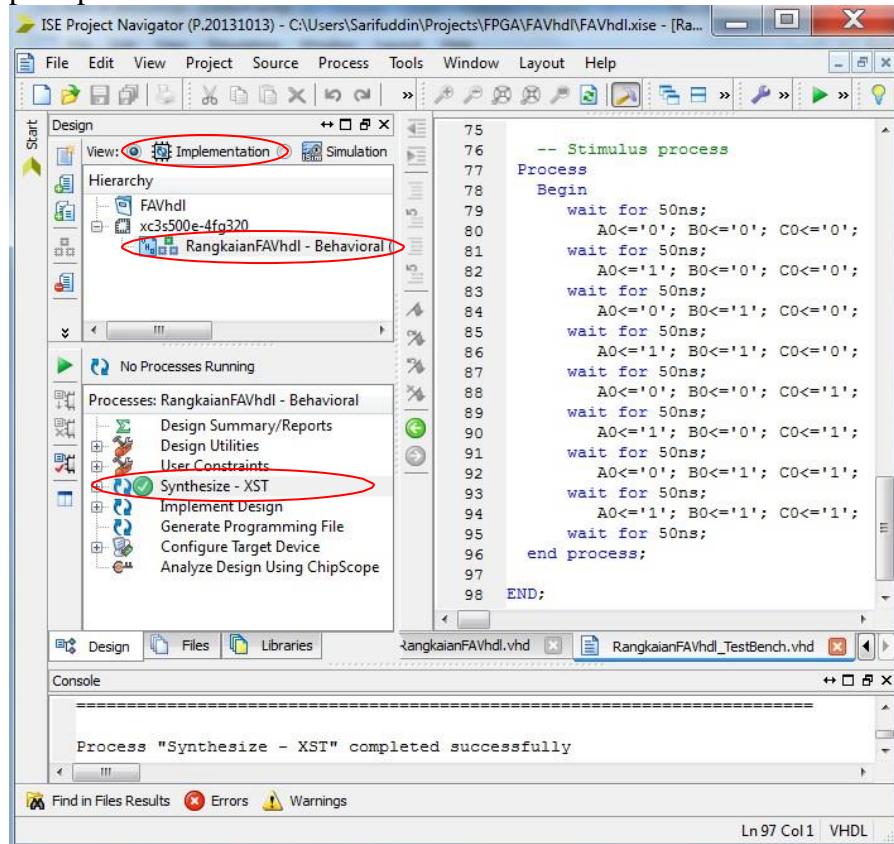
5. Proses Sintesis VHDL

Setelah proses simulasi selesai dilakukan dan telah yakin hasilnya sudah sesuai dengan yang sebenarnya, maka tahap selanjutnya adalah proses sintesis (Synthesize). Proses ini akan mengkonversi code VHDL menjadi bahasa assembler yang mendeskripsikan gerbang-gerbang logika dan jalur-jalur keterhubungannya sehingga membentuk sebuah rangkaian digital. Berikut urutan langkah-langkah proses sintesi:

