

SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTING

COMPUTER SCIENCE AND ENGINEERING PROGRAM

COMPUTER ARCHITECTURE AND ORGANIZATION PROJECT

SMALL PROCESSOR DESIGN

BY:

NAME ID GROUP

Adonay Yirga---------------------R\0101\08---------------------------------1

Amanuel Loeta-------------------R\0149\08---------------------------------1

Amanuel Mebrahtu--------------R\0150\08---------------------------------1

Bsrat Takelle----------------------R\0306\08---------------------------------2

Ashenafi Wada--------------------R\00135\07-------------------------------1

Atsbha Mebrahtu ----------------R\0197\08---------------------------------1

Amanuel Addisu -----------------R\00091\07--------------------------------1

Abeje Lambamo -----------------R\0045\08----------------------------------1

Instructor: Simegnew Y. Alaba

Date: Feb 12, 2018

Table of Contents

[1 Objective of the project 3](#_Toc506226878)

[2 THE SIMPLE CPU DESIGN AND METHODOLOGY 3](#_Toc506226879)

[2.1 Step 1: A simple ALU 4](#_Toc506226880)

[THE SUBCIRCUIT THAT BUILT-ON THE ALU ARE: 4](#_Toc506226881)

[2.2 Step 2: The CPU Data path 5](#_Toc506226882)

[2.3 Step 3: The control Unit 5](#_Toc506226883)

[2.4 Step 4: Simulation result and discussion (Putting it all together) 6](#_Toc506226884)

[3 CONCLUSION 7](#_Toc506226885)

[References 8](#_Toc506226886)

ABSTRACT

This paper describes a sequence of designs, each building upon the next, and leading to a working simulation of a simple 8-bit CPU (Central Processing Unit). The design features a classic Von Neumann architecture comprising a simple data path with a few registers, a simple ALU (Arithmetic Logic Unit. The first step involves the design of the ALU, which is capable of three basic operations. The second step is construction of a data path complete with several 8-bit registers. The third step involves the design and implementation of a control unit. All simulations are performed using a free and open source simulator called Logisim, which performs digital logic simulations with the ability to build larger circuits from smaller sub circui

Part-I

# Objective of the project

Computer architecture and organization form important parts of the computer science “body of knowledge”. A professional in any field of computing should not regard the computer as just black box that executes programs by magic. They need to understand computer architecture in order to make best use of the software tools and computer languages they use to create programs. One way to learn about this topic is to study existing CPU architectures. Many different CPU architectures abound, and different textbooks have selected different processor architectures as examples. Several commercial processors are used as an example to describe computer architecture and organization concepts.

A better way for students to construct processor is to use a basic logic simulator, which provides a simpler environment. It provides digital logic elements with basic digital design issues. In particular, Logisim provides a friendly educational tool for performing basic digital simulations. Logisim enables designing and simulating digital circuits ranging from small logic circuits up to entire CPUs. A recent comparison of logic simulators for computer design courses found Logisim to be a good option for educational use. Furthermore, Logisim is an open source tool that can be freely downloaded from the web.



Figure . Block diagram of a simple ALU

# THE SIMPLE CPU DESIGN AND METHODOLOGY

This paper will describe a series of designs using Logisim to provide a step-by-step design and simulation of a simple CPU. The steps are broken down into several parts, each building upon the next. These steps are described in the following subsections.

## Step 1: A simple ALU

The ALU is built with eight-bit adder, one bit-extender for subtracting, eight-bit multiplier, and basic logic gates(OR, XOR,NOR, controlled buffer) for its operation. It is an 8-bit ALU using the above-mentioned logic gates. The ALU have two 8-bit bus inputs (labelled a and b) and one 8-bit output as illustrated in Figure 2.2.The function of the ALU is selected using three ALU control lines (F0, F1 and F2) which select one of eight operations in the ALU as indicated in Table 2.1. The ALU is constructed in a straightforward manner by building the digital circuit for each operation and using a 1-of-8 multiplexer to select which appears at the output.

