A person wearing a plaid shirt

Description automatically generated with low confidence

Percobaan IV

SYNTHESIZABLE MIPS32® MICROPROCESSOR BAGIAN III : TOP LEVEL DESIGN DAN TESTBENCH

**Ahmad Aziz (13220034)**Asisten: Oktavian Putra Masyiakh (13219039)  
Tanggal Percobaan: 25/11/2022  
EL3111 Praktikum Arsitektur Sistem Komputer  
Laboratorium Sinyal dan Sistem – Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung

*Abstrak*— Pada praktikum modul 5 yaitu SYNTHESIZABLE MIPS32® MICROPROCESSOR BAGIAN III : TOP LEVEL DESIGN DAN TESTBENCH, terdapat 2 sub topik yang menjadi bahasan yaitu perancangan top level design dan penggunaan testbench. Praktikum ini bertujuan untuk memahami arsitektur mikroprosesor MIPS32® beserta datapath eksekusinya, memahami instruction set dari MIPS32® dan dapat membuat program sederhana dalam bahasa assembly yang dapat dieksekusi pada MIPS32®, melakukan simulasi eksekusi program MIPS32® pada program simulasi SPIM dan memahami cara setiap instruksi dieksekusi, menggabungkan komponen-komponen desain yang telah dibuat dari praktikum sebelumnya dalam kode VHDL untuk membuat top level design dari mikroprosesor Single-Cycle MIPS32® yang synthesizable dan dapat disimulasikan dengan Altera® Quartus® II v22, serta dapat membuat testbench untuk menguji desain mikroprosesor Single-Cycle MIPS32® dalam kode VHDL dan dapat disimulasikan dengan Altera® Quartus® II v22.

*Kata Kunci*—*ALU*, *control unit*, *counter, adder*.

# Pendahuluan

P

ada praktikum modul 5 yaitu SYNTHESIZABLE MIPS32® MICROPROCESSOR BAGIAN III : TOP LEVEL DESIGN DAN TESTBENCH dilakukan sebanyak 2 percobaan yang bertujuan diantaranya sebagai berikut:

* Praktikan memahami arsitektur mikroprosesor MIPS32® beserta datapath eksekusinya.
* Praktikan memahami instruction set dari MIPS32® dan dapat membuat program sederhana dalam bahasa assembly yang dapat dieksekusi pada MIPS32®.
* Praktikan dapat melakukan simulasi eksekusi program MIPS32® pada program simulasi SPIM dan memahami cara setiap instruksi dieksekusi.
* Praktikan dapat menggabungkan komponen-komponen desain yang telah dibuat dari praktikum sebelumnya dalam kode VHDL untuk membuat top level design dari mikroprosesor Single-Cycle MIPS32® yang synthesizable dan dapat disimulasikan dengan Altera® Quartus® II v9.1sp2.
* Praktikan dapat membuat testbench untuk menguji desain mikroprosesor Single-Cycle MIPS32® dalam kode VHDL dan dapat disimulasikan dengan Altera® Quartus® II v9.1sp2.

Praktikum pada modul ini menggunakan software Quartus v22 untuk membuat percobaan arsitektur dan melakukan simulasi. Ada beberapa topik yang dibahas pada modul praktikum ini yaitu sebagai berikut:

1. Implementasi Top-Level Design MIPS32®.
2. Pengujian Menggunakan TestBench.

Dalam melakukan percobaan dan analisis pada praktikum modul ini, perangkat lunak dan alat yang digunakan adalah sebagai berikut:

1. Quartus v22
2. PCSpim
3. Code editor Visual Studio Code

# Landasan Teoretis

## Datapath dan Contro

Dalam desain mikroprosesor Single-Cycle MIPS32®, rangkaian digital dapat dikelompokkan menjadi dua macam, yaitu datapath (jalur data) dan control (kontrol). Datapath merupakan komponen dari mikroprosesor yang melakukan operasi aritmetik serta melakukan penyimpanan data. Dalam datapath pula kelima tahap pemrosesan instruksi meliputi instruction fetch, instruction decode, execute, memory access, dan write back dilaksanakan. Sedangkan control merupakan komponen dari mikroprosesor yang mengatur datapath berdasarkan instruksi yang sedang dieksekusi.

