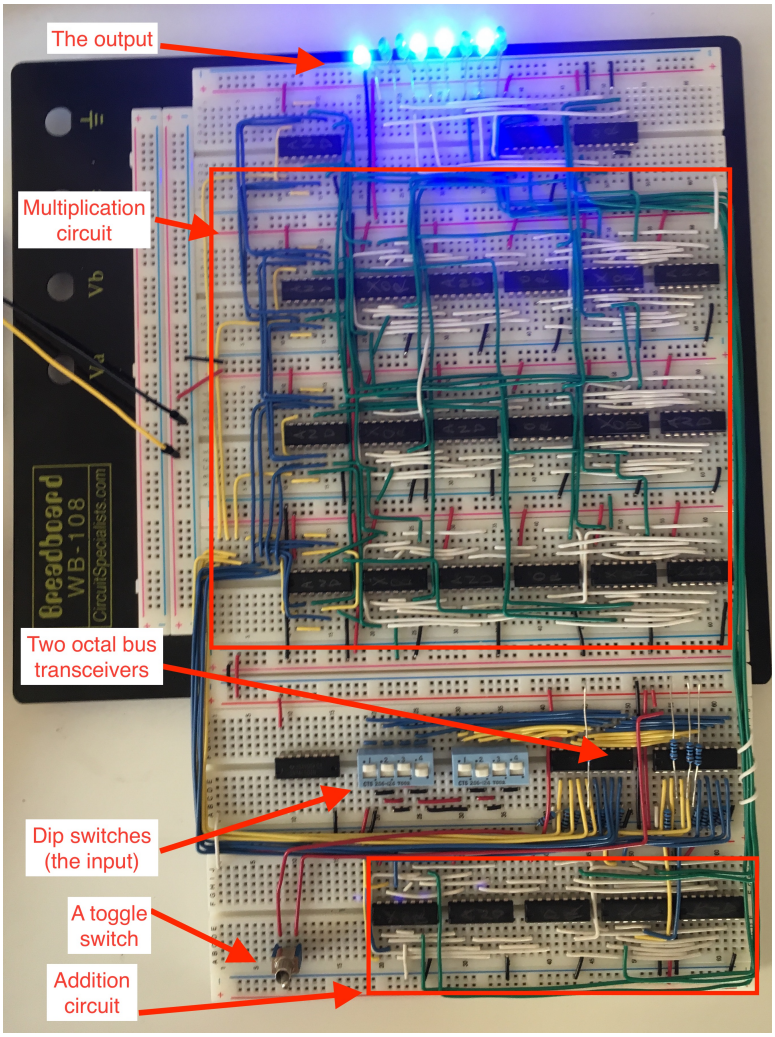
Lab 12 - Final Project Report

Anton Lukanov

05/7/2018

I've built a 4-bit calculator that adds and multiplies two binary numbers. I've constructed an adder and a multiplier sub-circuits using the AND, OR, and XOR logic gates of LS74 series. I've used a toggle switch to control the flow of the data for the circuit to perform either addition or multiplication at a single moment. The 8-bit output number is represented by the LEDs each of which represents a 1 when it glows and a 0 when it doesn't. I then measured the time it takes for the addition to compute the least and the most significant bits of the output, and found the one of the latter to be greater.



Introduction

My initial intention was to make a simple 8-bit central processing unit, but Amin Jazaeri told me that it didn't fit for a two week project. Therefore I decided to make a 4-bit calculator that adds and multiplies two unsigned binary numbers.

