Sri Lanka Institute of Information Technology

Digital electronics- IE1044

Assignment

Implementing a mini processor for BCD arithmetic operations

IT24101643 – J.A.D.S.L. Ranasingha

BSc (Hons) in Computer Science

# Table of contents

# Introduction

Binary–coded Decimal (BCD) is a type of represents a decimal value in four binary digits. BCD values are mostly used in Digital systems and electronics. Because BCD has some common bridges with computer language and human understanding. BCD uses a four–bit binary representation for each decimal digit. That means that instead of converting the whole number into binary, we convert each decimal digit separately to a binary number. Binary numbers cannot be converted normally to BCD, we use a method called the "double dabble algorithm to convert binary numbers to BCD.

BCD is more memory and processing efficient than pure binary, because it uses more bits to represent the same value. However, BCD is useful for applications where the critical consideration is an exact representation of data and is easier to represent and human-readable.

As an example : decimal 47 converted to BCD

4 = 0100

7 = 0111

BCD = 0100 0111

## Problem Statement

This assignment aims to design and simulate a Mini-Processor with arithmetic operations (Adder, subtractor, multiplier) on binary coded decimal (BCD) inputs. According to this digital system that accepts two 1-digit BCD numbers and gives a two-digit BCD output according to a 2-bit control input, it determines how to perform:

00 – Addiction

01- Multiplication

10- Subtraction

Additionally, there must be an overflow indicator if the result exceeds 2 digits.

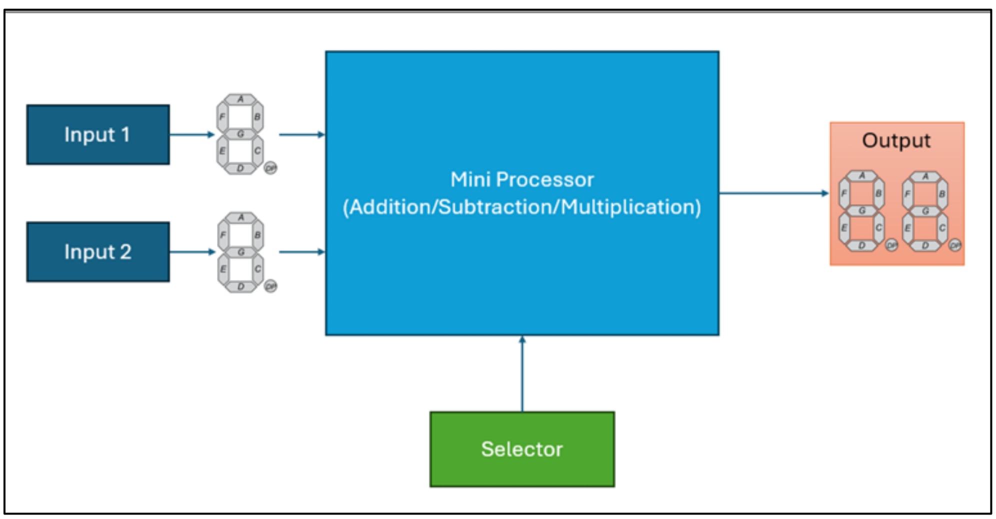



*Figure a Block diagram of the mini-processor*

# Main Components and Their Functions

The main components of the circuit are a BCD Adder, a BCD Subtractor using two's complement, a BCD Multiplier, a Multiplexer to control operation selection, and Registers.

- **BCD Adder**

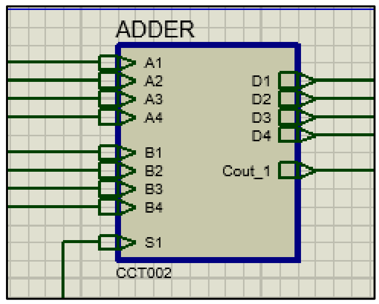



*Figure b : adder*

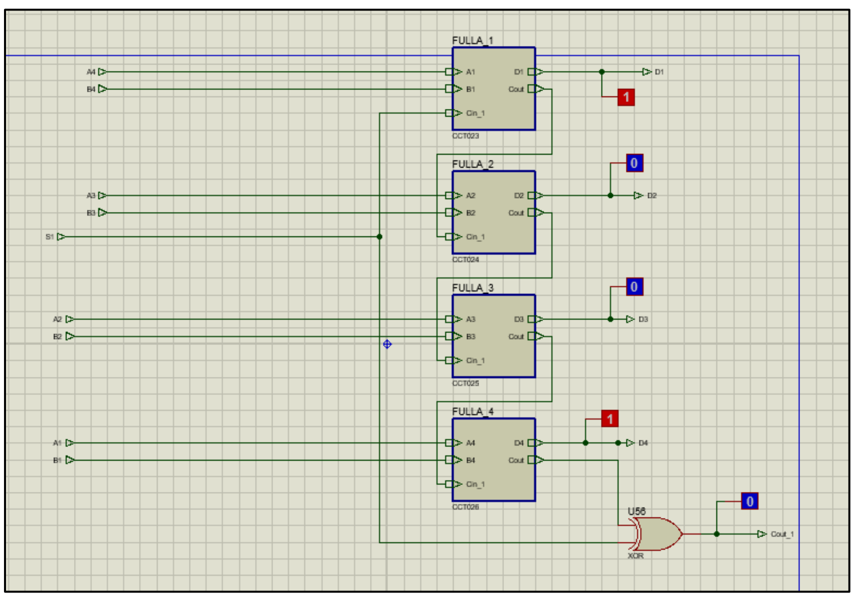



*Figure c : adder sub-circuit*

The BCD adder was implemented by using four full adders. It is used to get a addition of two 4-bit inputs and full adders used to get addiction of this 4-bit binary numbers. In the BCD, if that addition >9 we add a 6 (0110) to it. because BCD value range is 0 to 9. We used XOR for get carry out parity check and overflow detection in signed numbers.