1. Sebelum memulai proses sintesis, pastikan checkbox "Implementation" pada

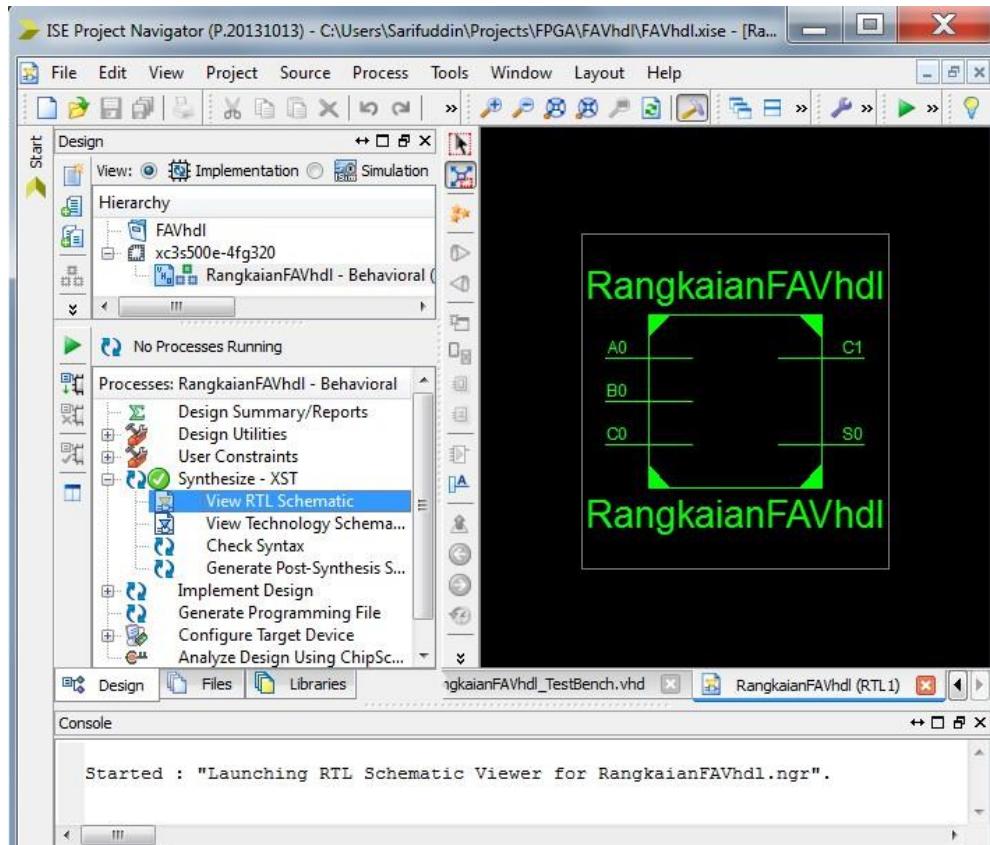
panel "Design" - "View" sudah dipilih. Kemudian pilih file rangkaian "RangkaianFAVhdl".

- Pada panel "Processes":, click dua kali pada "Synthesize - XST", tunggu hingga proses sintesis dilakukan dan pada panel "Console" gambar 3.37 menampilkan informasi hasil proses: "Process "Synthesize - XST" completed successfully". Hal ini berarti proses sintesis berhasil tanpa kesalahan dan diberi tanda Synthesize - XST. Bila saat proses sintesis ditemukan kesalahan penulisan sintaks pada code VHDL, maka akan dinyatakan pada panel Console kesalahan sintaks tersebut dan diberi tanda Synthesize - XST. Untuk melihat jenis error-nya click tombol/icon pada panel Console.

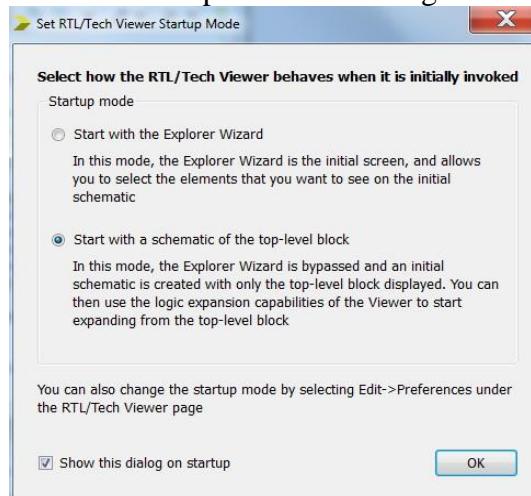


Gambar 3.37. Layar interface Project Navigator setelah proses sintesis.

