

BAB II

Coding Structural, Data Flow, dan Behavioral

2.1. Tujuan

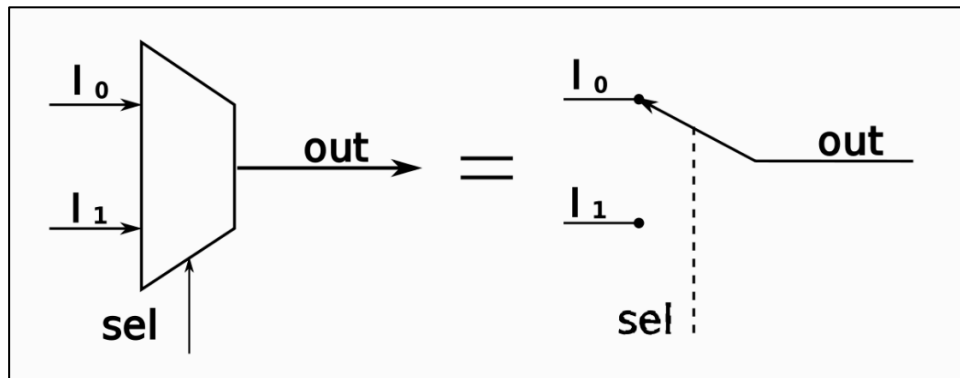
1. Praktikan dapat memahami perbedaan dari tiga level abstraksi kode Verilog.
2. Praktikan dapat menulis dan menjalankan kode Verilog menggunakan aplikasi Xilinx Vivado.
3. Praktikan dapat melakukan konfigurasi papan Nexys A7 untuk mengimplementasikan kode Verilog.

2.2. Alat dan Bahan

1. PC/Laptop
2. Papan Nexys A7
3. Xilinx Vivado

2.3. Dasar Teori

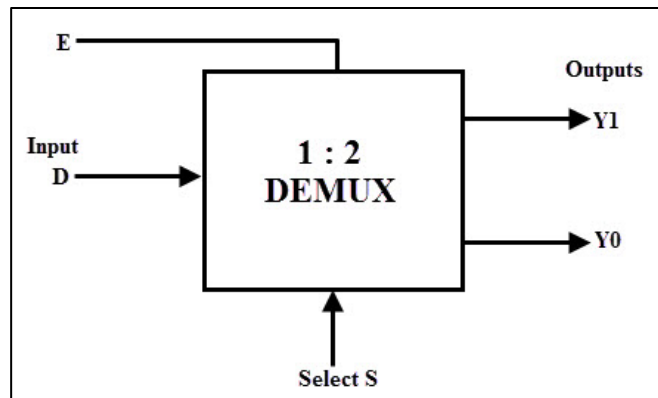
2.3.1. Multiplexer



Multiplexer adalah perangkat yang dapat membolehkan satu atau lebih sinyal analog atau digital untuk berjalan pada satu sambungan transmisi komunikasi. Tujuan dari multiplexer adalah untuk menggabungkan dan mengirimkan sinyal pada media yang tergabung untuk mengoptimasi efisiensi dan mengurangi biaya komunikasi

(Sumber: technopedia.com)

2.3.2. Demultiplexer



Demultiplexer merupakan perangkat yang memiliki fungsi kebalikan dari multiplexer dimana jika multiplexer menggabungkan beberapa sinyal komunikasi, demultiplexer membagi sinyal komunikasi menjadi beberapa bagian. Jadi demultiplexer mengambil satu masukkan sinyal dan mengubahnya ke salah satu keluaran individual satu per satu.

(Sumber: electronics-tutorials.ws)

2.3.3. Papan FPGA

FPGA (Field Programmable Gate Arrays) adalah sirkuit terintegrasi yang perangkat kerasnya dapat dikonfigurasi untuk memenuhi kebutuhan spesifik dari pengguna setelah proses manufaktur. Hal ini membolehkan peningkatan fitur dan perbaikan kerusakan langsung di tempat.