- **BCD Subtractor**

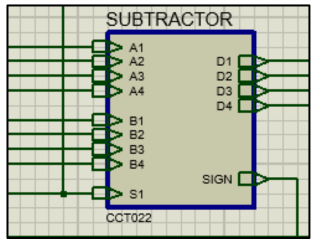



*Figure d : subtractor*

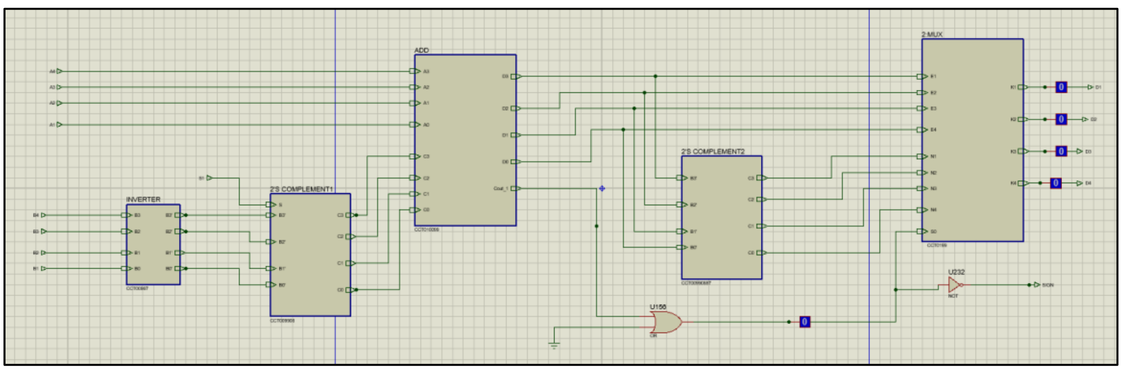


*Figure e : subtractor sub-circuit*

The BCD subtractor was implemented using the two’s complement method. The two’s complement method is the commonly used method for binary subtraction.

**Two’s complement method**

Two's complement is a method used by computers to represent negative binary numbers. The most significant bit (MSB) in this system is utilized to denote the sign, 0 for positive and 1 for negative. To obtain the two's complement of a binary number, invert all the bits and add 1 to the result; for example, the two's complement of 00000101 (which equals +5) is 11111011 (which equals 5). This coding allows for easier arithmetic since positive as well as negative numbers can now be added using the same binary addition procedure

## ➢ BCD Multiplier

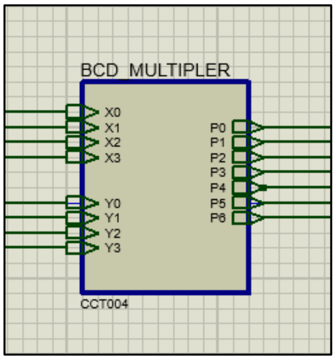



*Figure f : multiplier*

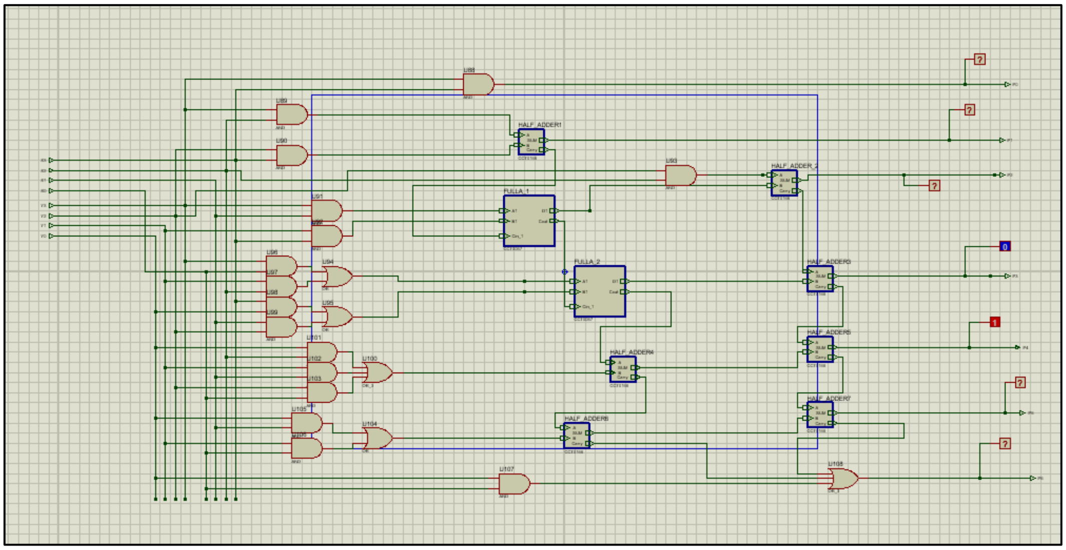


*Figure g : multiplier sub-circuit*

This BCD multiplier is implementation of a 4-bit binary multiplier using basic digital components such as AND gates, half adders and full adders.

- **Multiplexer**

  The multiplexer is a digital circuit that selects one input out of many inputs and forwards that to a single output line. A multiplexer can act like a digital switch. In this circuit, the 4:7 MUX takes inputs from the adder, subtractor, and the multiplier.