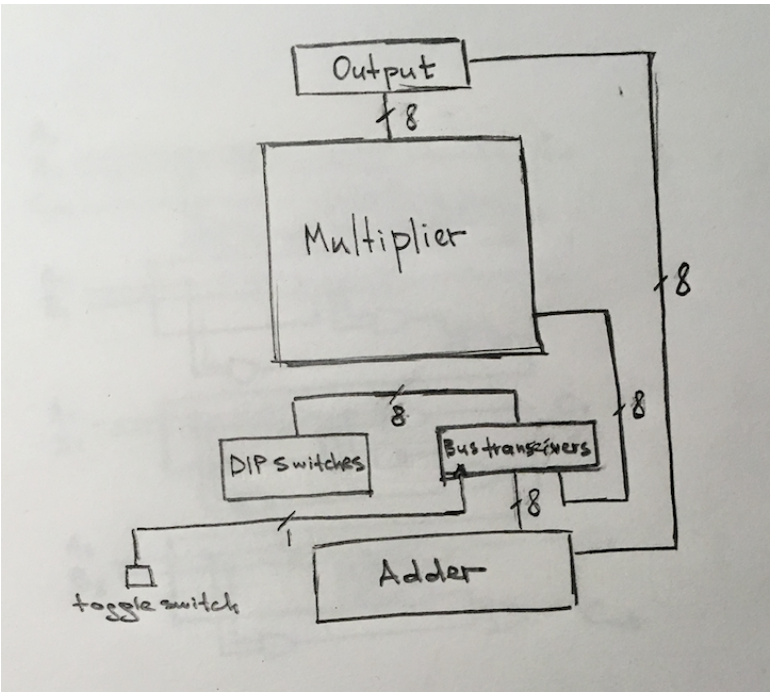
I first built an adder circuit and tested it to make sure it worked properly. I then built a multiplier circuit that is a little more complex in terms of wiring although it is essentially built out of three separate adder circuits. I tested it using just a couple of input numbers since it was cumbersome to test it thoroughly without having the dip switches installed beforehand. The multiplier circuit worked well. Then I started setting the dip switches to use them for the input numbers. I ran into a problem that when I switched a certain bit off on a dip switch, the output signal going from that switch wasn't at 0V but at around 1.4V, which the logic gates took as a high-level input voltage. I solved this problem by realizing that the dip switches that Amin Jazaeri gave me had two different inputs corresponding to two different outputs for each bit. I interconnected those outputs for each bit so that each bit's output is 5V if that bit's switch is turned on and 0V if that is turned off.

I then started to set up two octal bus transceivers, the function of which is to transfer the input data from the dip switches to either an adder or a multiplier based on the setting of the toggle switch. I intended to simultaneously activate a certain bus transceiver and deactivate another one with a toggle switch for the circuit to either perform the addition or multiplication, but not both at the same time. The problem I encountered here was that when a certain bus transceiver is deactivated, the output of it was at about 1.6V for each bit while I expected it to be at 0V. This led the logic gates to take those signals as high-level inputs which consequently disrupted the calculations. I solved this problem by making a voltage divider for each of the bits so that when a bus transceiver is deactivated, it outputs voltage within low-level voltage threshold and when it's activated, high voltage bits are still taken as high-level inputs by the logic gates. The resistor value that I experimentally found to be working well for this purpose was 1KΩ.

I then connected all of the blocks of sub-circuitry together and with little debugging the circuit began to work as expected. Note that the input numbers in the image on the previous page are 1110_2 (14_{10}) and 1011_2 (11_{10}) and the output is 10011010_2 (154_{10}) when the circuit is set to multiplication.

