Objectives

- To know about different components of the CPU and how they work.
- To Learn how to build a CPU (Central Processing Unit) using “Logisim” software.
- To Learn how computers perform an instruction.
- To know the internal architecture of computers and how computers process data
- To Learn how to build a specific bit CPU and program the computer

Introduction

In the domain of computer architecture, crafting a streamlined CPU with essential components serves as a pivotal exercise in comprehending the inner workings of processors. Our project focused on the design of an efficient 22-bit CPU, incorporating crucial elements such as the Memory Address Register (MAR), Memory Buffer Register (MBR), Instruction Register (IR), Program Counter (PC), Accumulator, Arithmetic Logic Unit (ALU), Control Unit (CU), and Main Memory. This report presents a detailed account of the development, implementation, and validation of this minimalist CPU.

Component Overview:

1. **Memory Address Register (MAR):** Responsible for storing memory addresses for data or instruction retrieval and storage. The MAR is equipped with 5-bit capacity.
2. **Memory Buffer Register (MBR):** Temporarily holds data retrieved from or destined for memory, boasting a width of 22 bits.
3. **Instruction Register (IR):** Captures the current instruction fetched from memory, utilizing a concise 4-bit storage space.
4. **Program Counter (PC):** Tracks the memory address of the next instruction to be fetched, facilitated by 5-bit register.
5. **Accumulator (Ac):** Acts as the repository for arithmetic and logical operation results, featuring a robust 22-bit architecture.

6. **Arithmetic Logic Unit (ALU):** Utilizes 22-bit configuration to execute a diverse range of arithmetic and logical operations, encompassing addition, subtraction, AND, OR, and more.
7. **Control Unit (CU):** Orchestrates the operation of various CPU components based on the fetched instruction, ensuring seamless functionality.
8. **Main Memory:** Functions as the conduit for continuous data exchange between the CPU and memory repository, enabling efficient processing operations.

Design Overview

In this 22-bit Computer, there are 10 instructions that this computer can perform. I used 4 bits for opcode and 5 bits for address. The instructions with their opcode:

Opcode	Instructions	Details
0	AND	Ac = Ac & [address]
1	ADD	Ac = Ac + [address]
2	STO	Stores value of Ac
3	ISZ	Increment and skip if zero
4	BSB	Branch to subroutine
5	BUN	Unconditional Jump
6	LOAD	Loads value at Ac
7	HLT	Stops the program
8	XOR	Ac=Ac xor [address]
9	DEC	Decrement the value

Sample

Test Program: ((((5+3)&7) xor 3)--) where a = 7, b = 5 and c = 3. Store the answer in d.

If we write this code in assembly language for our CPU:

1. LOAD b
2. ADD c
3. AND a
4. STO d
5. XOR c
6. STO d
7. DEC d
8. STO d
9. HLT

The program code for this will be: ***6b 1c 0a 2d 8c 2d 9d 2d 70***

Explanation

We assume there are three data in the a = 7, b = 5, and c = 3. We need to write a code for our CPU. At first, Load the value of b in the accumulator. Then add c with the accumulator. Now, Ac=5+3 = 8. Now AND operation with a. So, 8(1000) & 7 (111) becomes Ac = 0000 (0). Store this in d. Then XOR c from the accumulator. So, 0(0000) xor 3(011) becomes 3(0011). Now, store it in d. Now, decrement the value in d. Store this in d. So, d = 2(0010). Then stop the program.

Discussion

Designing the 22-bit minimal CPU was challenging. I had to ensure each component was simple yet capable. The basic Instruction Set Architecture (ISA) allows the CPU to perform arithmetic and logic operations. The Control Unit (CU) was crucial for coordinating all parts. However, enhancing its capabilities for more

tasks or speed requires further development. I tested everything to ensure functionality. However, advanced CPUs need more extensive testing methods. Despite its simplicity, this project provided valuable insights into CPU workings.