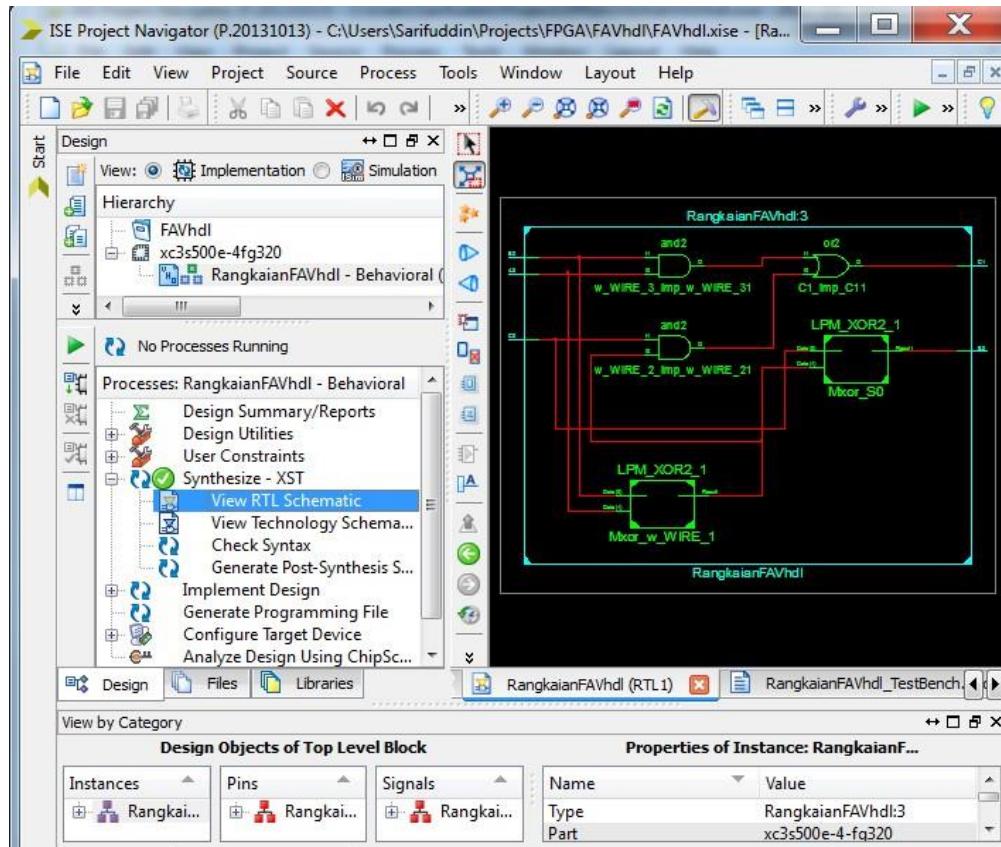
- Setelah proses sintesis berhasil dengan sukses, Hal ini bermakna pula rangkaian skematis dari persamaan logika sudah terbentuk. Untuk melihat bentuk rangkaian tersebut click icon pada "Synthesis-XST" sehingga terlihat turunan semua proses yang sudah dilakukan, seperti yang diperlihatkan oleh gambar 3.38 sebelah kiri. Click dua kali pada "View RTL Schematic", maka akan tampak seperti pada gambar 3.39. Pilih Start whit a schematic of the top-level block kemudian tekan OK. Selanjutnya, pada area editor akan ditampilkan kotak rangkaian dengan nama RangkaianFAVhdl yang memiliki tiga jalur masukkan A0, B0 dan C0 serta dua jalur keluaran S0 dan C1.
- Untuk melihat rangkaian logika di dalam kotak tersebut, click dua kali pada kotak itu kemudian pilih "Show Block Contents" pada jendela yang muncul. Untuk melihat seluruh rangkaian secara utuh, click pada icon Zoom to full view. Tampak pada gambar 3.40 bentuk rangkaian Full Adder yang terbentuk menggunakan 2 gerbang AND, 2 gerbang XOR dan satu gerbang OR.



Gambar 3.38. Menampilkan skema rangkaian top level.



Gambar 3.39. Layar interface pilihan model tampilan.

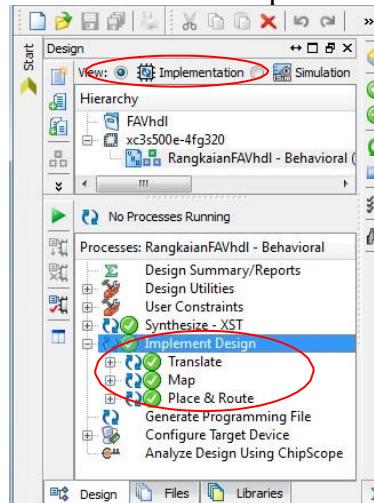


Gambar 3.40. Layar interface tampilan rangkaian logika (low level).