The description of the circuit

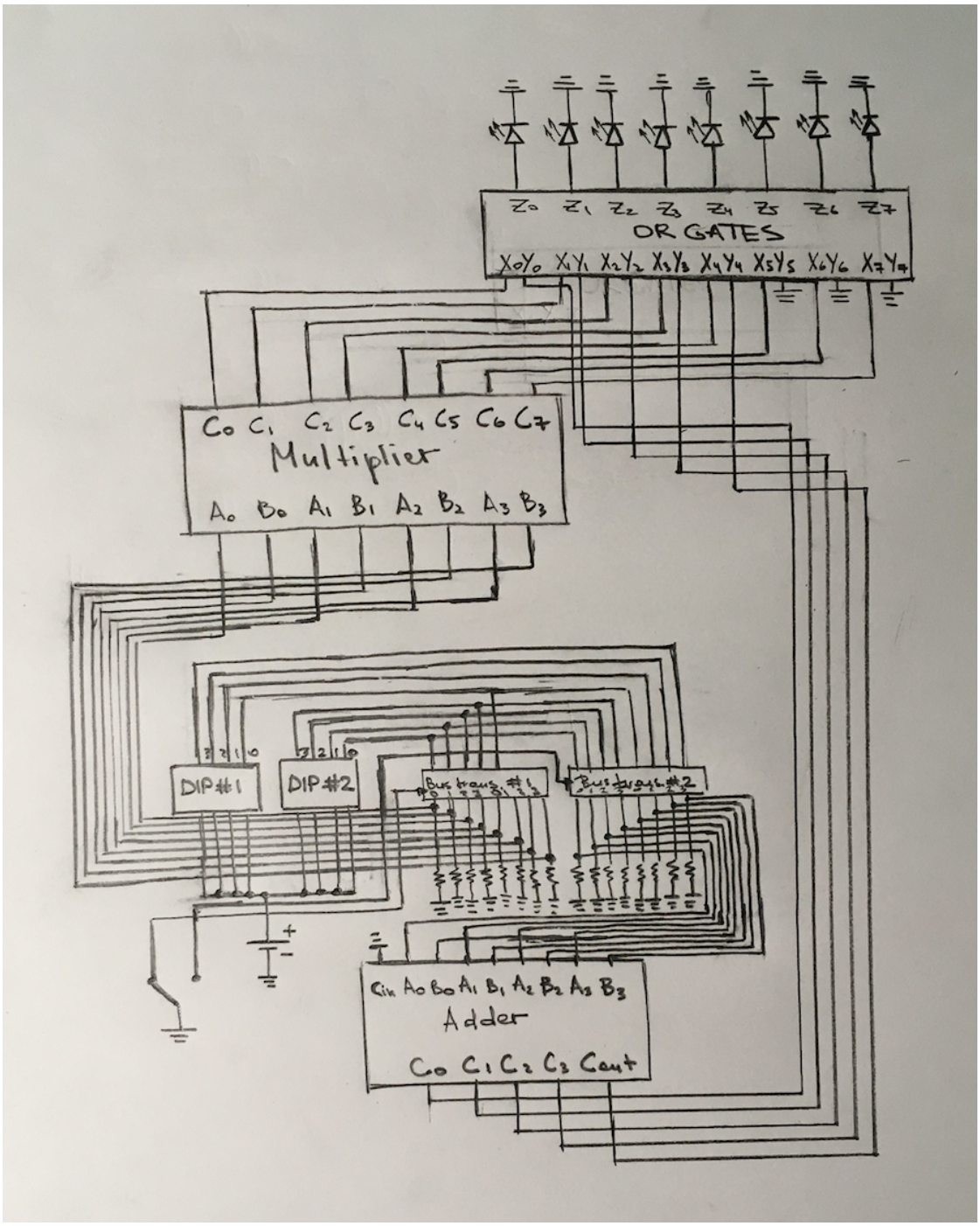
Below is a block diagram containing major blocks and components of my circuit.



The following operations are proceeded in the circuit.

1. Two 4-bit numbers are inputted into the DIP switches.
2. They are transferred to two octal bus transceivers, one of which is for addition and another for multiplication.
3. The toggle switch is set so that it simultaneously activates one of the octal bus transceivers and deactivates the other one.
4. The data of the activated bus transceiver flows into an adder or a multiplier circuit and is used as an input in it.
5. Two numbers are either added or multiplied resulting in an output.

Below is a detailed circuit diagram.