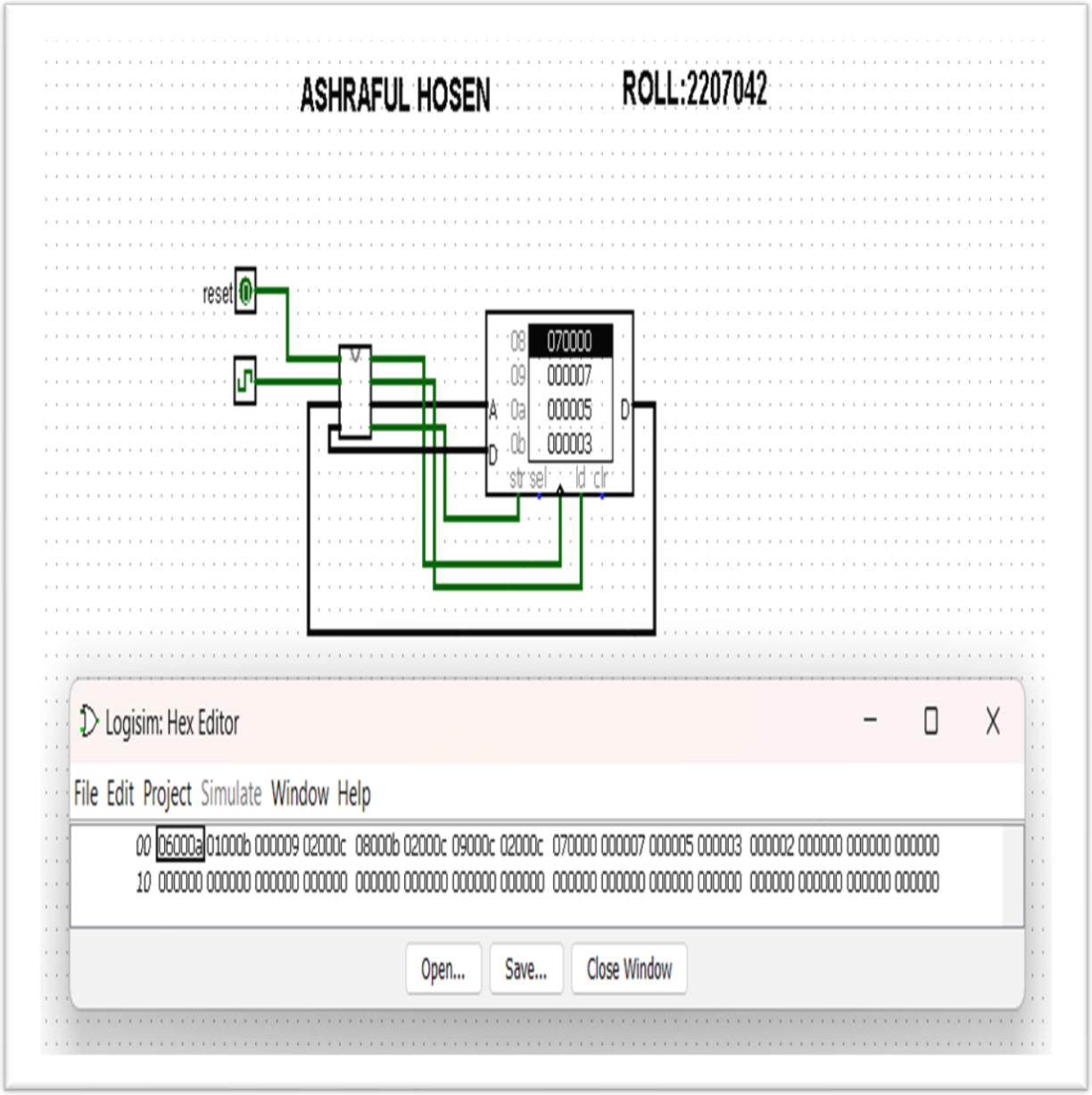

Figure : Final output of sample program