Bagian kontrol telah direpresentasikan oleh control unit yang telah kita desain pada praktikum sebelumnya. Untuk bagian datapath, kita perlu menggabungkan komponen-komponen yang telah kita buat meliputi multiplexer, ALU, register, data memory, instruction memory, dan sebagainya untuk membentuk sebuah jalur yang dapat dilewati oleh data. Dengan demikian, control dan datapath tidak dapat dipisahkan dalam desain sebuah mikroprosesor. Datapath dan control akan membentuk sebuah desain mikroprosesor Single-Cycle MIPS32® yang disebut sebuah top-level design (desain paling atas). Top-level design pada umumnya hanya berisi port mapping dari satu komponen ke komponen lain.

Diagram, schematic

Description automatically generated

## Menggunakan Testbench untuk Simulasi

Dalam melakukan simulasi sebuah rangkaian digital, biasanya kita memberikan sinyal input secara manual melalui sebuah waveform editor. Cara ini boleh dibilang sederhana namun akan sangat tidak efektif apabila kita melakukan simulasi secara berulang-ulang. Cara lain untuk melakukan simulasi suatu rangkaian digital tanpa memberikan input satu per satu menggunakan waveform editor adalah menggunakan testbench.

Diagram

Description automatically generated

Pada dasarnya, testbench terdiri atas kode VHDL atau Verilog HDL, tergantung pada implementasi. Testbench sendiri dapat berisi sebuah desain yang menyimpan nilai-nilai sinyal input yang harus diberikan kepada desain yang sedang diuji (design under test). Kemudian, testbench ini akan mengeluarkan masing-masing input yang harus diberikan kepada desain yang sedang diuji berdasarkan suatu trigger, misalnya clock.

# Hasil dan Analisis

Setelah melakukan percobaan pada semua tugas didapatkan hasil sebagai brerikut:

## Tugas 1: Perancangan Program Counter

.

## Tugas 2: Perancangan Left Shifter Dua Kali

.

# Simpulan

* Control unit mendapatkan input dari function dan opcode untuk menentukan outputnya yang akan digunakan untuk melakukan eksekusi perintah yang akan dijalankan.
* Control unit memiliki nilai yang sudah didefinisikan, dan akan bernilai 0 jika tidak terdapat dalam nilai terdefinisi yang akan dianggap sebagai jeda dalam siklusnya.
* Adder dengan jenis CLA dapat diimplementasikan dengan full adder dengan menambahkan sinyal carry, propagate dan generate dalam sistem secara parallel dalam setiap komponen full addernya.
* CLA dapat melakukan operasi penjumlahan lebih cepa dengan RCA karena adanya sinyal propagate untuk carry setiap addernya sehingga tidak perlu menunggu hasil operasi dari masing-masing full addernya.
* Operasi pengurangan dapat dilakukan dengan komponen adder yaitu dengan mengubah operan kedua menjadi negasinya atau dikalikan dengan minus 1.
* Dalam VHDL, sinyal dalam bentuk vector dapat dimanfaatkan layaknya array dalam bahasa tingkat tinggi sebagai penyiimpan variable dalam proses pengolahan sinyal yang kemudian dipecah setiap bitnya sesuai dengan format yang ditentukan sehingga dapat menghemat memory dengan program yang ringkas.

Referensi

1. Bryant, Randal, dan David O’Hallaron. *Computer Systems: A Programmer’s Perspective 2nd Edition*. 2011. Massachusetts: Pearson Education Inc.
2. Patterson, David, dan John Hennessy. *Computer Organization and Design: The Hardware/Software Interface*. 2012. Waltham: Elsevier Inc*.*
3. *Kernighan, Brian, dan Dennis Ritchie. The C Programming Language 2nd edition. 1988. Englewood Cliffs : Prentice Hall.*

