Logo, company name

Description automatically generated

**Department of IT and Computer Science**

**Pak-Austria Fachhochschule: Institute of Applied Sciences and Technology, Haripur, Pakistan**

**COMP-261L Computer Organization and Assembly Language**

**Final Project Report**

**Group 1**

|  |  |  |
| --- | --- | --- |
| **Muhammad Suleman**  **B20F0012CS035** | **Yaseen Ejaz**  **B20F0283CS014** | **Muhammad Danish**  **B20F0375CS030** |

|  |  |
| --- | --- |
| **Submitted to:** Dr. Hashim Ali | **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Instructor Signature** |

Table of Contents

[**Section 1: Introduction** 3](#_Toc92920778)

[**Section 1.1: Objective:** 4](#_Toc92920779)

[**Section 2: Problem Statement** 4](#_Toc92920780)

[**Section 3: Components** 4](#_Toc92920781)

[**Section 3.1: Register File** 5](#_Toc92920782)

[**Section 3.2: Data Memory** 5](#_Toc92920783)

[**Section 3.3: Instruction Memory** 6](#_Toc92920784)

[**Section 3.4: Program Counter** 6](#_Toc92920785)

[**Section 3.5: ALU (4 Bit)** 6](#_Toc92920786)

[**Section 4: Controller** 7](#_Toc92920787)

[**Section 4.1: Truth Table** 7](#_Toc92920788)

[**Section 4.2: K-Map Simplification** 7](#_Toc92920789)

[**Section 4.2.1: A1** 7](#_Toc92920790)

[**Section 4.2.2: A0** 7](#_Toc92920791)

[**Section 4.2.3: A/S** 8](#_Toc92920792)

[**Section 4.2.4: Cin** 8](#_Toc92920793)

[**Section 4.2.5: RWR** 8](#_Toc92920794)

[**Section 4.2.6: MRD** 9](#_Toc92920795)

[**Section 4.2.7: MWR** 9](#_Toc92920796)

[**Section 4.2.8: RegD** 9](#_Toc92920797)

[**Section 4.2.9: MemToReg** 10](#_Toc92920798)

[**Section 4.2.10: ALUSRC** 10](#_Toc92920799)

[**Section 4.2.11: BR** 10](#_Toc92920800)

[**Section 4.2.12: J** 10](#_Toc92920801)

[**Section 4.3: Circuit** 11](#_Toc92920802)

[**Section 5: Final Microprocessor** 12](#_Toc92920803)

[**Section 6: Testing** 13](#_Toc92920804)

[**Section 7: Step by Step** 17](#_Toc92920805)

[**Section 8: Conversion to HEX** 19](#_Toc92920806)

[**Section 9: Conclusion** 19](#_Toc92920807)

**Microprocessor Design**

# **Section 1: Introduction**

This document is a project report for a simple MIPS microprocessor architecture. The project's purpose is to develop a 4-bit processor at the logic level, then replicate it at the layout level. With a custom-defined instruction set, this document explains the fundamental concepts of microprocessor design in the simplest way possible. Logisim was used to create the project.

## **Section 1.1: Objective:**

The purpose of this project is to recreate all the basic building elements of a RISC-based microprocessor so that a complete data-path and control unit can be designed. Later, in order to test the processor's operation, a basic program must be written to test the operation of components as well as the microprocessor itself.

# **Section 2: Problem Statement**

Build a Processor using all the concepts that you have learnt in this course. The processor must have all the necessary components required to run a simple C code. The code that you must run and test on your microprocessor is as follows:

A ← 0

B ← 0

C ← 0

A ← A + 7

B ← B – 8

C ← DMEM[2] ← 0

IF ( C ) THEN C ← A + B ELSE {B ← B | 7; A ← B & 7;}

C ← A + B

DMEM[15] ← C

# **Section 3: Components**

Following are the components we need to make our Microprocessor.

