ISTANBUL TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

EXPERIMENT NO	:	3
EXPERIMENT DATE	:	07.04.2023
LAB SESSION	:	FRIDAY - 10.30
GROUP NO	:	G3

GROUP MEMBERS:

150200010	:	Ahmet Emre Buz
150200097	:	Mustafa Can Çalışkan

SPRING 2023

Contents

1 INTRODUCTION

In this experiment, we implemented desired designs, and constructed expected tables.

2 PRELIMINARY

2.1

When we want to add two signed binary numbers in 2s complement notation, we follow the same procedure as adding unsigned numbers, but with an additional step to account for the sign bit. We add the two numbers as if they were unsigned binary numbers, and then interpret the result as a signed number by checking the sign bit. If the sign bit is 1, we take the 2s complement of the result to get the correct value. This involves inverting all the bits and adding 1 to the result. In unsigned binary addition, we add the two numbers as if they were regular binary numbers, ignoring any sign bit. If the result overflows the available number of bits, we discard the overflow and wrap around to the lowest bit.

2.2

In 2s complement notation, binary subtraction follows similar rules to binary addition. To subtract two signed binary numbers, we can first add the 2s complement of the second number to the first number. Then, we interpret the result as a signed number. If the result is negative, we need to take the 2s complement of the result to get the correct value. For unsigned binary subtraction, we simply subtract the second number from the first number, ignoring any sign bit. If the result is negative, it cannot be represented in unsigned notation, and we say that there is an underflow.

2.3

Carry occurs when the result of an addition operation exceeds the maximum value that can be represented by the number of bits used to represent the operands. In this case, the most significant bit of the result is set to 1, and the excess bit, or the carry bit, is added to the result of the next addition. Borrow occurs when the result of a subtraction operation requires the next higher bit to be borrowed. For example, when subtracting a larger number from a smaller one, we need to borrow from the next higher bit to make the subtraction possible. The borrow bit is added to the result of the next subtraction. Overflow occurs when the result of an arithmetic operation exceeds the maximum or minimum value that can be represented by the number of bits used to represent the operands. For example, when adding two positive numbers results in a negative number, or when subtracting a negative number from a positive number results in a positive number. Overflow can occur in both signed and unsigned arithmetic operations, but its interpretation depends on the sign of the operands.

3 EXPERIMENT

3.1 PART 1

The given circuit is implemented and tested. Truth table is constructed.

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Table 1: Truth table of a half adder

3.2 PART 2

The carry in input is added. Truth table is constructed.

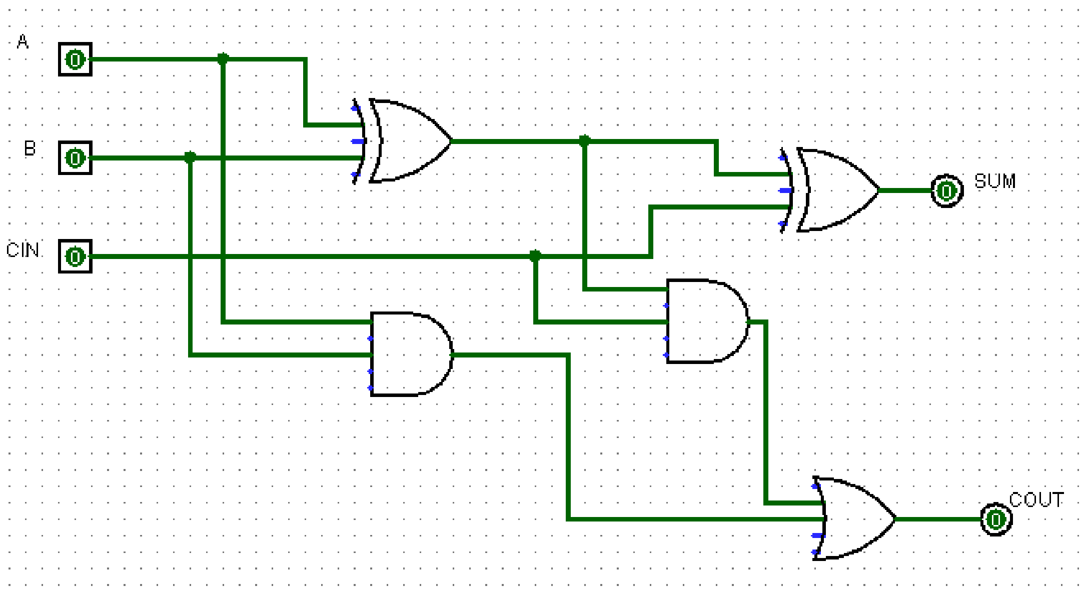

Figure 1: Full Adder

A	B	Cin	Sum	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table 2: Truth table of a full adder

3.3 PART 3

Expected calculations are performed. Expected tables are constructed.

A	B	Carry	Result in Binary	Result in Decimal
0101	0111	0	1100	12
1101	1001	1	0110	22
1111	1111	1	1110	30
0110	1101	1	0011	19

Table 3: Unsigned A + B

A	B	Borrow	Result in binary	Result in decimal
0101	0111	1	1100	12
1101	1001	0	0100	4
1111	1111	0	0000	0
0110	1101	1	1001	9

Table 4: Unsigned A - B

A	B	Overflow	Result Sign	Result in Binary	Result in Decimal
0101	0111	1	1	1100	-4
1101	1001	1	0	0110	6
1111	1111	0	1	1110	-2
0110	1101	0	0	0011	3

Table 5: Signed A + B

A	B	Overflow	Result Sign	Result in Binary	Result in Decimal
0101	0111	0	1	1100	-4
1101	1001	0	0	0100	4
1111	1111	0	0	0000	0
0110	1101	1	1	1001	-7

Table 6: Signed A - B

4 CONCLUSION

As a result, we designed and implemented expected circuits, and constructed expected tables.