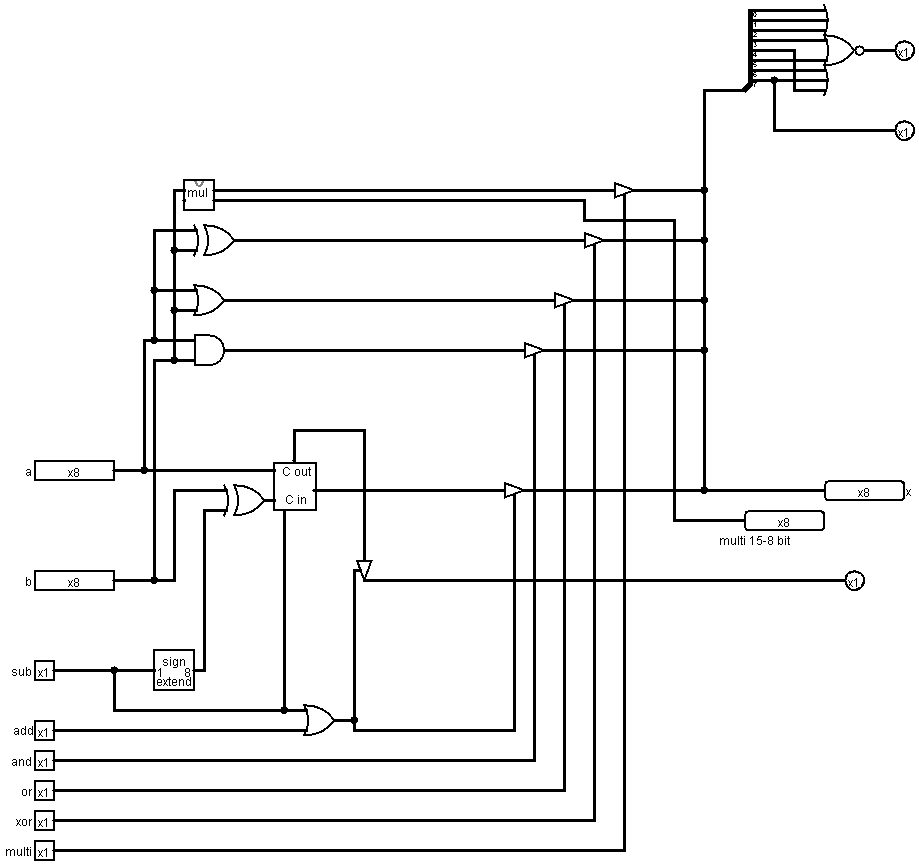


Figure . A simple ALU

### THE SUBCIRCUIT THAT BUILT-ON THE ALU ARE:

#### Eight-bit adder

The eight-bit adder is built from two four-bit adders, which are built from four full-adders individually.

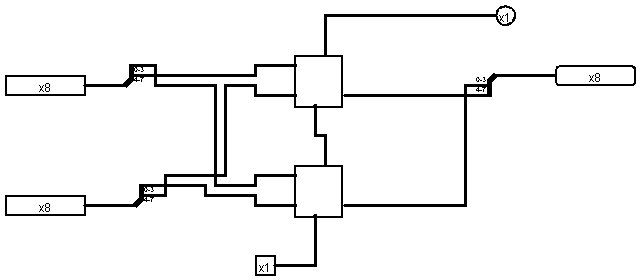


Figure . Eight-bit adder

#### Eight-bit multiplier

The eight-bit multiplier is built from eight one-bit multipliers, which are each built using full adder and right shifter.

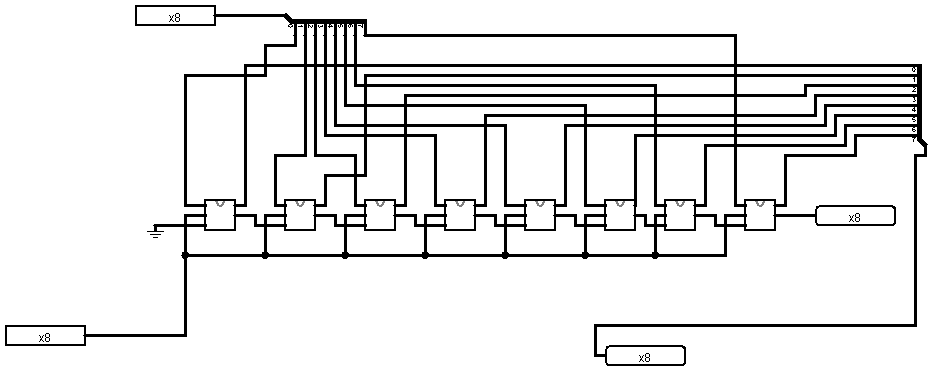


Figure . Eight-bit multiplier

## Step 2: The CPU Data path

The next step in the process is to take the ALU sub circuit built in the previous step and place it inside a data path using Logisim. The CPU includes MAR, MBR and data bus and address bus.

