**REPORT**

**Design and Implementation of a Simple Processor**

Ver 1.0

09/01/2022

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Full name** | **Function** | **Student Code** | **Date** |
| Written by | Lý Minh Tú  Cao Kỳ Nam | Nhóm trưởng | 19020648  19020581 | 09/01/2022  09/01/2022 |
| Verified by | TS. Nguyễn Kiêm Hùng | Giảng viên |  |  |
| Approved by | TS. Nguyễn Kiêm Hùng | Giảng viên |  |  |

|  |
| --- |
| **Abstract (from 5 to 10 lines)** |
| This report presents the basic process of creating a general-purpose processor that supports the given set of instructions |

|  |
| --- |
| **Keywords** |
| FSM, FSMD, dapath, 16bit microprocessor |

|  |
| --- |
| **Work context** |
| **Lưu ý: Nghiêm cấm mọi hình thực copy bài (bao gồm cả report và mã nguồn VHDL) của nhau. Nếu phát hiện sự giống nhau giữa 2 bài thì tùy mức độ mà có thể sẽ bị trừ điểm hoặc chia lấy điểm trung bình làm điểm của project.**  **Report cần được in ra và nộp cho GV (phòng 2.1, E4) muộn nhất trước ngày thi 10/1/2021. Đồng thời file phần mềm của Report và mã VHDL cần được gửi vào hòm thư** [**hungnvnu@gmail.com**](mailto:hungnvnu@gmail.com)(nên share qua Google Drive)**. Ngày thi, SV cần mang máy tính laptop theo để chạy mô phỏng kiểm chứng!**  **Gợi ý: trong thư mục *simp\_cpu.rar* chứa các file VHDL mẫu mô tả cpu và cpu\_tb cùng với các khối chính tạo nên CPU. Riêng memory đã được hoàn thiện cả phần mô tả thiết kế và testbench như một ví dụ mẫu. SV có thể sử dụng các template để phát triển code của riêng mình.** |

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Time** | **Revised by** | **Description** |
| V1.0 | 15/11/2021 | Nguyễn Kiêm Hùng | Original Version |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Document History 3](#_Toc6297410)

[Table of Contents 4](#_Toc6297411)

[1. Introduction 5](#_Toc6297412)

[2. Requirements 5](#_Toc6297413)

[3. Architecture Design 7](#_Toc6297414)

[3.1. FSMD 7](#_Toc6297415)

[3.2. Datapath architecture 8](#_Toc6297416)

[3.3. Controller 10](#_Toc6297417)

[4. Modeling 12](#_Toc6297418)

[5. Simulation and Synthesis 12](#_Toc6297419)

[Appendix A: VHDL Code 15](#_Toc6297420)

# Introduction

*(Introduction to the motivation, Objectives, and main Contents of the project)*

**Objective:** Vận dụng các kiến thức, kỹ năng đã được học đểthiết kế, mô phỏng và thực thi một bộ vi xử lý 16-bit đơn giản.

# Requirements

* Bộ vi xử lý hỗ trợ 16 lệnh cơ bản như được liệt kê trong Bảng 1;

Bảng 1: Cấu trúc tập lệnh.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TT** | **Assembly Instruction** | **First Byte** | | **Second Byte** | | **Operation** |
| Opcode | Operand1 | Operand2 | |
| 1 | MOV Rn, direct | 0000 | Rn | direct | | Rn = M(direct) |
| 2 | MOV direct, Rn | 0001 | Rn | direct | | M(direct) = Rn |
| 3 | MOV Rn, @Rm | 0010 | Rn | Rm |  | M(Rm) = Rn |
| 4 | MOV Rn, #immed | 0011 | Rn | immediate | | Rn = immediate |
| 5 | ADD Rn, Rm | 0100 | Rn | Rm |  | Rn = Rn + Rm |
| 6 | SUB Rn, Rm | 0101 | Rn | Rm |  | Rn = Rn - Rm |
| 7 | JZ Rn, Addr | 0110 | Rn | Addr | | PC = Addr only if Rn = 0 |
| 8 | OR Rn, Rm | 0111 | Rn | Rm |  | Rn = Rn OR Rm |
| 9 | AND Rn, Rm | 1000 | Rn | Rm |  | Rn = Rn AND Rm |
| 10 | JMP Addr | 1010 | Rn | Adrr | | PC = Addr |