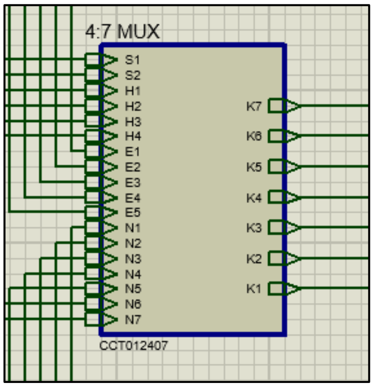



*Figure h : multiplexer*

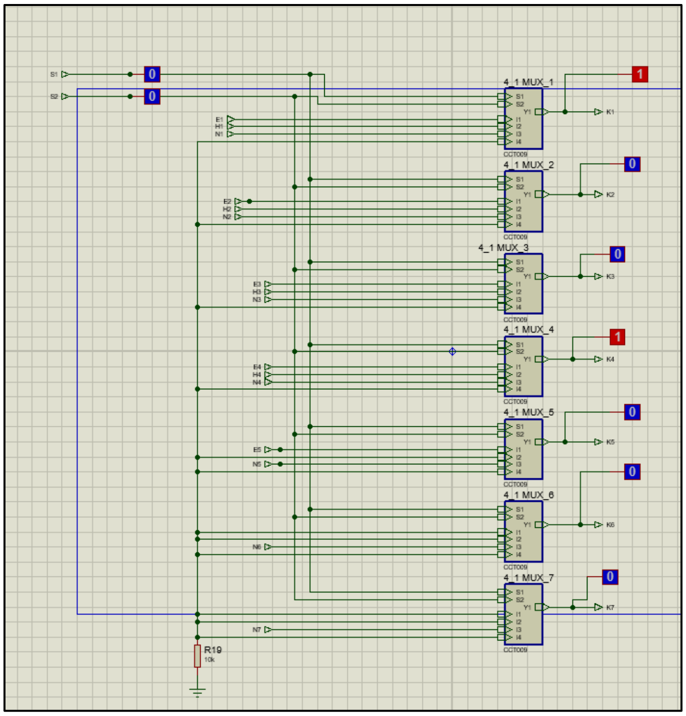


*Figure i : multiplexer sub-circui*

## ➢ Binary to BCD converter

In this circuit, we use binary numbers to BCD converter to convert the given binary number by multiplying, we use the “Double Dable Algorithm”, which double the numbers one by one, and addition in this way.

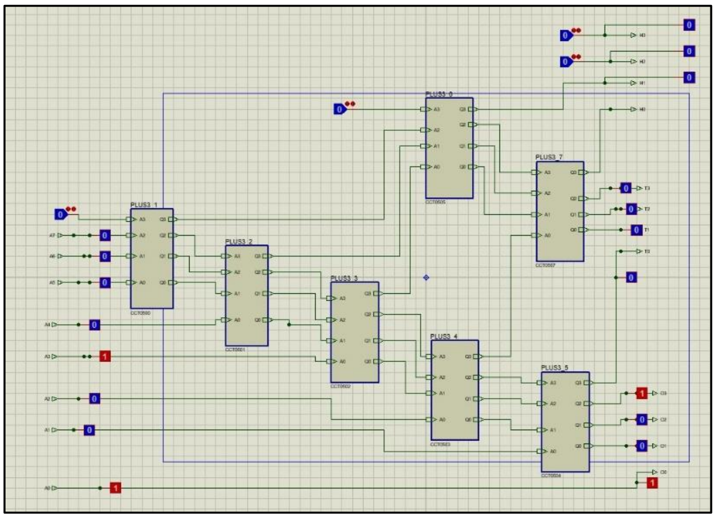
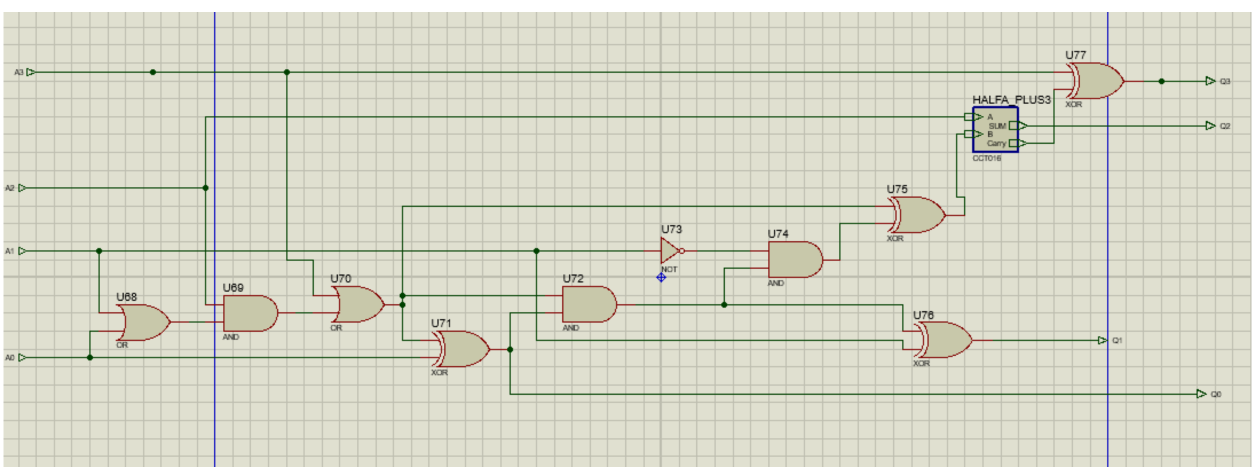


*Figure j : BCD converter*




This is one of the PlUS3 inner circuit.

## ➢ Registers

Registers are fast storage locations within the CPU, and they hold values for arithmetic or logical operations. In this circuit, registers were used to store the input values and output values temporarily.