Implement Disain VHDL

Tahap implementasi (Implement) bertujuan untuk mendefinisikan konfigurasi perangkat keras. Ada tiga proses yang dilakukan yaitu Translate, Mapping dan PAR (Place and Route).

1. Pada panel "Processes", klik kanan dua kali pada menu "Implement Disain" dan tunggu hingga tanda checklist berwarna hijau muncul seperti yang terlihat pada gambar 3.41. Pada panel Console akan muncul pesan " Process "Generate Post-Place & Route Static Timing" completed successfully". Selanjutnya untuk melihat berapa jumlah sumber daya yang akan digunakan dalam IC FPGA, klik "Desing Summery (Implemented)" pada panel sebelah kanan bawah. Telihat pada gambar 3.42 resume tersebut, setelah proses palce & route, rangkaian FA yang dibuat menempati atau menggunakan 1 slice dan 5 pin IOB.

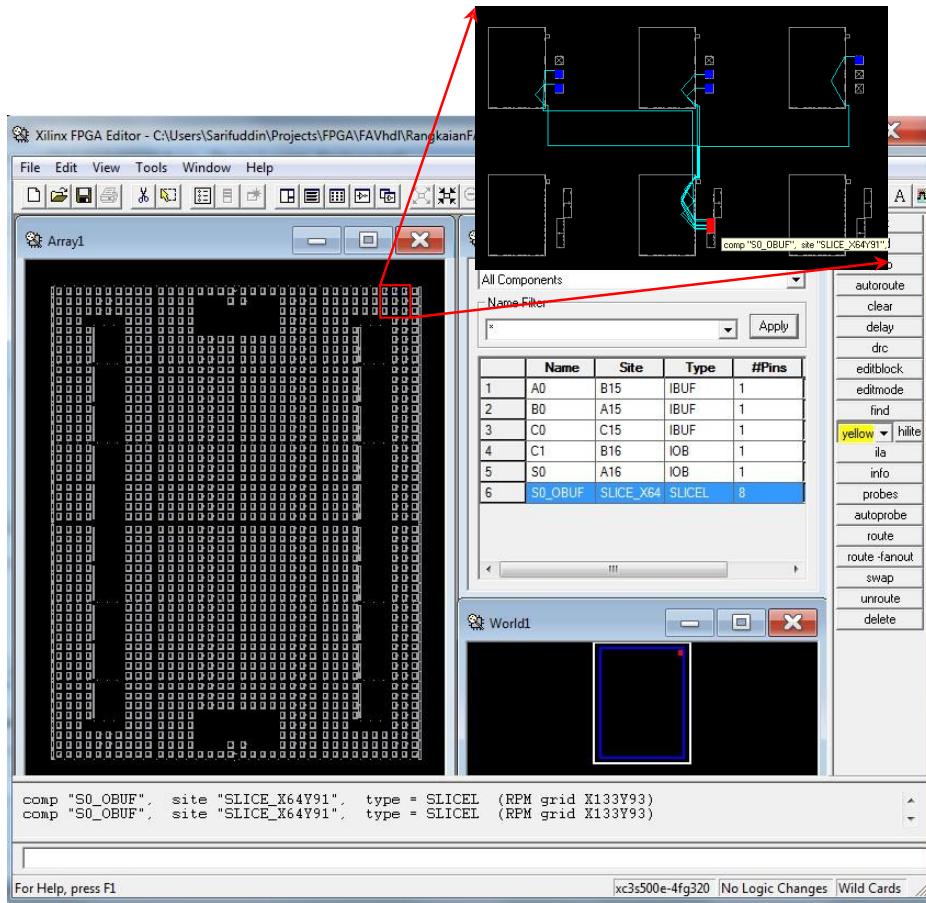


Gambar 3. 41.Proses Implement Design

Device Utilization Summary				L-1
Logic Utilization	Used	Available	Utilization	Note(s)
Number of 4 input LUTs	2	9,312	1%	
Number of occupied Slices	1	4,656	1%	
Number of Slices containing only related logic	1	1	100%	
Number of Slices containing unrelated logic	0	1	0%	
Total Number of 4 input LUTs	2	9,312	1%	
Number of bonded IOBs	5	232	2%	
Average Fanout of Non-Clock Nets	1.60			

