# Your Title

Your Name

September 26, 2024

# 1 Introduction

An arithmetic logic unit (ALU) is a multi-operation, combinational-logic digital function. It can perform a set of basic arithmetic operations and a set of logic operations. The ALU has a number of selection lines to select a particular operation in the unit. The selection lines are decoded within the ALU so that k selection variables can specify up to $2^k$ distinct operations.

Figure 1 shows the block diagram of a 4-bit ALU. The four data inputs from A are combined with the four inputs from B to generate an operation at the F outputs. The mode-select input $s_2$ distinguishes between arithmetic and logic operations. The two function-select inputs $s_1$ and $s_0$ specify the particular arithmetic or logic operation to be generated. With three selection variables, it is possible to specify four arithmetic operations (with $s_2$ in one state) and four logic operations (with $s_2$ in the other state). The input and output carries have meaning only during an arithmetic operation. The input carry in the least significant position of an ALU is quite often used as a fourth selection variable that can double the number of arithmetic operations. In this way, it is possible to generate four more operations, for a total of eight arithmetic operations.

The input carry in the least significant position of an ALU is quite often used as a fourth selection variable that can double the number of arithmetic operations. In this way, it is possible to generate four more operations, for a total of eight arithmetic operations.

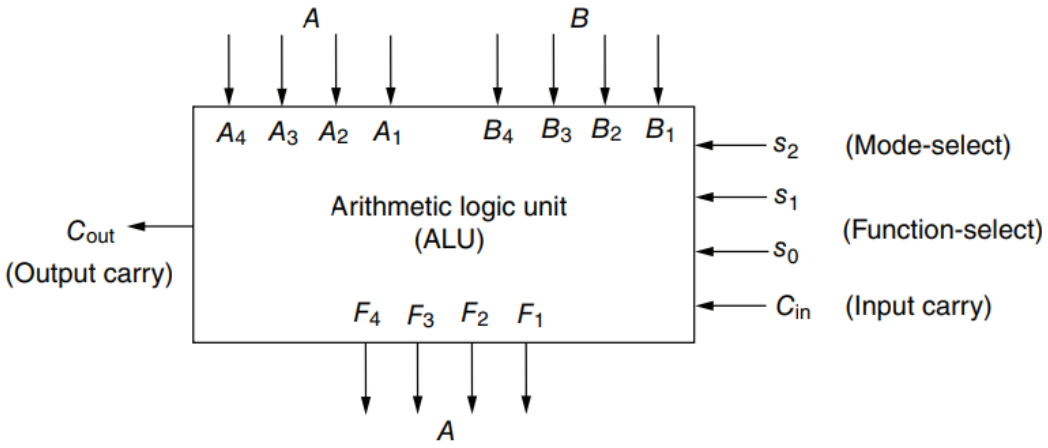


Figure 1: Block Diagram of a 4-bit ALU

The design of a typical ALU will be carried out in three stages. First, the design of the arithmetic section will be undertaken. Second, the design of the logic section will be considered. Finally, the arithmetic section will be modified so that it can perform both arithmetic and logic operations.

# 2 Problem Specification with assigned instructions

**Problem Specification:**

1. Efficiently design (with minimum possible ICs) the ALU according to the specification.
2. Implement the following flags:
   - Carry (C)
   - Sign (S)

- Overflow (V)
- Zero (Z)

3. Flags should be affected as per the rules of Assembly Language (with some exceptions).

4. However, some exceptions (only for this assignment) has been added to incorporate flexibility to flag status bits after logical operations. This flexibility is to make the assignment easier, although it breaks some of the rules of the assembly programming language.

   - For NOT Operation:
     (a) After the NOT operation, Z flag is 1 if the answer is 0000 and the Z flag is 0 otherwise; ie. The Z flag functions as it normally would.
     (b) After the NOT operation, if S remains unchanged or it reflects the highest order bit of the result, both will be accepted. But if the S flag is changed and it is changed to a wrong value, it will not be accepted.
     (c) To make your life easier, we shall not check the C and V flags after NOT operation, i.e., you can consider these as Don't care.
   - For AND/OR/XOR Operation:
     (a) C and V should be cleared (0) after the operation.
     (b) S and Z should be changed according to the output.

5. Any 2-input SSI (AND, OR, NOT, XOR, etc.) and MSI (MUX, Decoder, Adder, etc.) chip can be used.

6. Emphasis should be given to the efficiency of design and minimization of ICs used.

7. For simulation, one can use any simulation software.

8. Software design must be at IC level.

9. Everyone have to write a report.

**Functional Design Specifications:**

| **CS2** | **CS1** | **CS0** | **Functions** |
|---|---|---|---|
| 0 | 0 | 0 | Add |
| 0 | 0 | 1 | AND |
| 0 | 1 | X | Sub with borrow |
| 1 | 0 | 0 | Complement A |
| 1 | 0 | 1 | OR |
| 1 | 1 | X | NEG A |

Table 1: Control Signals and Functions of the 4-bit ALU

Here, CS2, CS1, and CS0 are the control signals. X means that the value of the signal is not important.

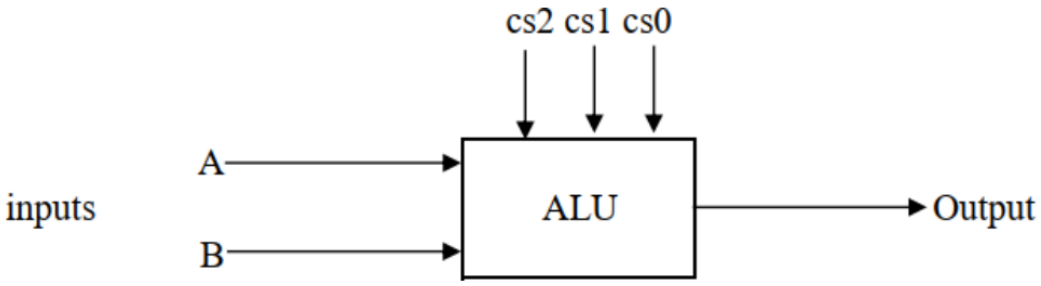


Figure 2: Block Diagram of a 4-bit ALU

| CS2 | CS1 | CS0 | Functions | X | S11 | S10 | Y | $\bar{E}2$ | S2 | Cin |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | Add | A | 0 | 0 | B | 0 | 0 | 0 |
| 0 | 0 | 1 | AND | AB | 0 | 1 | 0 | 1 | X | 0 |
| 0 | 1 | X | Sub with borrow | A | 0 | 0 | B' | 0 | 1 | 0 |
| 1 | 0 | 0 | Complement A | A' | 1 | 0 | 0 | 1 | X | 0 |
| 1 | 0 | 1 | OR | A+B | 1 | 1 | 0 | 1 | X | 0 |
| 1 | 1 | X | NEG A | A' | 1 | 0 | 0 | 1 | X | 1 |

Table 2: Truth Table of the 4-bit ALU

## 3 Detailed design steps with k-maps (if applicable)

## 4 Truth Table

## 5 Block Diagram

## 6 Complete Circuit Diagram

## 7 ICs used with count as a chart

| IC | Count |
|---|---|
| 74153 | 2 |
| 74157 | 1 |
| 7408 | 2 |
| 7432 | 2 |
| 7404 | 2 |
| 7486 | 1 |
| 7483 | 2 |