## **Section 3.1: Register File**

Diagram, schematic

Description automatically generated

## **Section 3.2: Data Memory**

Diagram, schematic

Description automatically generated

## **Section 3.3: Instruction Memory**

A picture containing diagram

Description automatically generated

## **Section 3.4: Program Counter**

Diagram

Description automatically generated

## **Section 3.5: ALU (4 Bit)**

Diagram, schematic

Description automatically generated

# **Section 4: Controller**

## **Section 4.1: Truth Table**

Following is the Controller Input/Output table.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **FUNC** | **Type** | **Input** | **Output** | | | | | | | | | | |
| **Op** | **A1A0** | **A/S** | **Cin** | **RWR** | **MRD** | **MWR** | **RegD(LW)** | **MtoRegSW** | **aluSrc** | **Br** | **J** |
| AND | R | 0000 | 00 | x | x | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| OR | R | 0001 | 01 | x | x | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| ADD | R | 0010 | 10 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| SUB | R | 0011 | 10 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| ANDi | I | 0100 | 00 | x | x | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| ORi | I | 0101 | 01 | x | x | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| ADDi | I | 0110 | 10 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| SUBi | I | 0111 | 10 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| SLT | R | 1000 | 11 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| SLTi | I | 1001 | 11 | 1 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| BEQ | I | 1010 | 10 | 1 | 1 | 1 | x | x | x | x | 0 | 1 | 0 |
| BNE | I | 1011 | 10 | 1 | 1 | 1 | x | x | x | x | 0 | 1 | 0 |
| J | J | 1100 | xx | x | x | x | x | x | x | x | x | 0 | 1 |
| Unuse | --- | 1101 | xx | x | x | x | x | x | x | x | x | x | x |
| LW | I | 1110 | 10 | 0 | 0 | 0 | 0 | 1 | **0** | **1** | 1 | 0 | 0 |
| SW | I | 1111 | 10 | 0 | 0 | 1 | 1 | 0 | **x** | **0** | 1 | 0 | 0 |

Note: RWR, MRD, and MWR are active low.

## 

## **Section 4.2: K-Map Simplification**

### **Section 4.2.1: A1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 1 | 1 |
| 01 | 0 | 0 | 1 | 1 |
| 11 | x | x | 1 | 1 |
| 10 | 1 | 1 | 1 | 1 |

Simplified Equation: A + C

### **Section 4.2.2: A0**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 0 | 0 |
| 01 | 0 | 1 | 0 | 0 |
| 11 | x | x | 0 | 0 |
| 10 | 1 | 1 | 0 | 0 |

Simplified Equation: C’D + AC’

### **Section 4.2.3: A/S**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | x | x | 1 | 0 |
| 01 | x | x | 1 | 0 |
| 11 | x | x | 0 | 0 |
| 10 | 1 | 1 | 1 | 1 |

Simplified Equation: AC’ + A’D

### **Section 4.2.4: Cin**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | x | x | 1 | 0 |
| 01 | x | x | 1 | 0 |
| 11 | x | x | 0 | 0 |
| 10 | 1 | 1 | 1 | 1 |

Simplified Equation: AC’ + A’D

### **Section 4.2.5: RWR**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | x | x | 1 | 0 |
| 10 | 0 | 0 | 1 | 1 |

AB’C + ACD

Simplified Equation: AC(B’+D)

### **Section 4.2.6: MRD**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 1 | 1 | 1 |
| 11 | x | x | 1 | 0 |
| 10 | 1 | 1 | x | x |

Simplified Equation: A’ + B’ + D

### **Section 4.2.7: MWR**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 1 | 1 | 1 | 1 |
| 11 | x | x | 0 | 1 |
| 10 | 1 | 1 | x | x |

A’ + B’ + D’

Simplified Equation: (ABD)’

### **Section 4.2.8: RegD**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 1 | 1 | 1 | 1 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | x | x | x | 0 |
| 10 | 1 | 0 | x | x |