Gambar 3. 42. Rangkuman penggunaan sumber daya FPGA

2. Selanjutnya, untuk melihat hasil implementasi rangkaian di dalam IC FPGA, klik dua kali pada menu "View/Edit Routed Design (FPGA Editor)" yang terletak dalam menu "Place & Route", maka akan tampil interface Xilinx FPGA Editor gambar 3.43-a. Interface ini terdiri dari tiga bagian: panel sebelah kiri berisi blok-blok sumberdaya yang ada dalam IC FPGA, panel sebelah kanan atas menampilkan Pin IO dan Slice yang digunakan serta posisinya masing-masing. Blok Slice dan I/O yang telah digunakan berwarna biru dan yang masih kosong berwarna putih. Untuk RangkaianFAVhdl yang dibuat terletak pada bagian paling kanan atas. Pilih menu simbol dan pilih area RangkaianFAVhdl yang akan dizoom lalu tekan menu simbol ZoomIn , maka akan tampak pembesaran dan dapat terlihat Slice dan I/O mana saja yang digunakan serta jalur-jalur kabel penghubungnya (seperti pada gambar 3.43-b). Pada interface ini dapat dilakukan modifikasi (mengedit) posisi Sclice dan/atau posisi Pin I/O serta jalur-jalur penghubung antar Slice dan I/O. Untuk melakukan hal tersebut dan mengetahui lebih dalam tentang Xilinx FPGA Editor, silahkan pelajari lebih mendalam dokumentasi yang tersedia melalui menu "Help"-nya.

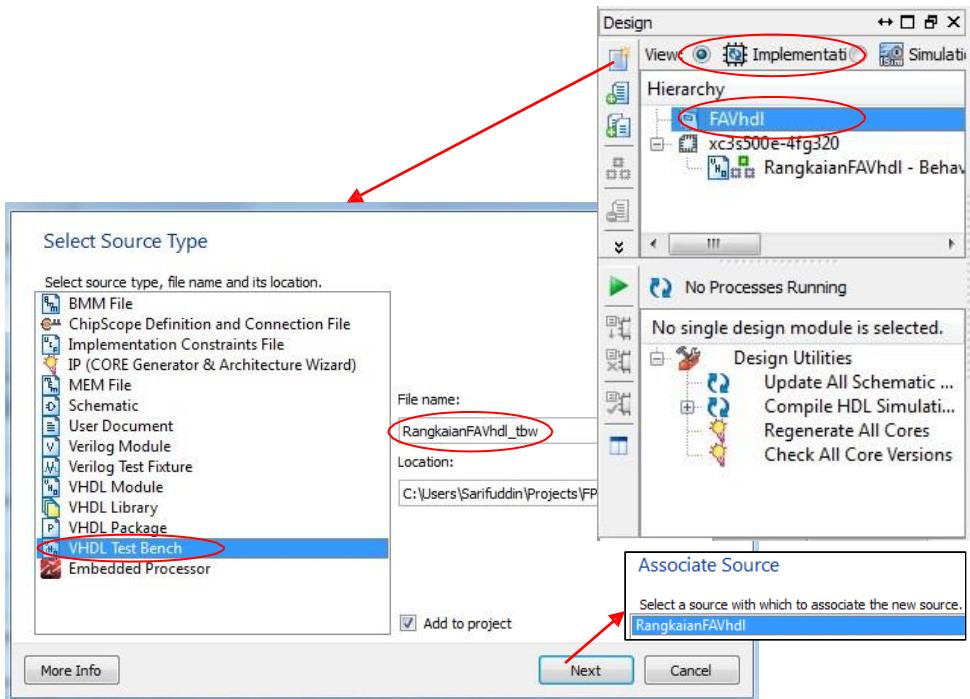


Gambar 3.43. Bentuk implementasi rangkaian FA pada level Place & Route VHDL.