(Sumber: arm.com)

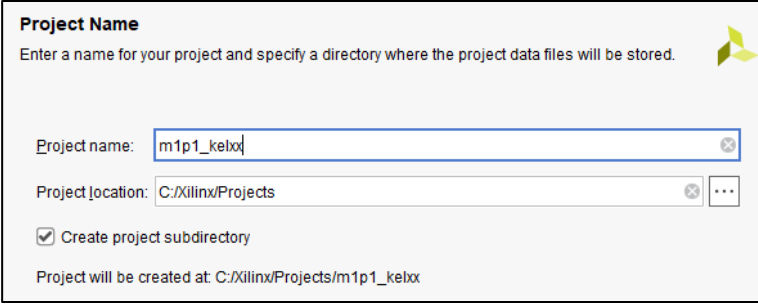
2.4. Percobaan

2.4.1. Percobaan 1

1. Jalankan program **Vivado 2022.2** dan buat *project* baru.



2. Beri nama *project m1p1_kelXX* (XX diganti nomor kelompok).



Project Name
Enter a name for your project and specify a directory where the project data files will be stored.

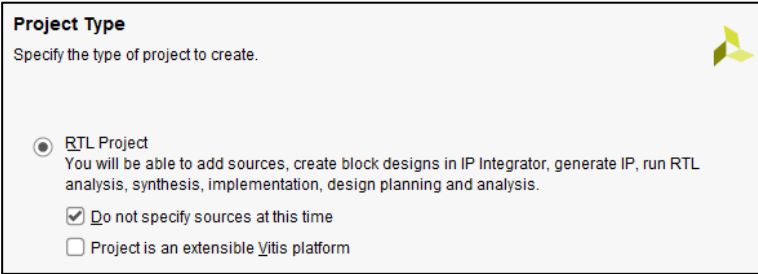
Project name:

Project location:

☒ Create project subdirectory

Project will be created at: C:/Xilinx/Projects/m1p1_kelxx

3. Pilih tipe RTL Project dan centang opsi Do not specify sources at this time.



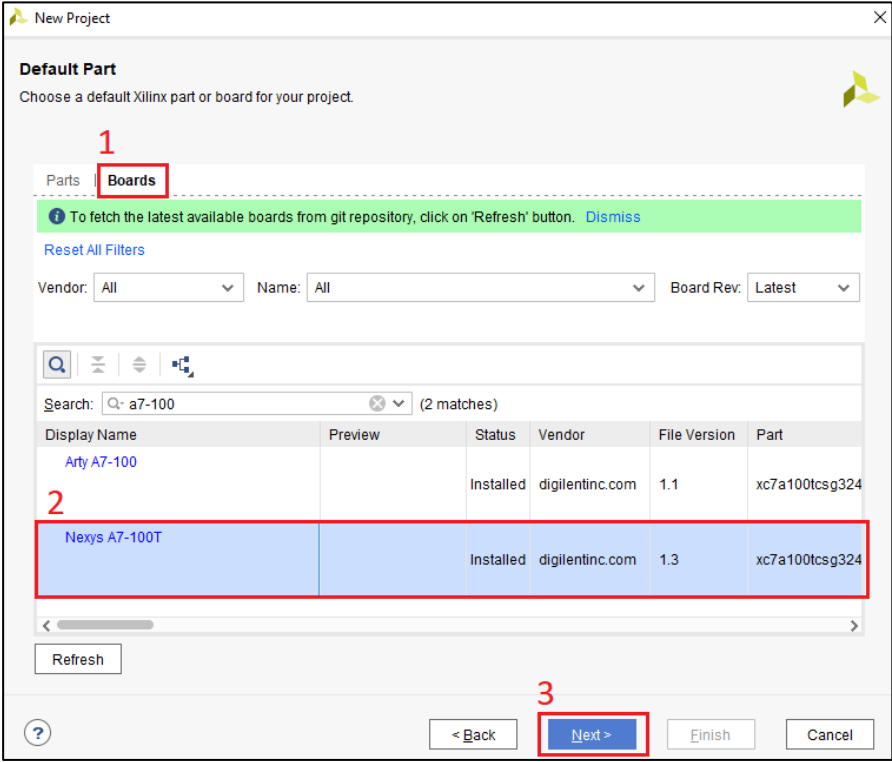
Project Type
Specify the type of project to create.

☒ **RTL Project**
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

☒ Do not specify sources at this time

☐ Project is an extensible Vitis platform

4. Pilih papan **NexysA7-100T**.



New Project

Default Part
Choose a default Xilinx part or board for your project.