A’B’ + B’C’D’

Simplified Equation: B’(A’ + C’D’)

### **Section 4.2.9: MemToReg**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | x | x | 0 | 1 |
| 10 | 0 | 0 | x | x |

Simplified Equation: ABD’

### **Section 4.2.10: ALUSRC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 1 | 1 | 1 | 1 |
| 11 | x | x | 1 | 1 |
| 10 | 0 | 1 | 0 | 0 |

Simplified Equation: B + AC’D

### **Section 4.2.11: BR**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | 0 | x | 0 | 0 |
| 10 | 0 | 0 | 1 | 1 |

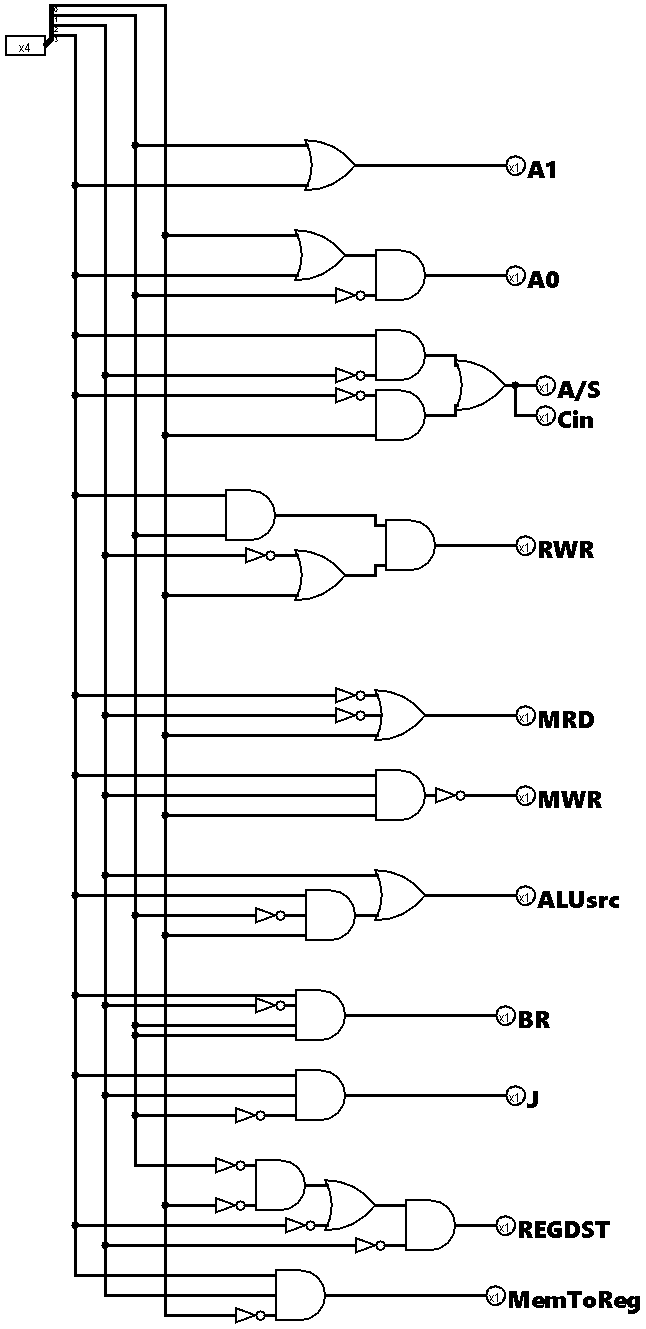
Simplified Equation: AB’CD

### **Section 4.2.12: J**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB/CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 0 | 0 | 0 |
| 01 | 0 | 0 | 0 | 0 |
| 11 | 1 | x | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 |