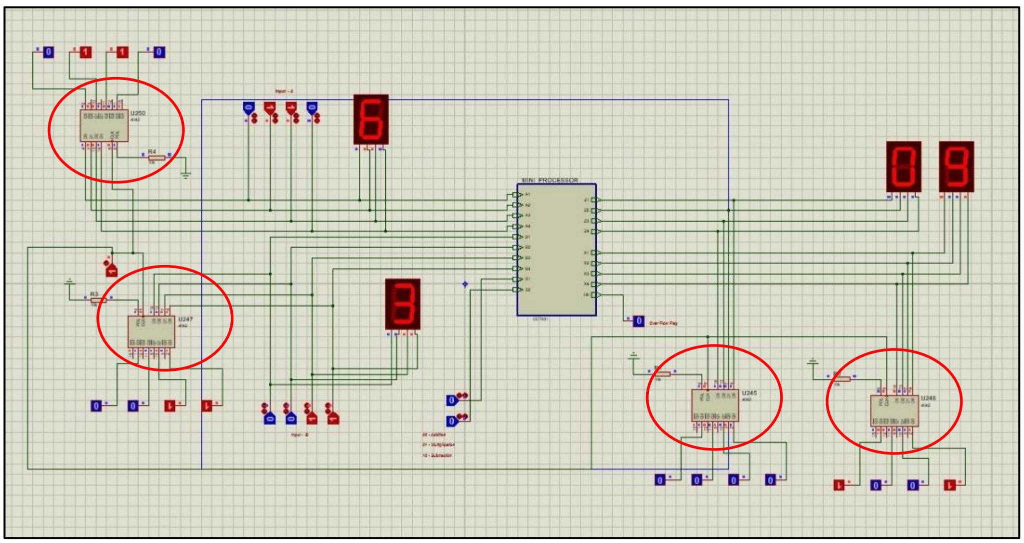
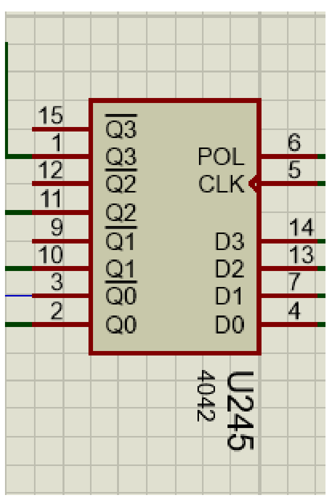


*Figure k : registers*



## Mini processor

The mini processor is a combination of the adder, subtractor, multiplier, and multiplexer.

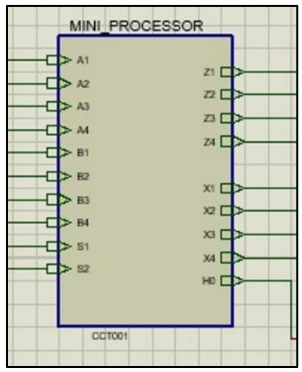



*Figure l : Mini processor*

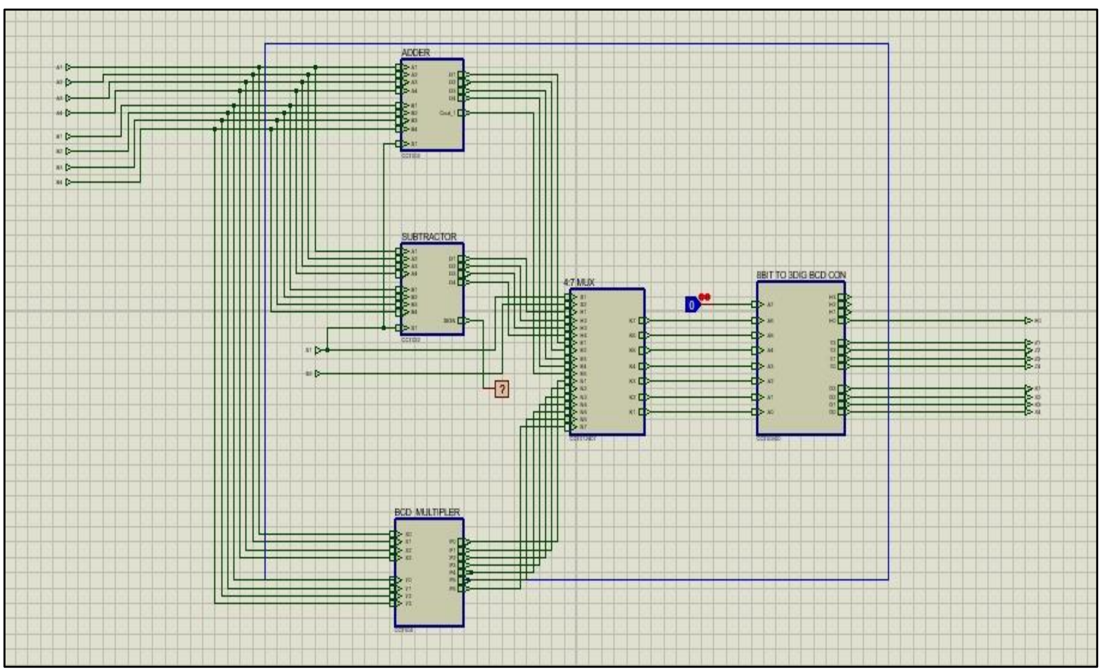
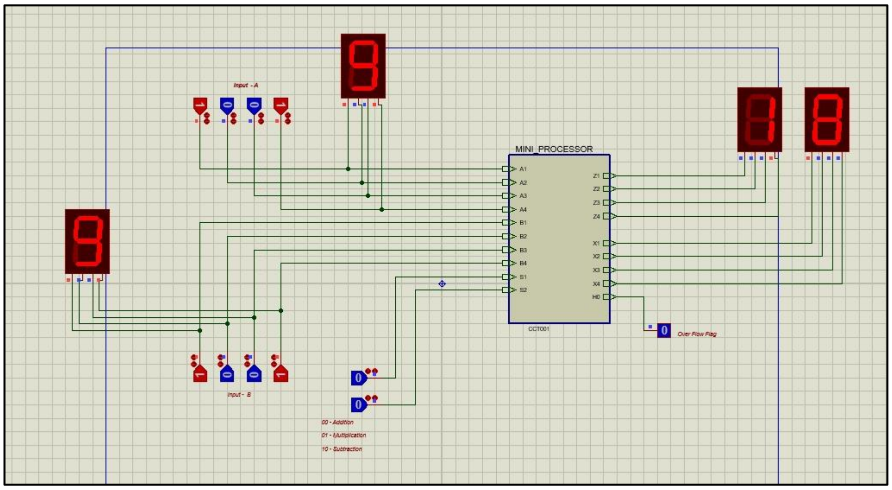
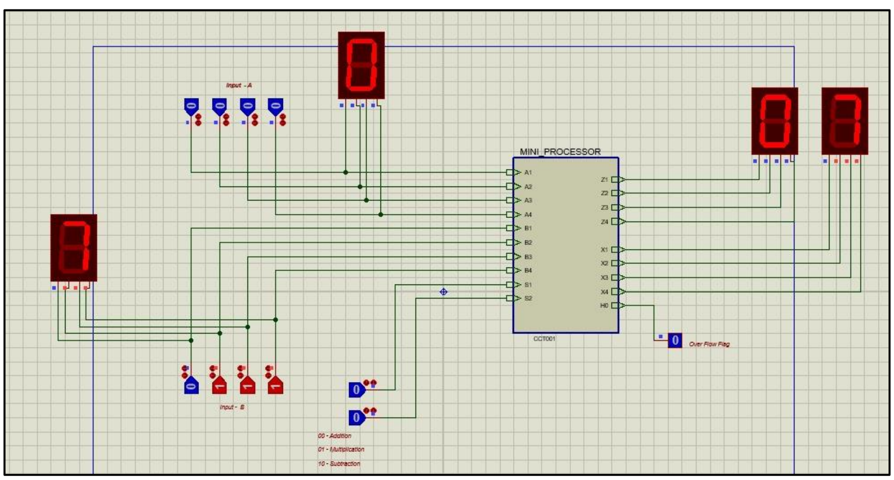


