FAST National University

Digital Logic Design

Project Report

Submitted By

Ali Kamal
19I-1865

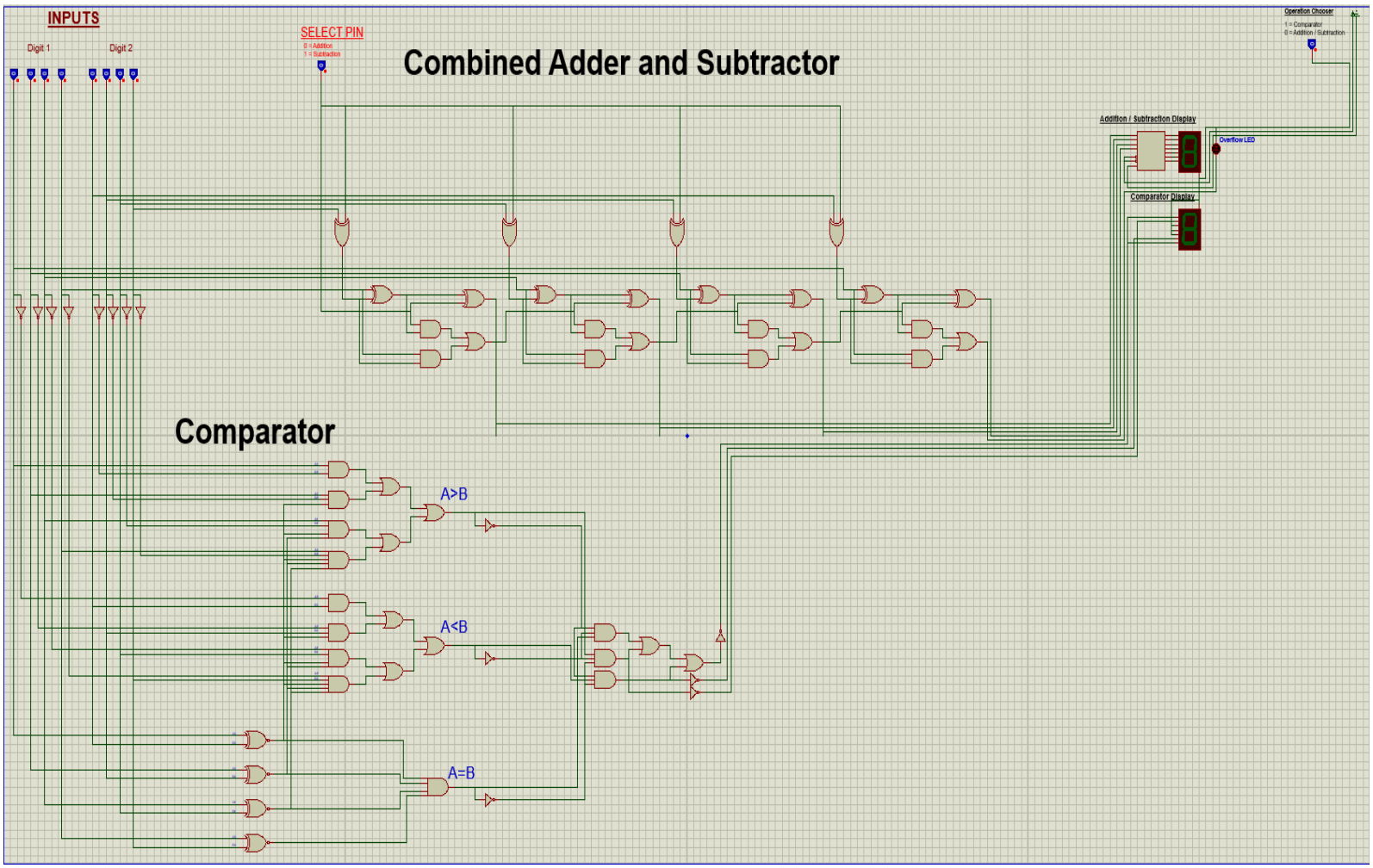
Ali Kamal
19I-1865

Table of Contents

Ali Kamal
19I-1865

4-Bit Arithmetic-Logic Unity (ALU)

Project Diagram



Ali Kamal
19I-1865

Components Used

- 14x – NOT Gates
- 4x – 2 Input XNOR Gates
- 12x – 2 Input XOR Gates
- 10x – 2 Input AND Gates
- 5x – 3 Input AND Gates
- 3x – 4 Input AND Gates
- 2x – 5 Input AND Gates
- 12x – 2 Input OR Gates
- 1x – BCD to 7-Segment Decoder
- 1x – 7 Segment Common Cathode Display
- 1x – 7 Segment Common Anode Display
- 1x – LED-Green
- 1x – POWER
- 1x – GROUND
- 10x – LOGIC TOGGLE

Implementation Details

Combined Adder and Subtractor

I first decided that, instead of making separate circuits for addition and subtraction, I would combine both operations into a single circuit. I first made a simple 1-bit full adder, by first making a truth table and K-Map for the SUM, which gave me A⊕B⊕C, with A and B being the inputs, and C being the carry-in. Similarly, for the CARRY output, after simplifying through truth table and K-Map, I got the equation AB+BC+AC. I implemented these two equations by using two XOR gates, two 2-Input AND gates, and one 2-Input OR gate.