1 **Boards**

2 **Nexys A7-100T**

3 **Next >**

Search: (2 matches)

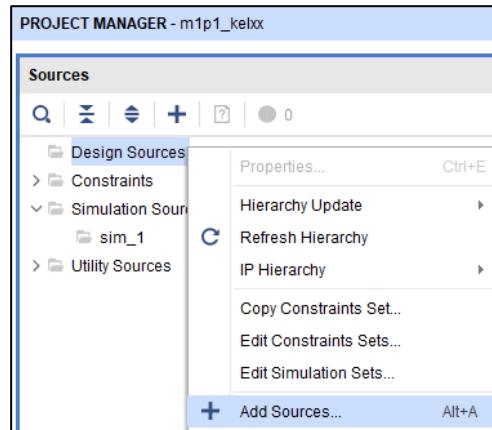
Display Name	Preview	Status	Vendor	File Version	Part
Arty A7-100		Installed	digilentinc.com	1.1	xc7a100tcsg324
Nexys A7-100T		Installed	digilentinc.com	1.3	xc7a100tcsg324

Refresh

< Back **Next >** Finish Cancel

5. Klik **Finish** dan tunggu hingga *project* selesai dibuat.

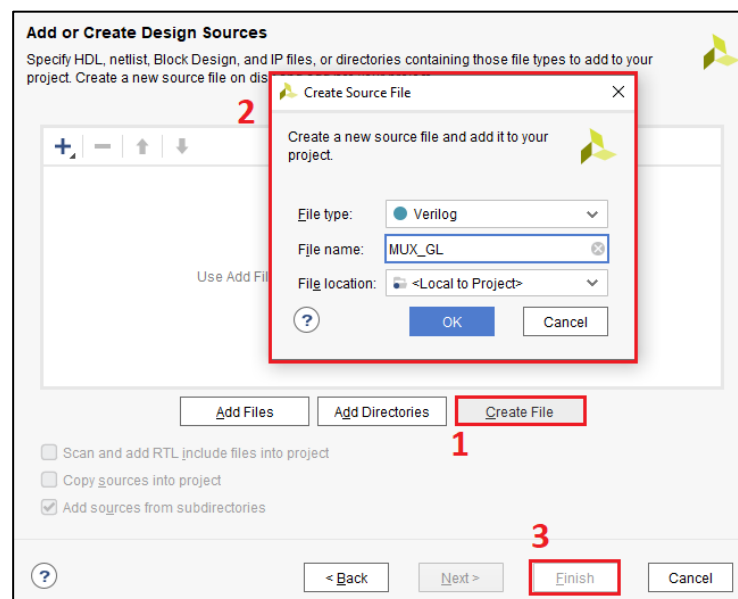
6. Pada bagian **Sources**, klik kanan pada **Design Sources** dan pilih **Add Sources**.



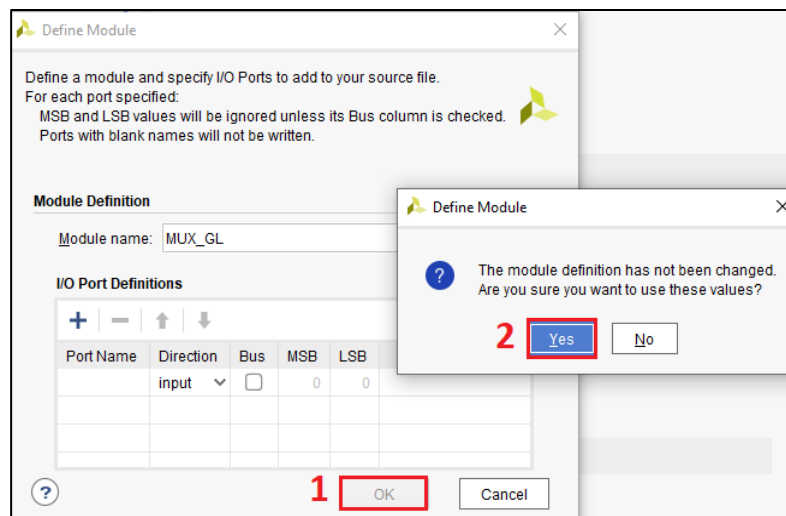
7. Pilih **Add or create design sources**.