* Cấu trúc của bộ vi xử lý bao gồm các khối chức năng như trong Hình 2, trong đó:
  + Thanh ghi PC (Program Counter): 16-bit, dùng để chứa địa chỉ của lệnh tiếp theo mà bộ vi xử lý sẽ thực hiện
  + Thanh ghi IR (Instruction Register): 16-bit, dùng để chưa lệnh mà vi xử lý sẽ thực hiện
  + Tệp thanh ghi RF (Register File): 16×16 bit, dùng lưu dữ liệu trong quá trình tính toán của ALU
  + ALU (arithmetic and logic unit): hỗ trợ các phép tính trên dữ liệu 16-bit
* Bộ nhớ Memory: 64K×16 bit, dùng để lưu chương trình và dữ liệu cho bộ vi xử lý



Hình 1: Cấu trúc các khối chức năng cơ bản của bộ vi xử lý.

# Architecture Design

## FSMD



Hình 2: Máy trạng thái FSMD.

## Datapath architecture

Hình 3: Cấu trúc Datapath.



Hình 5: Các phép tính được hỗ trợ bởi ALU.

|  |  |
| --- | --- |
| **ALUs** | **ALUr** |
| 00 | OPr1 + OPr2 |
| 01 | OPr1 – Opr2 |
| 10 | OPr1 OR OPr2 |
| 11 | OPr1 AND OPr2 |



Hình 6: Tệp thanh ghi RF.

## Controller



Hình 4: Giao diện ghép nối vào/ra của controller.



Hình 5: Mô hình máy trạng thái FSM của bộ điều khiển.

# Simple System-on-Chip



Hình 6. Thiết kế một hệ thống SoC đơn giản sử dụng CPU phần trên.

# Modeling

*(Viết mã nguồn VHDL của project ở đây hoặc trong Appendix A: VHDL Code.)*



Hình 7. Tổ chức của các tệp VHDL.

Tất cả mã nguồn VHDL của project [ở đây](https://drive.google.com/drive/u/1/folders/1nT5EZ9khqsWWiQD2wAVWSDzpiifKbt6J).

# Simulation and Synthesis

Machine code for initializing program memory:

X"0712", -- MOV1: Mov R7,18 => R7 = M(18)

X"1716", -- MOV2: Mov 22,R7 => M(22) = R7

X"0813", -- MOV1: Mov R8,19 => R8 = M(19)

X"2560", -- MOV3: Mov R5,@R7 => M(R7) = R5

X"3513", -- MOV4: Mov R5,#19 => R5 = 19

X"3940", -- MOV4: Mov R9,#64 => R9 = 64

X"3401", -- MOV4: Mov R4,#1 => R4 = 1

X"3602", -- MOV4: Mov R6,#s => R6 = 2

X"4750", -- ADD: ADD R7,R5 => R7 = R7 + R5

X"5750", -- SUB: SUB R7,R5 => R7 = R7 - R5

X"7640", -- OR: OR R6,R4 => R6 = R6 OR R4 : 0010 OR 0001=0011

X"8470", -- AND: AND R4,R7 => R4 = R4 AND R7 : 0001 AND 1010

X"6501" -- JZ: JZ R5,1

## Kết quả mô phỏng

*(Chỉ ra kết quả mô phỏng bằng Modelsim và diễn giải ý nghĩa của các kết quả đó ở đây)*

Tại cửa sổ Transcript của ModelSIM lần lượt gõ các lệnh run như sau:

* VSIM> run 25

để kiểm tra hoạt động của tín hiệu Reset

* VSIM> run 45

để kiểm tra hoạt động của lệnh X"0712"

MOV1: Mov R7,18 => R7 = M(18) = A = 10

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"1716"

MOV2: Mov 22,R7 => M(22) = R7

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"0813"

MOV1: Mov R8,19 => R8 = M(19) = B = 11

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"2560"

MOV3: Mov R5,@R7 => M(R7) = R5

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"3513"

MOV4: Mov R5,#19 => R5 = 19

* VSIM> run 100

để kiểm tra hoạt động của lệnh X"3940"

MOV4: Mov R9,#64 => R9 = 64

* VSIM> run 100

để kiểm tra hoạt động của lệnh X"3401"