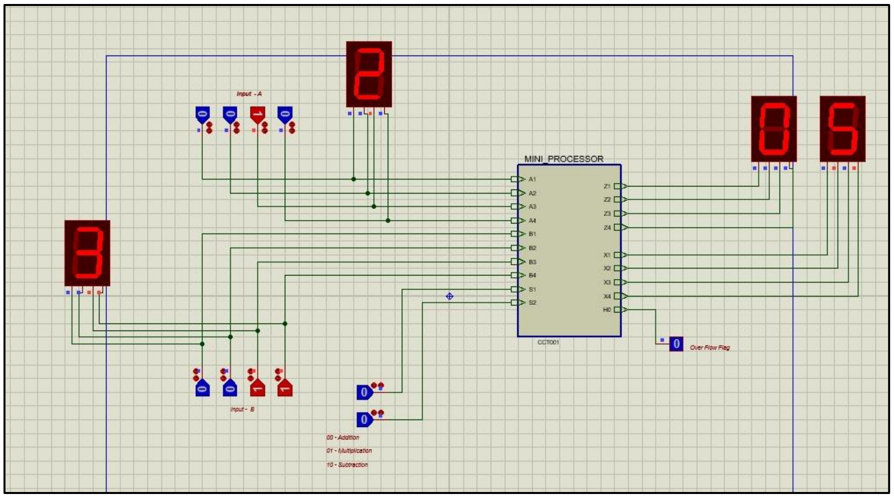

*Figure m : Sub-circuit of the processor*

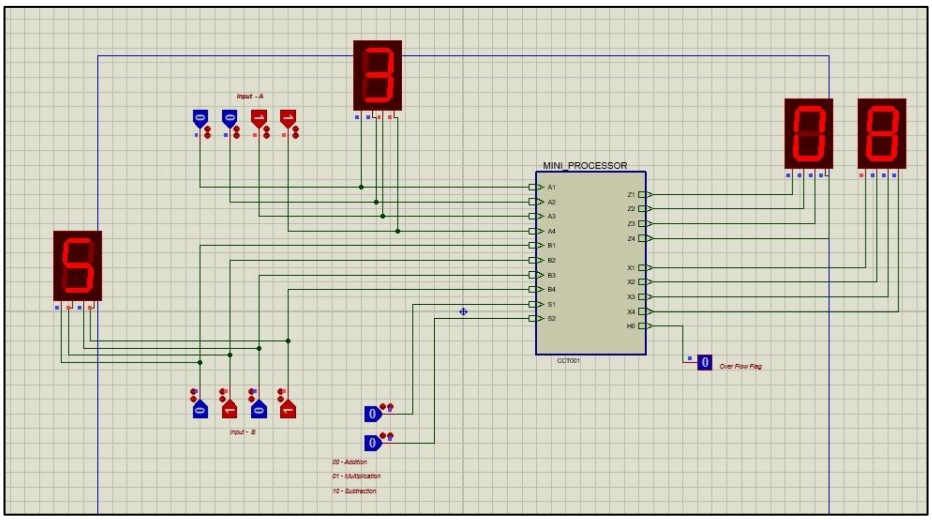
# Simulation results

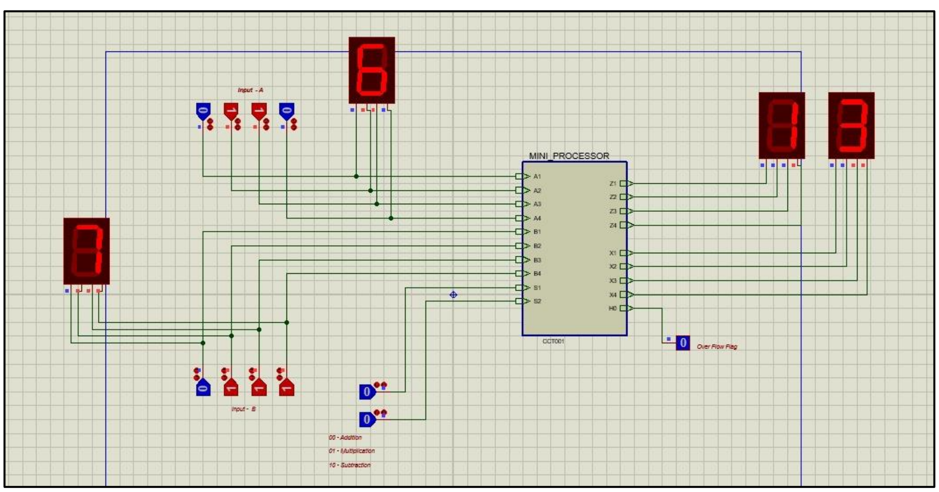
BCD Adder



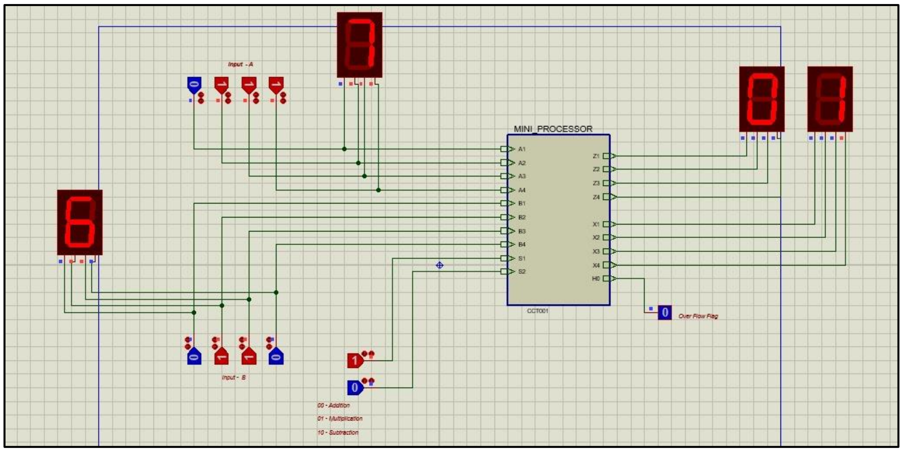



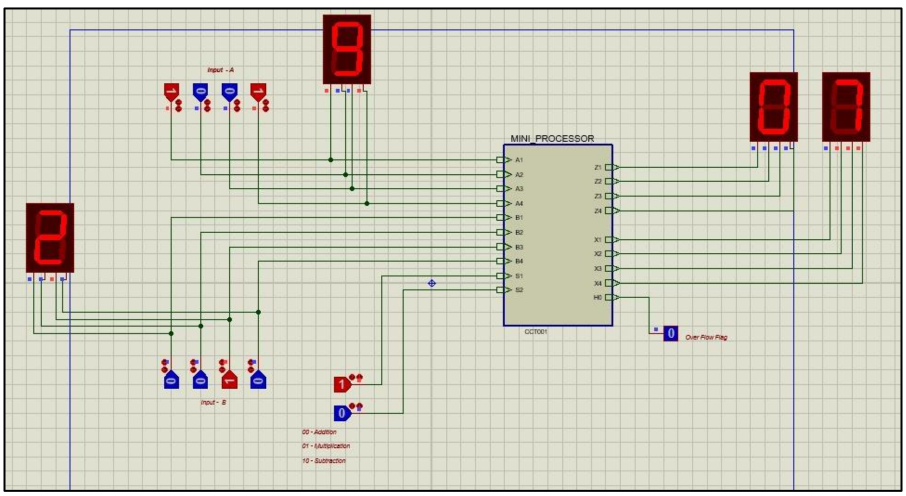

Input - A
MINI_PROCESSOR