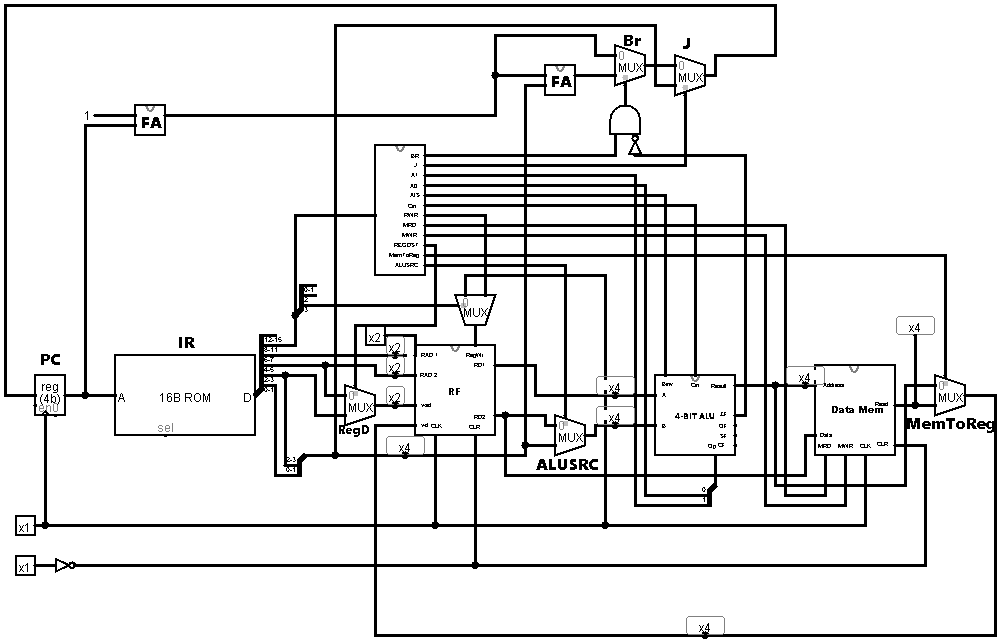
Simplified Equation: ABC’

## **Section 4.3: Circuit**



# **Section 5: Final Microprocessor**

Integrating all the components together, we have designed the following Microprocessor:



# **Section 6: Testing**

Now, we will test the microprocessor that we designed, using the code provided in the problem statement.

R0=A, R1=B, R2=C, R3=D

**PC = 0x0000**

ADDi R0,R0,7

0000 0110 00 00 0111

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Immediate** |
| 0000 | 0110 | 00 | 00 | 0111 |

Rs = R0

Rt = R0

Immediate = 0x0111 or (7)10

**PC = 0x0001**

SUBi R1,R1,8

0000 0111 01 01 1000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Immediate** |
| 0000 | 0111 | 01 | 01 | 1000 |

Rs = R1

Rt = R1

Immediate = 0x1000 or (8)10

**PC = 0x0002**

LW R2, 2(R3)

0000 1110 11 10 0010

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Offset** |
| 0000 | 1110 | 11 | 10 | 0010 |

Rs = R3

Rt = R2

Offset = 0x0010 or (2)10

**PC = 0x0003**

BNE R2, R3, 7

0000 1011 10 11 0111

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Address** |
| 0000 | 1011 | 10 | 11 | 0111 |

Rs = R2

Rt = R3

Address = 0x0111 or (7)10

**PC = 0x0004**

ORi R1, R1, 7

0000 0101 01 01 0111

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Immediate** |
| 0000 | 0101 | 01 | 01 | 0111 |

Rs = R2

Rt = R3

Immediate = 0x0111 or (7)10

**PC = 0x0005**

ANDi R0, R1, 7

0000 0100 01 00 0111

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Immediate** |
| 0000 | 0100 | 01 | 00 | 0111 |

Rs = R1

Rt = R0

Immediate = 0x0111 or (7)10

**PC = 0x0006**

J 0x1000

0000 1100 00 00 1000

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Address** |
| 0000 | 1100 | xx | xx | 1000 |