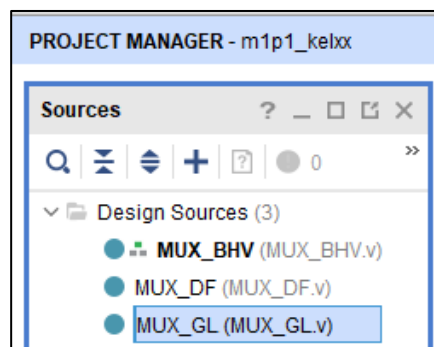
8. Buat file **MUX_GL** dengan tipe **Verilog**.



9. Pada bagian **Define Module**, klik **OK** lalu **Yes**.



10. Ulangi Langkah 6-9 untuk *file* **MUX_DF** dan **MUX_BHV**. Sehingga hasil akhirnya seperti berikut.



11. Buka masing-masing *file* dan tambahkan kode berikut. Jangan lupa untuk menyimpan perubahan setiap file.

MUX_GL.v
<pre>`timescale 1ns / 1ps module muxgl(output c, input a, b, s); wire nots; wire and1, and2; not (nots, s); and (and1, a, nots); and (and2, b, s); or (c, and1, and2); endmodule</pre>

MUX_DF.v

```
`timescale 1ns / 1ps

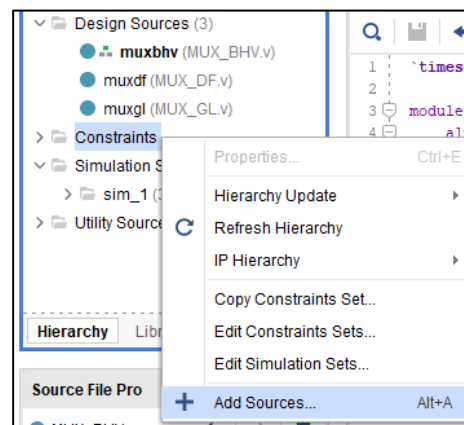
module muxdf(output c, input a, b, s);
    assign c = (a&~s) | (b& s);
endmodule
```

MUX_BHV.v

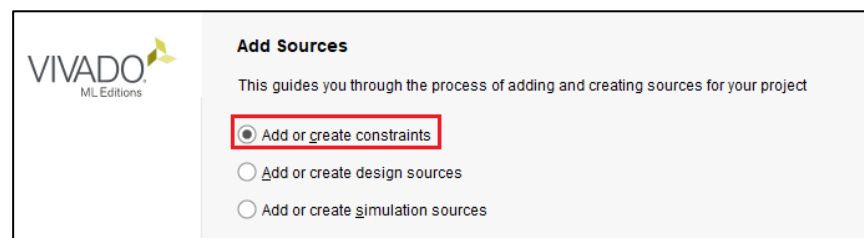
```
`timescale 1ns / 1ps

module muxbhv(output reg c, input a, b, s);
    always @ (a or b or s)
    begin
        case(s)
            0: c <= a;
            1: c <= b;
        endcase
    end
endmodule
```

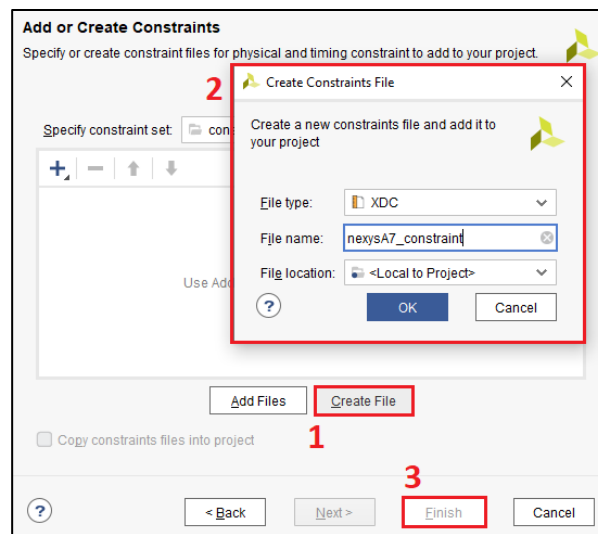
12. Pada folder **Constraints**, tambahkan *source* baru.