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

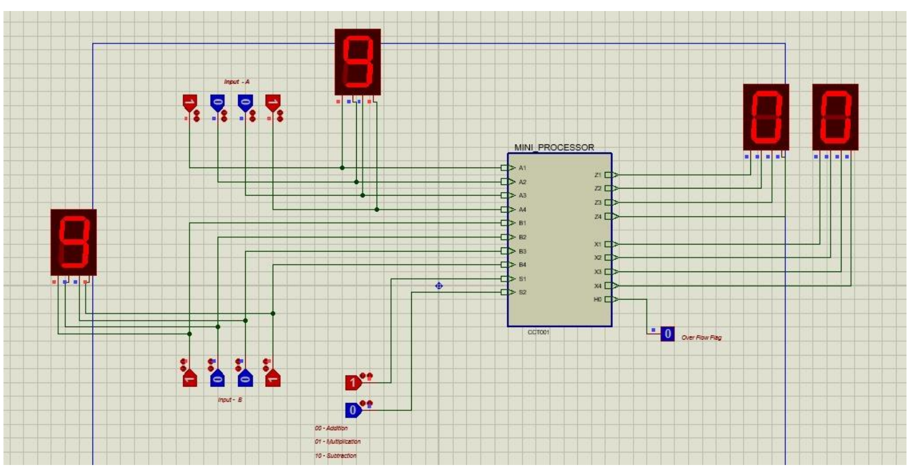

Input - A
MINI_PROCESSOR
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

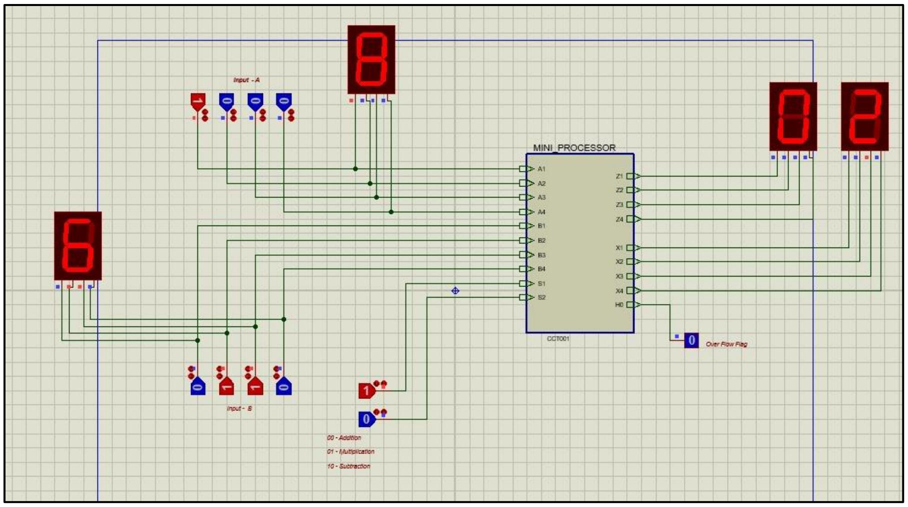

Input - A
MINI_PROCESSOR
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