**Lampiran**

### Source code dan simulasi untuk tugas 1

program\_counter.vhd

|  |
| --- |
| -- Praktikum EL3111 Arsitektur Sistem Komputer  -- Modul : 4  -- Percobaan : 1  -- Tanggal : 11 November 2022  -- Kelompok : 10  -- Rombongan : B  -- Nama (NIM) 1 : Ahmad Aziz (13220034)  -- Nama (NIM) 2 :  -- Nama File : program\_counter.vhd  -- Deskripsi : Program Counter  **LIBRARY** IEEE**;**  **USE** IEEE**.**STD\_LOGIC\_1164**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_ARITH**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_UNSIGNED**.ALL;**  **LIBRARY** altera\_mf**;**  **USE** altera\_mf**.**altera\_mf\_components**.ALL;**  **ENTITY** program\_counter **IS**  **PORT** **(**  clk **:** **IN** STD\_LOGIC**;**  PC\_in **:** **IN** STD\_LOGIC\_VECTOR **(**31 **DOWNTO** 0**);**  PC\_out **:** **OUT** STD\_LOGIC\_VECTOR **(**31 **DOWNTO** 0**)**  **);**  **END** program\_counter**;**  **ARCHITECTURE** behavior **OF** program\_counter **IS**  **BEGIN**  **PROCESS** **(**clk**,**PC\_in**)**  **BEGIN**  **IF** **(rising\_edge(**clk**))** **THEN**  PC\_out **<=** PC\_in**;**  **END** **IF;**  **END** **PROCESS;**  **END** behavior**;** |

Tugas 1: Functional

Graphical user interface, application, table

Description automatically generated with medium confidence

Tugas 1: Timing

Graphical user interface, table

Description automatically generated

### Source code untuk tugas 2

lshift\_26\_28.vhd

|  |
| --- |
| -- Praktikum EL3111 Arsitektur Sistem Komputer  -- Modul : 4  -- Percobaan : 2b  -- Tanggal : 11 November 2022  -- Kelompok : 10  -- Rombongan : B  -- Nama (NIM) 1 : Ahmad Aziz (13220034)  -- Nama (NIM) 2 :  -- Nama File : lshift\_26\_28.vhd  -- Deskripsi : Program untuk melakukan operasi shift left pada 26 bit dengan output 28 bit  **LIBRARY** IEEE**;**  **USE** IEEE**.**STD\_LOGIC\_1164**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_ARITH**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_UNSIGNED**.ALL;**  **LIBRARY** altera\_mf**;**  **USE** altera\_mf**.**altera\_mf\_components**.ALL;**  **ENTITY** lshift\_26\_28 **IS**  **PORT** **(**  D\_IN **:** **IN** STD\_LOGIC\_VECTOR **(**25 **DOWNTO** 0**);** -- Input 32-bit;  D\_OUT **:** **OUT** STD\_LOGIC\_VECTOR **(**27 **DOWNTO** 0**)** -- Output 32-bit;  **);**  **END** lshift\_26\_28**;**  **ARCHITECTURE** behavior **OF** lshift\_26\_28 **IS**  **BEGIN**  **PROCESS** **(**D\_IN**)**  **BEGIN**  D\_OUT**(**27 **DOWNTO** 2**)** **<=** D\_IN**(**25 **DOWNTO** 0**);**  D\_OUT**(**1 **DOWNTO** 0**)** **<=** "00"**;**  **END** **PROCESS;**  **END** behavior**;** |

lshift\_32\_32.vhd

|  |
| --- |
| -- Praktikum EL3111 Arsitektur Sistem Komputer  -- Modul : 4  -- Percobaan : 2a  -- Tanggal : 11 November 2022  -- Kelompok : 10  -- Rombongan : B  -- Nama (NIM) 1 : Ahmad Aziz (13220034)  -- Nama (NIM) 2 :  -- Nama File : lshift\_32\_32.vhd  -- Deskripsi : Program untuk melakukan operasi shift left 32 bit  **LIBRARY** IEEE**;**  **USE** IEEE**.**STD\_LOGIC\_1164**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_ARITH**.ALL;**  **USE** IEEE**.**STD\_LOGIC\_UNSIGNED**.ALL;**  **LIBRARY** altera\_mf**;**  **USE** altera\_mf**.**altera\_mf\_components**.ALL;**  **ENTITY** lshift\_32\_32 **IS**  **PORT** **(**  D\_IN **:** **IN** STD\_LOGIC\_VECTOR **(**31 **DOWNTO** 0**);** -- Input 32-bit;  D\_OUT **:** **OUT** STD\_LOGIC\_VECTOR **(**31 **DOWNTO** 0**)** -- Output 32-bit;  **);**  **END** lshift\_32\_32**;**  **ARCHITECTURE** behavior **OF** lshift\_32\_32 **IS**  **BEGIN**  **PROCESS** **(**D\_IN**)**  **BEGIN**  D\_OUT**(**31 **DOWNTO** 2**)** **<=** D\_IN**(**29 **DOWNTO** 0**);**  D\_OUT**(**1 **DOWNTO** 0**)** **<=** "00"**;**  **END** **PROCESS;**  **END** behavior**;** |