13. Pilih **Add or create constraints**.



14. Tambahkan *file* **nexysA7_constraint** dengan tipe **XDC**.



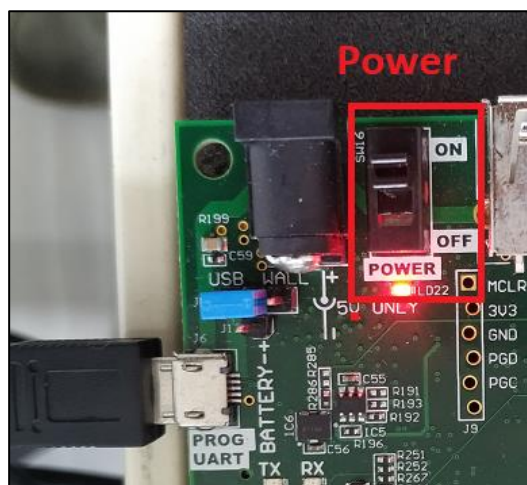
15. Buka *file* **nexysA7_constraint** dan tambahkan kode berikut.

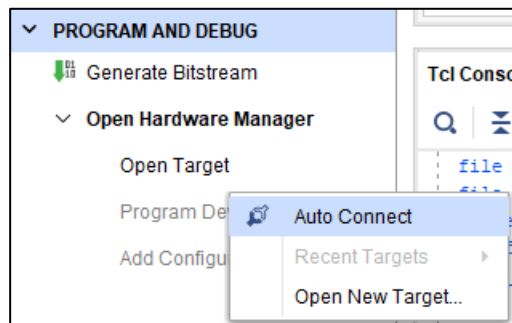
NexysA7 constraint.v

```
##Switch
set_property -dict { PACKAGE_PIN J15      IOSTANDARD LVCMOS33
} [get_ports { a }]; IO_L24N_T3_RS0_15 Sch=a
set_property -dict { PACKAGE_PIN L16      IOSTANDARD LVCMOS33
} [get_ports { b }]; IO_L3N_T0_DQS_EMCCLK_14 Sch=b
set_property -dict { PACKAGE_PIN M13      IOSTANDARD LVCMOS33
} [get_ports { s }]; IO_L6N_T0_D08_VREF_14 Sch=s

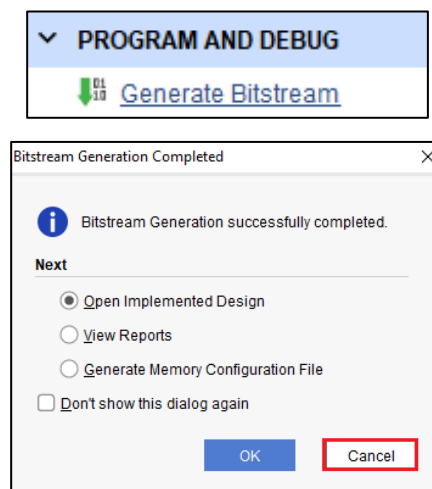
##LED
set_property -dict { PACKAGE_PIN H17      IOSTANDARD LVCMOS33
} [get_ports { c }]; IO_L18P_T2_A24_15 Sch=c
```

16. Sambungkan papan Nexys A7 dengan PC/laptop dan hubungkan ke Vivado.

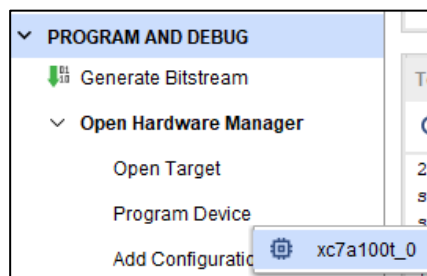




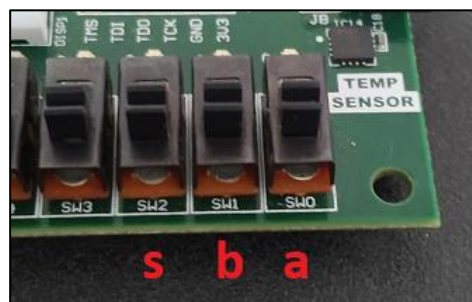
17. Klik **Generate Bitstream** dan tunggu hingga prosesnya selesai.