Simulasi Pada level Place & Route VHDL

Setelah proses Route perlu dilakukan kembali simulasi sinyal untuk memastikan bahwa rangkaian yang telah diimplementasi hingga pada level Route tetap berfungsi sesuai dengan disain semula. Berikut tahapan proses simulasai:

1. Dari interface ISE Project Navigator, pada panel "View" pilih "Implementation" dan pada panel "Hierarchy" pilih project "FAVhdl", lalu pilih "New Source" (lihat gambar 3.44-a), maka akan tampak jendela baru seperti pada gambar 3.44-b. Buat file program Verilog untuk simulasi level Route, misalnya pada panel File name tulis nama file RangkaianFAVhdl_tbw lalu pilih menu "VHDL Test Bench" dan klik "Next", akan muncul jendela baru gambar 3.44-c. Pada jendela tersebut akan muncul pilihan rangkaian yang akan digabungkan dan disimulasikan melalui file RangkaianFAVhdl_tbw. Pilih RangkaianFAVhdl lalu tekan menu "Next". Kembali ke interface ISE Project Navigator dan pada layar sebelah kanan akan secara otomatis muncul coding Verilog dari file RangkaianFAVhdl_tbw.vhd.



Gambar 3.44. Pembuatan file simulasi level Route VHDL.

1. Coding file RangkaianFAVhdl_tbw.vhd untuk simulasi level Route identik dengan coding simulasi level disain modul behavioral VHDL. Bagian blok "Stimulus Process" harus ditambahkan misalnya seperti yang diberi tanda lingkaran (lihat gambar 3.45).

The screenshot shows the ISE Project Navigator interface with the following details:

- File Menu:** File, Edit, View, Project, Source, Process, Tools, Window, Layout, Help.
- View Bar:** Design, Implementation, Simulation (highlighted).
- Hierarchy Tree:** FAVhdl > xc3s500e-4fg320 > RangkaianFAVhdl TestBench - behavior > RangkaianFAVhdl_tbw - behavior (C:\Us).
- Processes Panel:** Processes: RangkaianFAVhdl_tbw - behavior
 - ISim Simulator
 - Post-Place & Route Check Syntax (highlighted)
 - Simulate Post-Place & Route Model
- Code Editor:**

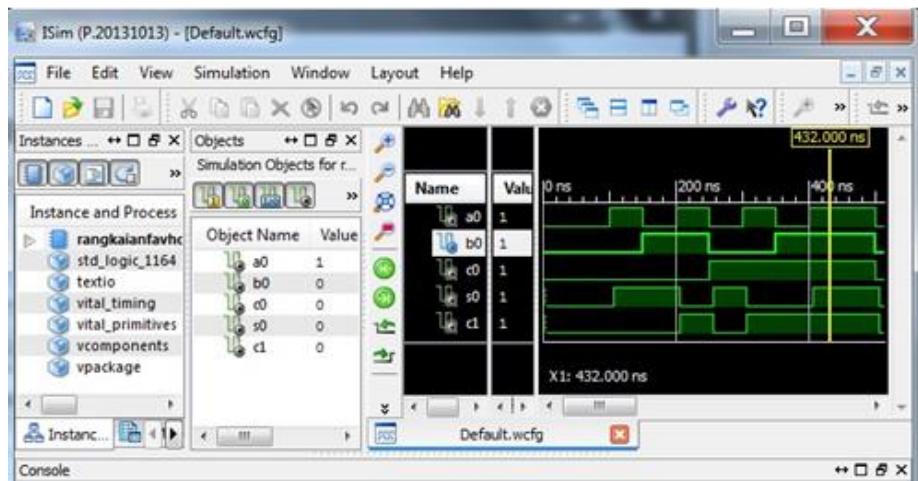
```

68 -- Instantiate the Unit Under Test
69 uut: RangkaianFAVhdl PORT MAP (
70   A0 => A0,
71   B0 => B0,
72   C0 => C0,
73   S0 => S0,
74   C1 => C1
75 );
76
77 -- Stimulus process
78 process
79 begin
80   wait for 50ns;
81   A0<='0'; B0<='0'; C0<='0';
82   wait for 50ns;
83   A0<'1'; B0<='0'; C0<='0';
84   wait for 50ns;
85   A0<='0'; B0<='1'; C0<='0';
86   wait for 50ns;
87   A0<'1'; B0<='1'; C0<='0';
88   wait for 50ns;
89   A0<='0'; B0<='0'; C0<='1';
90   wait for 50ns;
91   A0<='1'; B0<='0'; C0<='1';
92   wait for 50ns;
93   A0<='0'; B0<='1'; C0<='1';
94   wait for 50ns;
95   A0<='1'; B0<='1'; C0<='1';
96   wait for 50ns;
97 end process;
98
99 END;

```
- Console Panel:** Process "Post-Place & Route Check Syntax" completed successfully.
- Bottom Status:** Ln 76 Col 1 VHDL.