MOV4: Mov R4,#1 => R4 = 1

* VSIM> run 100

để kiểm tra hoạt động của lệnh X"3602"

MOV4: Mov R6,#s => R6 = 2

* VSIM> run 100

để kiểm tra hoạt động của lệnh X"4750"

ADD: ADD R7,R5 => R7 = R7 + R5

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"5750"

SUB: SUB R7,R5 => R7 = R7 - R5

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"7640"

OR: OR R6,R4 => R6 = R6 OR R4 : 0010 OR 0001=0011

* VSIM> run 120

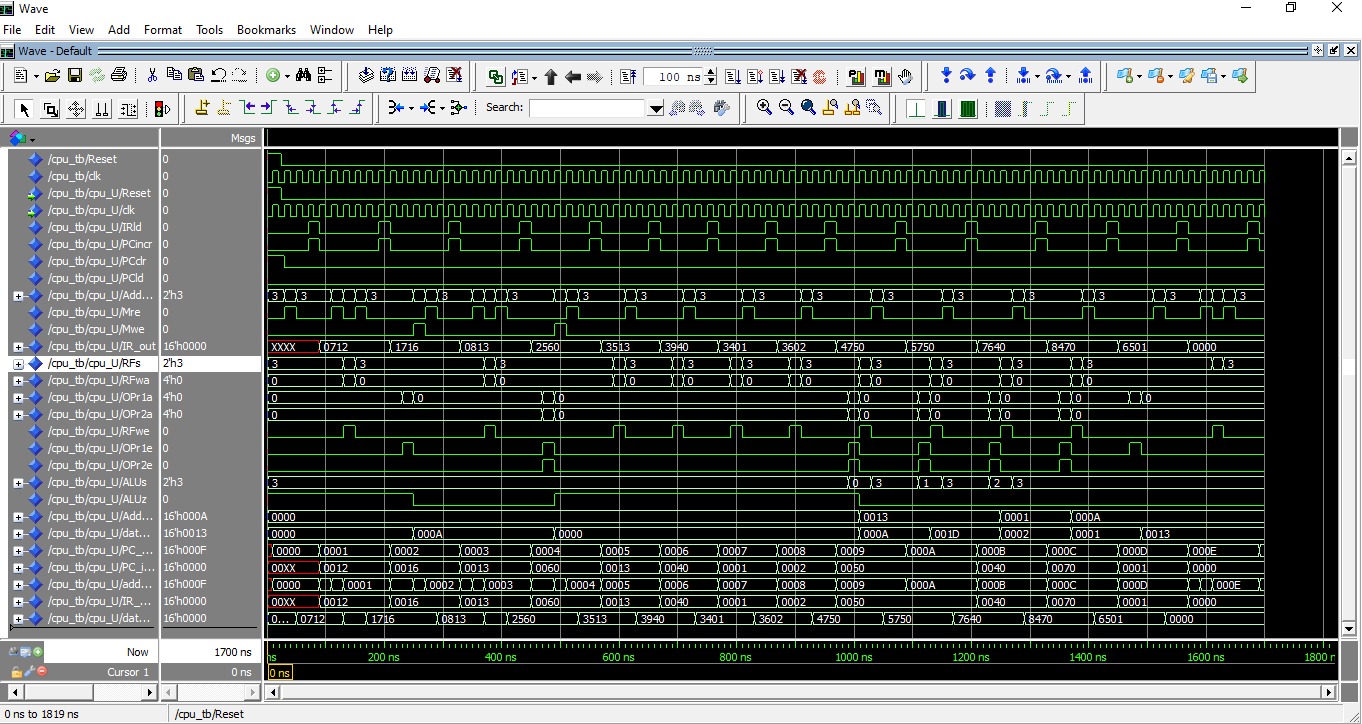
để kiểm tra hoạt động của lệnh X"8470"

AND: AND R4,R7 => R4 = R4 AND R7 : 0001 AND 1010

* VSIM> run 120

để kiểm tra hoạt động của lệnh X"6501"

JZ: JZ R5,1



Hình 8: Kết quả mô phỏng.

Appendix A: VHDL Code

Tất cả mã nguồn VHDL của project [ở đây](https://drive.google.com/drive/u/1/folders/1nT5EZ9khqsWWiQD2wAVWSDzpiifKbt6J).