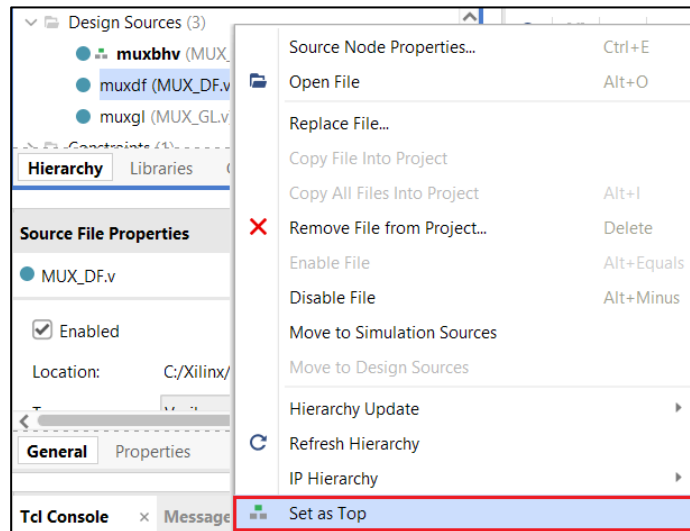
18. Klik **Program Device** dan tunggu hingga prosesnya selesai.



19. Amati perilaku papan dengan *switch* SW0, SW1, and SW2.



20. Untuk mengubah file yang dipilih untuk papan Nexys A7, pilih opsi **Set as top**. Kemudian, lakukan **Generate Bitstream** dan **Program Device** Kembali.



2.4.2. Percobaan 2

1. Buat *project* baru dan beri nama **m1p2_kelXX** (XX diganti nomor kelompok).
2. Buat 3 *file* Verilog dengan ketentuan sebagai berikut:

DEMUX_GL.v

```
`timescale 1ns / 1ps

module demuxgl(output a, b, input c, s);
    wire nots;

    not (nots, s);

    and (a, c, nots);
    and (b, c, s);
endmodule
```

DEMUX_DF.v

```
`timescale 1ns / 1ps

module demuxdf(output a, b, input c, s);
    assign a = c&~s;
    assign b = c&s;
endmodule
```

DEMUX_BHV.v

```
`timescale 1ns / 1ps

module demuxbhv(output reg a, b, input c, s);

    always @ (c or s)
    begin
        case(s)
            0: begin
                a <= c; b = 0;
            end
            1: begin
                b <= c; a = 0;
            end
        endcase
    end
endmodule
```

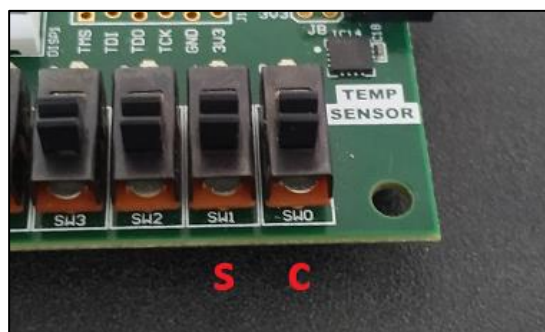
3. Buat file *constraint* **nexysA7_constraint.xdc** dan isikan dengan kode berikut:

nexysA7_constraint.xdc

```
##Switch
set_property -dict { PACKAGE_PIN J15    IOSTANDARD LVCMOS33
} [get_ports { c }]; IO_L24N_T3_RS0_15 Sch=c
set_property -dict { PACKAGE_PIN L16    IOSTANDARD LVCMOS33
} [get_ports { s }]; IO_L3N_T0_DQS_EMCCLK_14 Sch=s

## LED
set_property -dict { PACKAGE_PIN H17    IOSTANDARD LVCMOS33
} [get_ports { a }]; IO_L18P_T2_A24_15 Sch=a
set_property -dict { PACKAGE_PIN K15    IOSTANDARD LVCMOS33
} [get_ports { b }]; IO_L24P_T3_RS1_15 Sch=b
```

4. Generate Bitstream
5. Program Device
6. Amati perilaku papan dengan *switch* SW0 dan SW1



Tabel Kebenaran

Multiplexer

s	a	b	c

Demultiplexer

s	c	a	b