## Step 3: The control Unit

The manual control of the data path highlights the need to manage the control signals. The control lines help us with specifying which operation we are going to perform.

|  |  |
| --- | --- |
| Op codes | Output |
| 000 | SUB |
| 001 | ADD |
| 010 | AND |
| 011 | OR |
| 100 | XOR |
| 101 | MUL |
| 110 | LOAD |
| 111 | STORE |

Table . summary of simple ALU operations

## Step 4: Simulation result and discussion (Putting it all together)

Now it is time to put the ALU with the data paths and the control unit together. We can test using some instructions if the CPU works properly.

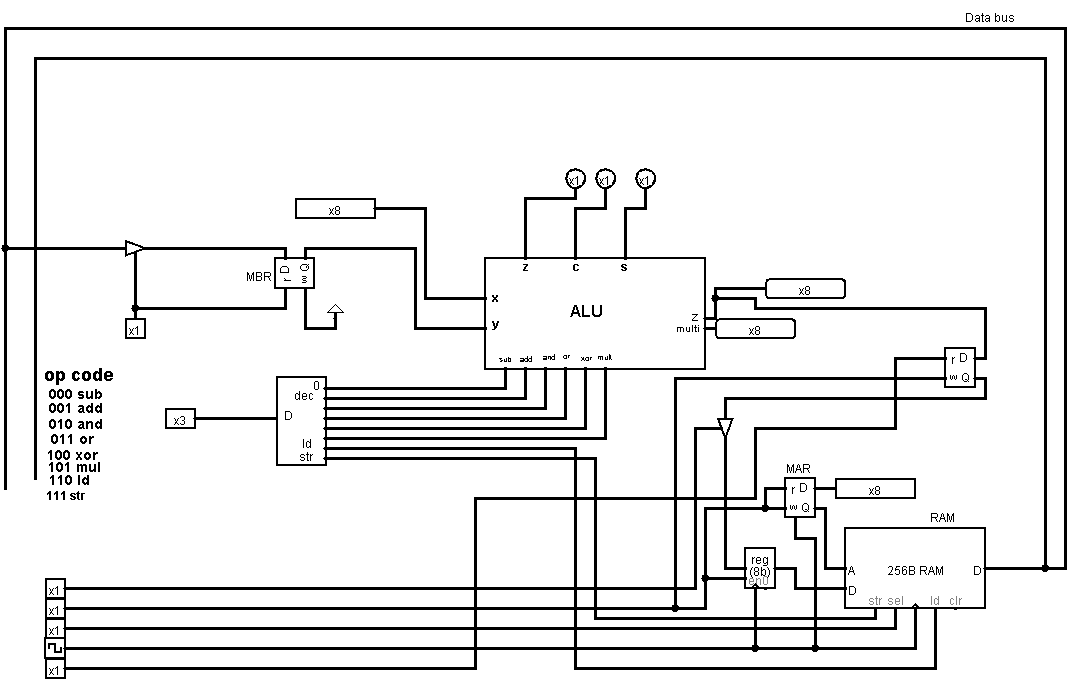


Figure . A simple working CPU

The following are the steps that are used to perform different operation throughout the CPU.

* First we need to enable read and write the MAR
* Then we enable the select line, to select the value in the RAM which are addressed by the MAR
* Then to load the data to the data bus, we perform an operation using the opcode 110
* Then we enable the controlled buffer and read enable to load the value in MBR
* In order the MBR not to change its value we disable the controlled buffer.
* Now, we can perform any of the ALUs operations.
* Finally, we can store back the operated value to the ram by STR operation.

# CONCLUSION

Having built our own simple CPU has understanding value, which helps the description of existing industrial CPU architectures. Logisim is a friendly and simple tool for performing digital logic simulations, yet it is powerful enough to simulate a basic CPU. This paper describes a sequence of step-by-step designs that gradually create simple CPU simulation.

# References

<https://www.youtube.com/dashboard?o=U>

Simegnew Y. Alaba, lecture 7 control unit

<https://www.youtube.com/crashcourse?o=U>