I used SN74LS08N for AND gates, SN74LS32N for OR gates, and SN74LS86AN for XOR gates. I used those and not other series's gates because those are the ones I've encountered most often when I was doing my research on an adder and a multiplier circuits. AND gates are essentially switches (transistors) in series, OR - in parallel, and XOR is some combination of those that results in 1 when one of two inputs is 1 and the other one is 0. I used AND, OR, and XOR gates for constructing an adder and a multiplier.

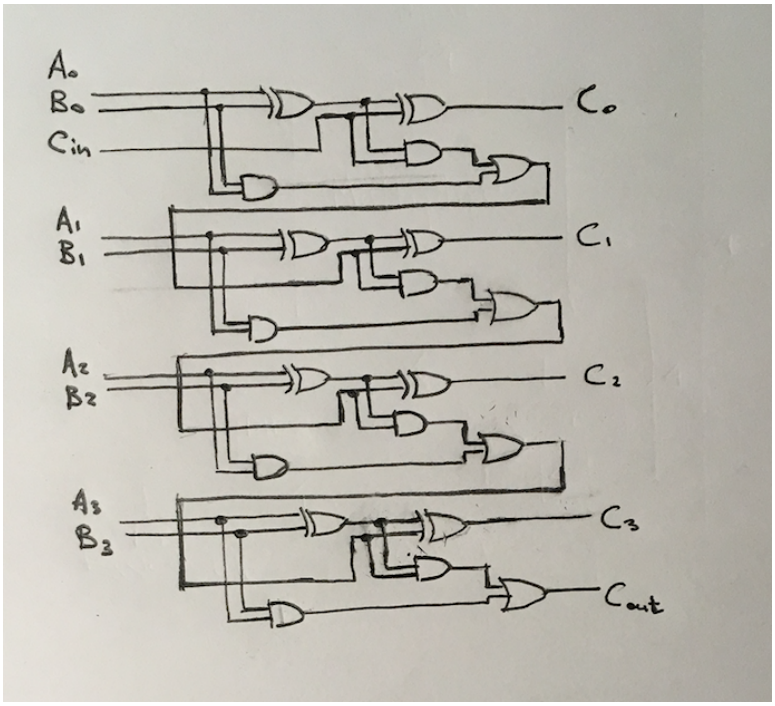
I used octal bus transceivers numbered SN74LS245N. I needed them to control the flow of the input numbers because I wanted the output of both the adder and the multiplier be displayed by the same sequence of LEDs. It would have been easier to build two separate outputs, one for an adder and another one for a multiplier without using the bus transceivers and a toggle switch. However, the modern day calculators do not have separate displays for different mathematical operations and I didn't want mine to have multiple displays. Octal bus transceivers transfer 8-bit data in a chosen direction when activated. Since my circuit doesn't receive and output data using the same bus as an ALU does, the direction of the octal bus transceivers in my circuit always stayed the same, from the outputs of the DIP switches to the adder and a multiplier.

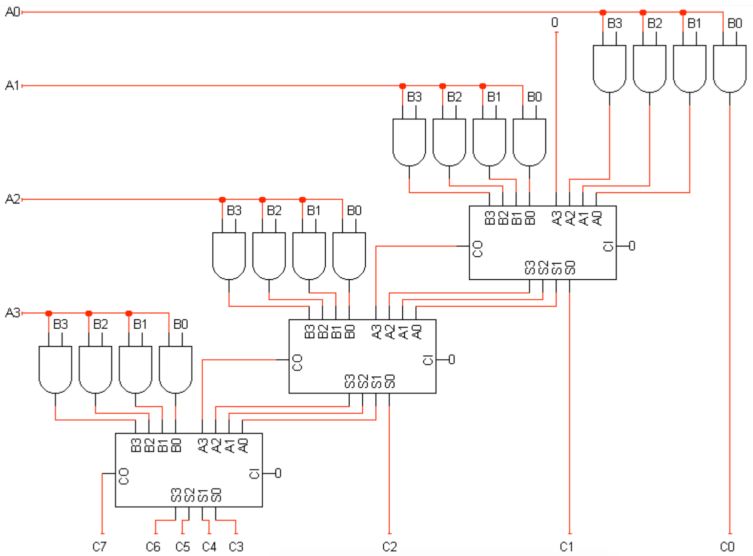
For the DIP switches, I used the ones Amin Jazaeri gave me. DIP switches let me easily switch between a binary 0 and a 1 by sliding the bar which acts as a switch between 0V and 5V.