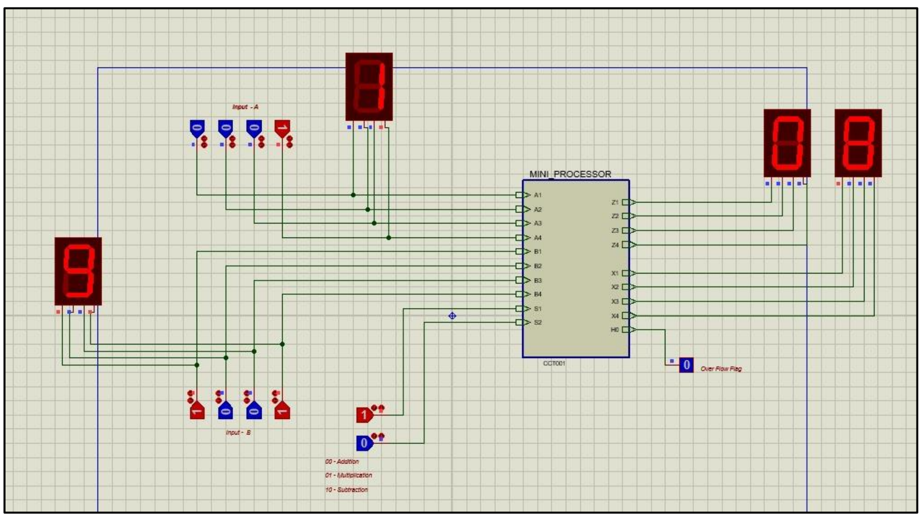
BCD Subtractor



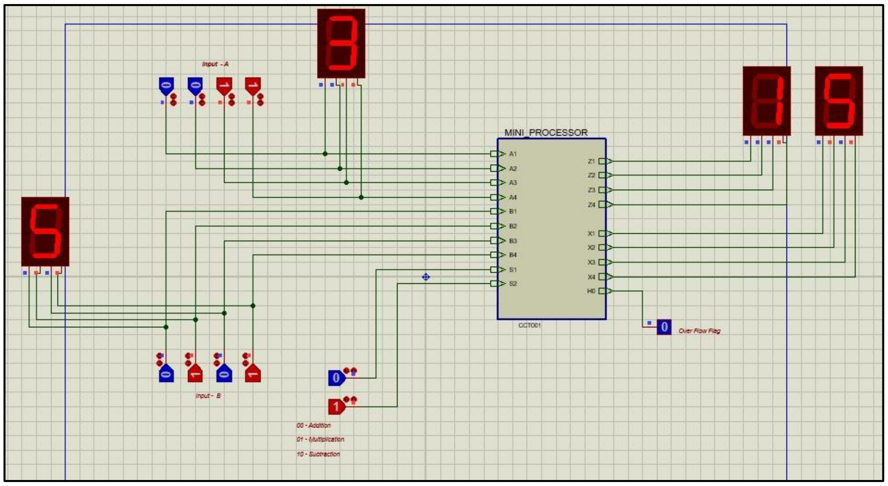



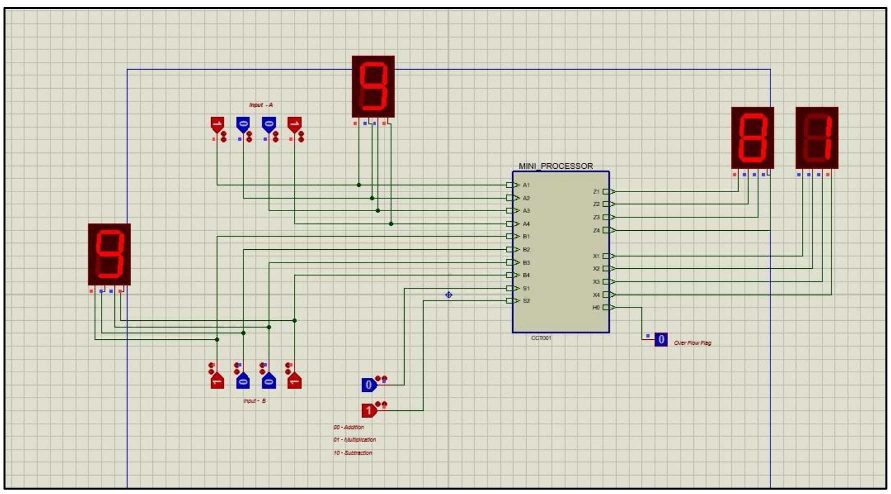



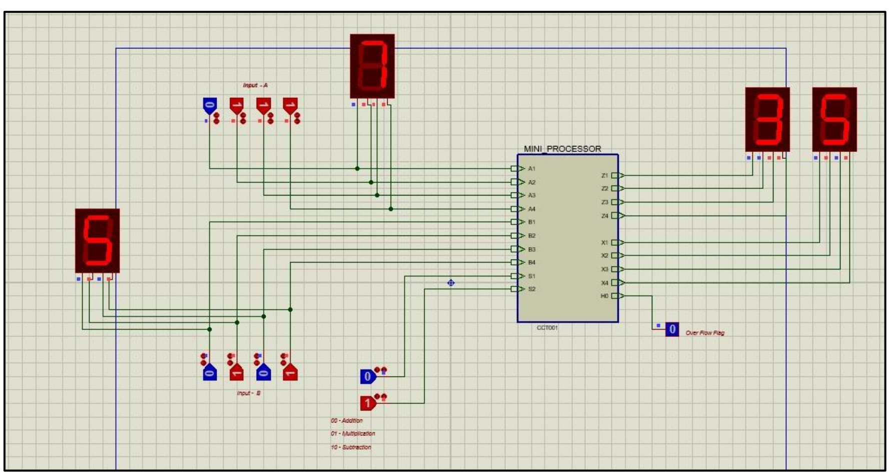

Input - A
MINI_PROCESSOR
A1
A2
A3
A4
B1
B2
B3
B4
S1
S2
Z1
Z2
Z3
Z4
X1
X2
X3
X4
H0
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