Gambar 3.45. File coding simulasi dan proses simulasi level Route VHDL.

2. Pada panel View sebelah kiri pilih: "Simulation" dan "Post-Route" serta file RangkaianFAVhdl.tbw.vhd, seperti yang dilingkari pada gambar 3.45. Selanjutnya pada panel Processes: ISim Simulator, klik dua kali pada "Post-Place & Route Check Syntax" hingga tanda checklist hijau muncul. Kemudian klik pada panel Console untuk memastikan muncul "Process "Simulate Post-Place & Route Model" completed successfully", yang menunjukkan codingnya sudah benar. Bila terdapat error, betulkan kembali codingnya sesuai dengan posisi yang ditunjuk, kemudian jalankan kembali "Post-Place & Route Check Syntax". Selanjutnya, klik dua kali pada "Simulate Post-Place & Route Model". Bila proses simulasinya sukses maka akan muncul interface ISim yang menampilkan sinyal masukan dan keluaran hasil proses simulasi seperti yang diperlihatkan pada gambar 3.46. Hasil simulasi sinyal masukan dan keluarannya dapat dilihat pada bagian kanan. Pada panel bagian kiri dapat dilihat berbagai informasi seperti standar logic yang digunakan, timing, components, package dan lainnya.

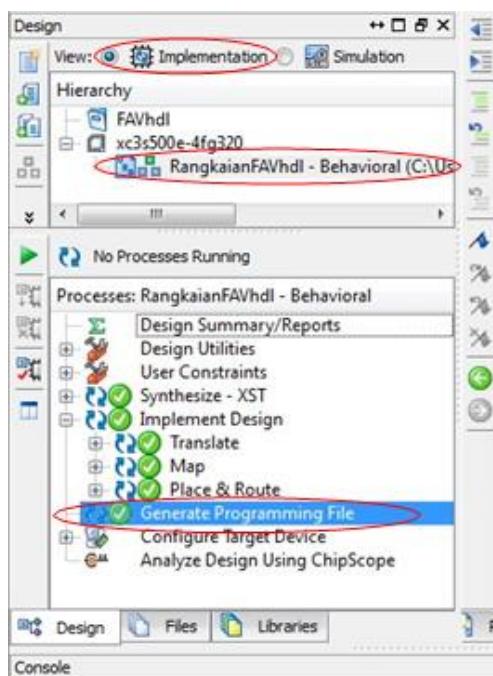


Gambar 3.46. Sinyal hasil simulasi level Route VHDL.

3. Tahap selanjutnya adalah proses kompilasi oleh Bitstream Generator (BitGen) untuk membangkitkan Bit file. Pada ISE Navigator dan panel "View" pilih "Implementation" dan file RangkaianFAVhdl.vhd (lihat gambar 3.47). Kemudian pada panel "Processes" klik dua kali pada "GenerateProgramming File", tunggu hingga tanda checklist hijau muncul dan pastikan pada panel "Console" muncul "Process 'Generate Programming File' completed successfully".

Tahap terakhir adalah proses transfer Bit file ke FPGA Board. Sebelum melanjutkan ke tahap ini pastikan:

1. Pastikan bahwa proses "Synthesize", "Implement" dan "GenerateProgramming File" dilakukan secara berurutan karena proses-proses tersebut bergantung satu terhadap yang lainnya.
2. Pastikan hubungan kabel USB antara PC yang digunakan dan FPGA Board tersambung, power listrik menyala dan pastikan semua koneksi ke board berfungsi dengan baik.



Gambar 2.47. Proses Pembangkitan Bit file.