Similarly, for a 4-bit full adder, I simply cascaded my 1-bit adder four times, effectively making a 4-bit full adder, with the carry in input of the full adders being the carry output of the previous adders (not counting the first full adder). This gave me four separate SUM bits (S0, S1, S2, S3), and 1 CARRY out bit, which would signify whether an overflow occurred or not. An overflow would mean the answer is we get on the display is not correct. The four separate SUM bits i-e

S0, S1, S2, S3 are then sent to a BCD to 7-Segment Decoder, and then subsequently sent to a 7 Segment Common Cathode Display, which is used for displaying the answer. The CARRY out bit, which is used to signify an overflow, is connected to a Green LED bulb, which would then light up if an overflow occurs.

To add subtraction functionality to the circuit, I simply used 4x XOR gates with each of the 'B'[1] input bits. For the second input of the XOR gates, I made a select pin, using a LOGIC TOGGLE switch, which would either give 0 or 1. A value of 0 would specify that the user is performing addition, while a value of 1 would specify that the user needs to perform subtraction. This value of the select pin goes to each of the inputs of the XOR gate (that is also connected to each of the 'B' input bits), and also to the CARRY input of the first full adder. A value of 1 (subtraction) then inverts every input of the 'B' input bit, as a property of the XOR gate is that on an input of 1, it inverts the value of the other input, while on an input of 0, the other input remains same. Hence, if an input of 0 (addition) is given to the select pin, no change is done to the 'B' input bits, and addition is done as usual. However, if an input of 1 (subtraction) is given to the select pin, every 'B' input bit is inverted, hence converting the input to 1's complement. A value of 1 is then also given to the CARRY input of the first full adder, hence converting the 1's complement to a 2's complement. Then, by addition, subtraction is done.

Comparator

In the comparator, for the EQUAL (A=B) output, I first devised a truth table for the inputs and outputs, and simplified through K-Maps. This gave me the equation of 'A XNOR B', for each of the four bits. I then put all of the four outputs from the XNOR gates into a 4 input AND gate, which only gives an output of 1 if all 4 outputs from the XNOR gate is 1, as all four bits must be equal in order for the number to be considered as equal.

Similarly, for the GREATER (A>B) output, the equation " A3B3' + x3A2B2' + x3x2A1B1' + x3x2x1A0B0' " was made, where x3 is the output of 'A3 XNOR B3', x2 is the output of 'A2 XNOR B2', and x1 is the output of 'A1 XNOR B1'. This circuit was then made using: one 2-Input AND gate, one 3-Input AND Gate, one 4-Input AND Gate, one 5-Input AND Gate, and three 2-Input OR Gates.

Similarly, for the LESSER (A<B) output, the equation " A3'B3 + x3A2'B2 + x3x2A1'B1 + x3x2x1A0'B0 " was made, where x3 is the output of 'A3 XNOR B3', x2 is the output of 'A2 XNOR B2', and x1 is the output of 'A1 XNOR B1'. This circuit was then made using: one 2-

[1] B is the second digit in the equation, i-e A-B

Input AND gate, one 3-Input AND Gate, one 4-Input AND Gate, one 5-Input AND Gate, and three 2-Input OR Gates.

I then put all these three outputs to a 7 Segment Common Anode Display. Before I could do that, I wanted to display the ">" sign for the A>B output, on the 7 Segment Display, the ">" sign for the A<B output, on the 7 Segment Display, and the "=" sign for the A+B output on, on the 7 Segment Display. For this, I again made a truth table for the individual LEDs (a, b, c, d, e, f, g) on the 7 Segment Display, and simplified the equations for the individual LEDs using K-Map. For the individual equations of the LEDs, I got the following equations.

- LED 'a' = A>B.(A<B)'.(A=B)' + (A>B)'.(A<B)'.A=B + (A>B)'.A<B.(A=B)'
- LED 'g' = A>B.(A<B)'.(A=B)' + (A>B)'.(A<B)'.A=B + (A>B)'.A<B.(A=B)'
- LED 'b' = A>B.(A<B)'.(A=B)'
- LED 'f' = (A>B)'.A<B.(A=B)'

I then put these individual outputs to their respective LEDs in the 7 Segment Common Anode Display.

Choosing an operation

In order to choose between displaying addition/subtraction results or the comparator results, I used Cathode and Anode 7 Segment Displays. For the addition/subtraction results, I used a Cathode display (which works when connected to GROUND i-e = 0), and for the comparator results, I used an Anode display (which works when connected to POWER i-e = 1). A LOGIC TOGGLE switch chooses between the two. A value of 1 on the switch chooses the comparator display, and the 1 value (POWER) is sent to the Anode display, which turns on and displays the result. A value of 0 on the switch chooses the addition/subtraction result, and the 0 value (GROUND) is sent to the Cathode display, which turns on and displays the result.

To choose between addition and subtraction, a LOGIC TOGGLE Select Pin is used, where a value of 0 is used for addition, and a value of 1 is used for subtraction. The functionality of this Select Pin has already been explained in detail previously.