Amin Jazaeri gave me the toggle switch. It's a regular toggle switch that connects ground to one of the two octal bus transceivers output-enable inputs depending on the switch's setting. The reason why it's ground and not 5V is because the octal bus transceivers are activated when output-enable inputs are at ground and deactivated when the inputs are any signal but ground.

Each resistor in the voltage divider at the output of each of the bus transceivers's data outputs has a value of 1KΩ, as I've already mentioned. These resistors are to lower the output voltage of the bus transceivers when they are deactivated so that the logic gates would take them as a low-level voltage signals.

Below is a logic gate circuit for addition and below it is one for multiplication. In choosing the design of these circuits I prioritized a small quantity of the required ICs in order to make the whole circuit more compact.





I used a function generator with a square wave of range [0V, 5V] to measure the time of computing different bits for both the addition and multiplication as Matthias Reinch suggested. I found that the computation of both addition and multiplication takes the same amount of time. However, I found that the time it takes to compute the most significant bit (the carry) of an addition circuit takes 24ns more than that to compute the least significant bit.

When I added 0000_2 to $000\#_2$ (where # is the output of the signal generator) and measured the time it takes to compute the value of the least significant bit, I got the following image on the oscilloscope. I measured the time between the input signal and the output be $\Delta t = 88.0ns$.

I then added 1111_2 to $000\#_2$ and measured the amount of time it takes to compute the most significant bit. I expected it to be a little larger since a carry makes the circuit go through all of the stages of the addition rather than output the answer right away as when you add 0000_2 to $000\#_2$ when there is no carry. I measured the time between the input signal and the output be $\Delta t = 112.0ns$, which confirmed my predictions. Below is a corresponding image of the oscilloscope screen.

All of the above measurements were done at about 1.87MHz frequency because I found this frequency to reveal the time delay of the calculations the best. The spec sheets of the logic gates state that a typical switching operation takes 12ns and that of the octal bus transceiver states that the propagation delay time and the output enable time together typically take 35ns. Thus, since the period of the wave is about 530ns, the time delays of separate components of the circuit didn't exceed the period of the wave and the circuit worked well. Furthermore, I noticed that the circuit began to experience some trouble computing correct results starting at about 9MHz, at which the period of the square wave is about 110ns. This is most likely not sufficient for all of the components of the circuit to have enough time to work properly.

Conclusion

I eventually built exactly what I planned to build and the circuit worked very well. When I built the sub-circuits that used many wires I had to take time and make sure that everything was connected as expected because there would be much trouble debugging the whole circuit when it's built if some connections were missing or incorrect. I thus spent a substantial portion of my time making sure that everything was connected as expected. That served me well since when I connected every sub-circuit together, I had very little debugging to do to make it work properly, and that debugging was concerning the connections between the dip switches, octal bus transceivers, and the toggle switch, not the adder and the multiplier circuits.

Overall, building a 4-bit calculator wasn't as challenging as I expected. The main challenge was wiring the right components together and then wiring the sub-circuits to make everything work in harmony. However, doing this project gave a better idea of how complex a real 32-bit ALU might be. If we had another project for this class, I would've chosen one that is more physics-oriented.

References

- Paul Malvino, Digital Principles and Applications, 7^{th}ed, Ch. 6.
- https://www.electronicshub.org/binary-multiplication/