The Binary Calculator

made using D-Flip Flops and Nand Gates

Owen Brake

2018

**Revised Binary Adder**

**Description/Purpose**

This circuit is a binary calculator/adder. This binary calculator can add 2 three digit binary numbers for a resulting four digit binary number. This may not seem impressive but the calculator does not solely add two binary numbers it also stores values which requires a number of flip flops and the tuning of a clock to the right frequency to operate. This circuit acts less like a simple ripple adder and is somewhat similar to early turing machines as it has the ability to save and load data, the only step left for it to be considered a turing machine would be for it to be able to load data and process that data as a program but that is beyond the scope of this project. This project shows how the simplest tasks that we take for granted can have an enormous complexity, the adding of two numbers up to a maximum of 13 and storing one of those values took 3 breadboards and cost a significant amount more than the cheap $5 calculator any person could buy at a convenience store.

**Modifications made to the original plan.**

Many modifications were made to the original circuit plan as the reality of dealing with integrated circuits did not completely match the total theory work done. Originally the circuit was to have a multiplier and contain 4 numbers to be added. This idea was scrapped, and the entire circuit was redesigned in Logisim. There were many factors that lead to this. One of the major factors was power, as more and more integrated circuits were being added the more current was drawn from the battery, the batteries were also constantly kept plugged in to test but this meant that the batteries would drain very quickly leading to batteries having a very poor performance operating at insufficient power. Another large factor was electric interference, the wires were very cheap and of a very poor quality, this meant that the wires came loose very easily and the more wires that were added the more fragile the circuit became requiring almost constant repair. Finally, the most important factor was space. To build a circuit with 3 inputs and that can add and store values took up a total of 3 breadboards. Adding a fourth input would mean a fourth breadboard and adding multiplication would take a massive amount of space spanning at least 2 more breadboards resulting as stated above in more fragility of the circuit itself.

**Future improvements**

In the future I would make improvements to the development of the circuit as well as in the design. The most important thing I would do is rather then using breadboards I would use perf board and solder all my joints in to make sure that components did not come loose. I would also use sturdy high quality wires to make the joints even more sturdy and making shorts less likely. Finally I would use a better power supply, the 9V system is helpful as it is simple but I would use in the future the regulated power supply that I made in grade 10 as it provides greater control over the voltage going into the circuit removing the need for the 5V regulator, the power supply also does not degrade resulting in a consistent performance.

**Using the circuit**

Preparation Steps

1. Lay the circuit on a very stable flat surface
2. Plug in the 9V battery into the nine volt battery clip
3. Gently move any loose wires out of the way from the button and switches

Once the above steps have been followed the circuit may now be used. The circuit has 3 switches, two larger sturdier switches and one smaller loose one, these switches act as the input for the number when the switches are moved to the left they are considered to be HIGH and correspond to a 1 on that digit in a binary number being added. Moving to the right results in LOW. There is also a button, when this button is pressed the values of the switches are saved in memory and are used as the second number to be added. There are 3 blue LEDs these three blue leds represent what is stored in memory. There are 4 more red leds and these leds represent the output for the circuit, so the summation of the number stored in memory and the number represented with the switches.

Operational Steps

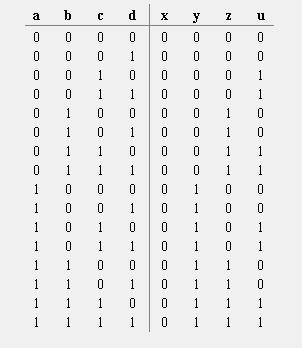
1. Set all the switches to LOW
2. Press the button (Save Data)
3. Set the switches to the desired value for the first number to be added
4. Press the button (Save Data)
5. View the first number and verify that is the correct value in the Blue LEDS
6. Set the switches to the desired value for the second number
7. View the Output represented in the red LEDS

**Parts Used**

* Numerous NAND Gates
* XOR Gates
* LEDS
  + 4 red
  + 3 blue
  + Others for testing purposes if multimeter not available
* 555 timer
* 2 10uF capacitors
* D flip flops
* Switches
* Buttons
* 5V regulator
  + With heat sink
* 9V battery
* 9V battery clip
* 3 bread boards
* A humongous amount of jumper wires
* Some resistors

**The D-Flip Flop**

The d-Flip flop was not taught in class but is critical to the function of this circuit. Most D-Flip flops have many inputs but in this case we only require 3. We need the clock input, the data input and the toggle. The way the d flip flop works is every time the clock oscillates then the value of the data input is stored and that storage is outputted. There is another pin on the south side of the flip flop and when this is true than values may be stored however when it is false no data is stored so we use this so our data cannot be overwritten.



Truth Table where a is digit 2, b is digit 1, c is digit 0 and d is the storing button in the notation. The same follows for x,y,z,u where x is digit 3, y is digit 2, z is digit 1 and u is digit 0.

**Full Logisim Circuit without sub circuits**