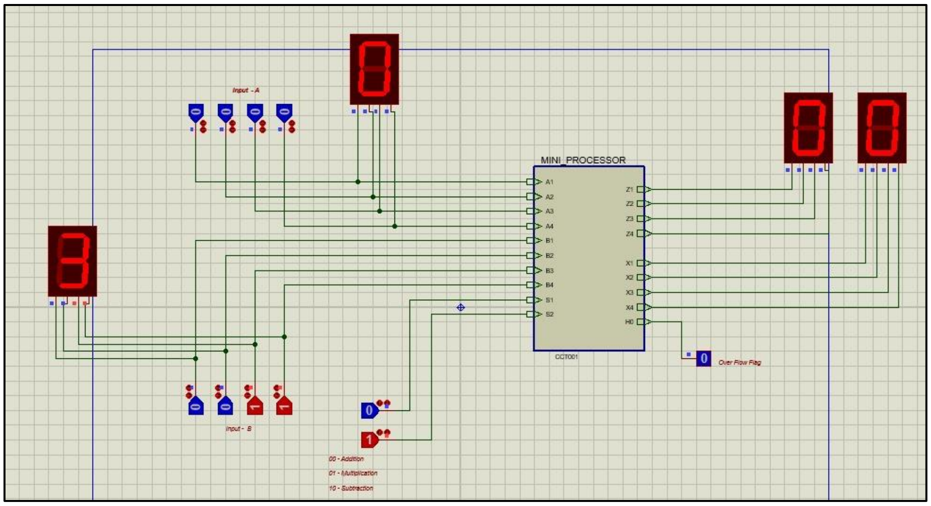

Input - A
MINI_PROCESSOR
A1
A2
A3
A4
B1
B2
B3
B4
S1
S2
Z1
Z2
Z3
Z4
X1
X2
X3
X4
H0
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

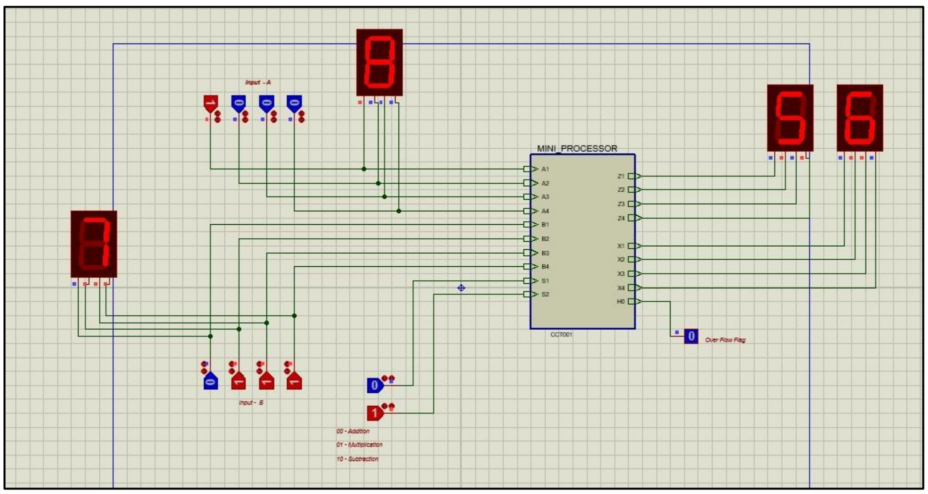
BCD Multiplier




Input - A
MINI_PROCESSOR
A1
A2
A3
A4
B1
B2
B3
B4
S1
S2
Z1
Z2
Z3
Z4
X1
X2
X3
X4
H0
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction


Input - A
MINI_PROCESSOR
A1
A2
A3
A4
B1
B2
B3
B4
S1
S2
Z1
Z2
Z3
Z4
X1
X2
X3
X4
H0
CCT001
Over Flow Flag
Input - B
00 - Addition
01 - Multiplication
10 - Subtraction

# Challenges faced and how they were resolved

1. Implement the component according to the requirements and correct logic. - Implementing the BCD Adder with correction logic, the BCD Subtractor using two's complement and correction, and the BCD Multiplier was a challenge. To solve this challenge, I searched for the components and logic separately and made components accordingly.

2. Converting binary values to BCD values.

3. Getting the overflow flag- I used the double dabble algorithm to indicate the overflow flag.

4. Handling each component separately –a multiplexer was used to get separate results from each component.

# Conclusion

In this project, I have successfully designed a Mini-Processor that accepts two 1-digit BCD numbers as input and, depending upon a control input, executes operations. Addition (00), subtraction (01), and multiplication (10) are the operations accepted by the system with appropriate results for all operations, and it performs through adder, subtractor, and multiplier components. The result is always provided as a 2-digit BCD value, as required by the project. Through careful design and construction, the Mini-Processor is a solid performer for basic arithmetic operations in BCD mode. The project demonstrates the practical application of digital systems in processing BCD numbers with success.

# References

[1] [Online]. Available: https://www.geeksforgeeks.org/twos-complement/.

[2] [Online]. Available: https://youtu.be/2QExUjO4GdQ?si=xVpjZ9qjSB0TtaE2.

[3] [Online]. Available: https://youtu.be/bqxe9Y1LA8c?si=hD3t9Gipo_fcyNPs.

[4] [Online]. Available: https://youtu.be/9O7OJi7RCLQ?si=qNvsJTdEierUUMM3.

[5] [Online]. Available: https://youtu.be/ss9FpyRgIjw?si=4XUs_aSJfwxz2x7p.