Address = 0x1000 or (8)10

**PC = 0x0007**

ADD R2, R0, R1

0000 0010 00 01 10 00

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Rd** | **Unused** |
| 0000 | 0010 | 00 | 01 | 10 | 00 |

Rs= R0

Rt= R1

Rd= R2

**PC = 0x0008**

ADD R2, R0, R1

0000 0010 00 01 10 00

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Rd** | **Unused** |
| 0000 | 0010 | 00 | 01 | 10 | 00 |

Rs= R0

Rt= R1

Rd= R2

**PC = 0x0009**

SW R2, 15(R3)

0000 1111 11 10 1111

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Unused** | **Opcode** | **Rs** | **Rt** | **Offset** |
| 0000 | 1100 | 11 | 10 | 1111 |

Rs = R3

Rt = R2

Offset = 0x1111 or (15)10

# **Section 7: Step by Step**

**ADDi R0, R0, 7:**

R0 = 0000 + 0111 = 0111

**SUBi R1,R1,8**

R1 = 0000 – 1000 = -(1000) = 1000

**LW R2, 2(R3)**

R2 = 2 + 0x0000 = 0x0002

Value in 0x0002 is 0000

R2 = 0000

**BNE R2, R3, 0x0111**

R2 = R3 = 0000

If (R2! = R3)

Branch

**ORi R1, R1, 7**

R1 = 1000

R1 = 1000 | 0111 = 1111

**ANDi R0, R1, 7**

R1 = 1111

R0 = 1111 & 0111 = 0111

**J 0x1000**

Unconditional Jump

***ADD R2, R0, R1***

*Branch Value*

**ADD R2, R0, R1**

R0 = 0111

R1 = 1111

R2 = 1111 + 0111 = 0110

**SW R2, 15(R3)**

R2 = 0110

R3 = 0000

Store 0110 in 15th address of Data Memory

# **Section 8: Conversion to HEX**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PC** | **Instruction** | **Binary** | **Hexadecimal** | **Execution** |
| 0x0000 | ADDi R0, R0, 7 | 0000 0110 0000 0111 | 0x0607 | R0 = 0111 |
| 0x0001 | SUBi R1,R1,8 | 0000 0111 0101 1000 | 0x0758 | R1 = 1000 |
| 0x0010 | LW R2, 2(R3) | 0000 1110 1110 0010 | 0x0EE2 | R2 = 0000 |
| 0x0011 | BNE R2, R3, 0x0111 | 0000 1011 1011 0111 | 0x0BB7 |  |
| 0x0100 | ORi R1, R1, 7 | 0000 0101 0101 0111 | 0x0557 | R1 = 1111 |
| 0x0101 | ANDi R0, R1, 7 | 0000 0100 0100 0111 | 0x0447 | R0 = 0111 |
| 0x0110 | J 0x1000 | 0000 1100 0000 1000 | 0x0C08 |  |
| 0x0111 | ADD R2, R0, R1 | 0000 0010 0001 1000 | 0x0218 | R2 = 0110 |
| 0x1000 | ADD R2, R0, R1 | 0000 0010 0001 1000 | 0x0218 | R2 = 0110 |
| 0x1001 | SW R2, 15(R3) | 0000 1111 1110 1111 | 0x0FEF | DMEM[15] = 0110 |

# **Section 9: Conclusion**

This project consists of the construction of a MIPS architecture microprocessor. The processor can perform operations like ADD, ADDi , SUB, SUBi, AND, ANDi, OR, ORi, LW, SW, J, BEQ, BNE. We can use codes to load in the Instruction Memory to perform the instructions. Many logics and operations have been applied to construct this microprocessor, and all the components like Program Counter, Instruction Memory, Register File, ALU, Data Memory, Multiplexers, and Controller were used and integrated together. Instead of going towards the shortcut of using a ROM as a controller, we went for the K-Map simplification approach to construct the controller of the processor. The project was run and tested by all the member of Group 1.