Memo 1

Drexel University

To: Dr Christopher Peters
From: Mahad Faisal
Date: 1/19/2024
Re: Week 1 Hardware and Arduino Lab 1 Technical Memo

Purpose

The purpose of this week's lab assignments were to introduce students to fundamental circuit-building, the application and importance of logic gates in more efficient digital electronics, with the use of 74HCxx series integrated circuits and the Arduino MEGA 2560 . This lab has three components, the first being simulation on the Tinkercad platform followed by physical logic gates circuits with an Arduino MEGA and a circuit without logic gates and with the Arduino MEGA. After this lab I should be comfortable with the concept of logic gates and improve my physical circuit building skills as well.

Methodology

The first component of the project was to build the circuit on a web-based simulation platform, 'Tinkercad'. This was a very important and beneficial assignment for me as I had minimal experience with building Arduino-based circuits, especially in the context of logic gates. As seen in **Fig 1.1**, the circuit included:

- A power supply with voltage set to 5V and current set to 5A
- Two switches to test the logic gates
- An LED to visualize the output of the gates
- A logic gate (74HC00 NAND gate in the referenced picture)
- A resistor to control the flow of current.
- A multimeter to measure the voltage flowing through the logic gate
- Wiring to connect the parts

Using this design, I was able to simulate the following gates: 74HC00 AND, 74HC08 AND, and 74HC32 OR. This was possible as they had the same arrangement for input and output pins. However, the 74HC02 NOR gate had a different layout for input and output pins and every time I tried running the simulation, the logic gate broke down because of the current flowing through it exceeding its capacity. I was unable to figure out the cause of the problem and ignored it until after the hardware lab in which I was still unable to make the LED turn on.

After that failure I attempted to find a solution in Tinkercad which was futile until I opened the *Schematic View,* where I realized that the wires weren't connected to the right pins as the NOR. With that information I was able to successfully simulate the circuit in Tinkercad and then on the physical circuit.

For the hardware lab I used: (1) Long breadboard, (1) 74HC08 AND gate, (1) 74HC00 NAND gate, (1) 74HC02 NOR gate, (1) 74HC32 OR gate, (2) Buttons, (2) 5 kΩ resistors, (1) LED, (1) 1 kΩ resistor, and (1) Arduino Mega 2560. After attaching the components (*except only one gate at*

a time) to the breadboard, I used, mostly, jumper wires to connect components to each other and to the Arduino MEGA 2560 board, as seen in **Fig 1.2**. Before powering the system by connecting the board to my laptop, I verified all the connections as even a minor mistake would cause the logic gate to fail and break as seen in the Tinkercad simulation.

As expected, the circuit powered on smoothly and worked as expected. There were 4 possible permutations for the buttons (neither pressed, only the left A pressed, only the right B pressed, and both A and B pressed), of which I did all and recorded the video and the results. I repeated this process without any major changes but as I mentioned earlier on, I hit a roadblock on the final 74HC02 NOR gate. The LED wouldn't turn on no matter how many times I reset the system as I had ignored the flaw in the Tinkercad in the first part of the lab. After that I reattempted to fix the problem on Tinkercad and succeeded in doing so via the schematic view, as mentioned above.

I realized that instead of connecting the buttons to the input pins (4A and 4B) and the LED to output 4 as I had done with other gates, simply replacing the gates meant that the buttons would connect to Input 4 and Output 4B, and the LED would connect to Input 4 instead and that created an incomplete circuit and to fix it I had to look at the pin type rather than blindly connecting to the same pins repeatedly.

The last part of the lab did not involve logic gates but rather used coding in the Arduino IDE to replace the physics logic gate/ICs. 3 more LEDs and their respective resistors were also added to the circuit as seen in **Fig. 2.2**. The code [**Fig 2.0**] first prepares the system and starts the sketch with *void setup(),* and after that it assigns the pins on the Arduino as input and output e.g. the buttons (2 and 3) as inputs and the LEDs as outputs. The *void loop()* tells the system to keep running the following code to run as long as there is power. This is followed by two line that tell the processor to read the values from the buttons (inputs), and then the digital write commands replace the logic gates in function telling the system to power certain LEDs depending on the input e.g. power the LED when both buttons are being pressed/returning true values as a result of the AND pin created by line 16 `digitalWrite(8, buttonA_value && buttonB_value);`.

With this setup, I was able to record the results of the Arduino project and create a truth table based on that information **[Fig 2.1].**

Results

Figure 2.1 Arduino project results

A	B	F8	F9	F10	F11
0	0	0	0	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	1	1	0	0

The results of the Tinkercad simulation [**Fig 3**], the physical circuit [**Fig 4**], and the Arduino project [**Fig 2.1**], have the same results based on the concept. None of the gates have any deviations from the predictions/commands based on the function of the logic gate/s and Arduino code.

In the Arduino simulation, for the AND 74HC08 gate, the LED should only turn on when both switches A and B return a true value of '1' and this was seen in the results shown in Fig 3. The NAND 74HC00 gate should theoretically have only returned a true/high value for the LED to turn on in every scenario except when both switches are activated, and this is what was seen. Based on logic gate theory, the 74HC32 OR gate in a circuit with an LED will result in the LED being turned on whenever any or both switches are on and are therefore giving a high output value. Conversely, the 74HC02 gate should only give a true output only when neither switch is activated. Both of these hypotheses were accurately reflected on the results as shown below.

The lab book asked to measure the voltage across the logic gate from the multimeter and current going through the power source and although Dr Peters later clarified that the only thing you need for the Tinkercad section is the LED results, there was an interesting anomaly in those reading as the voltage across the NOR gate when only the right switch was turned on, was -5V number not seen anywhere else in the results which could be because I connected the power to the negative end which is likely.

The physical circuit results should have been the same as the results of the Tinkercad simulation as there was absolutely no difference between the core components of the circuit. As expected, my results show that there was no flaw in the set up of the physical circuit as the LED followed the patterns seen in the Tinkercad environment.

It is important to know that there was an error I encountered with the NOR gate in the second and first phase and even though I managed to figure it out at the end, I must not repeat the mistake of moving on to another step of the process before fully understanding and successfully executing the foundation because it can make the entire process more tedious and can have other negative consequences e.g. breaking the logic gate as I did.

When you combine the scenarios seen in both the Tinkercad and physical circuit, and replace logic gates with Arduino programming you get, essentially, the same results but all seen at the same time with the 4 LEDs. This part of the lab shows how technology has progressed since the creation the analog logic gate integrated circuits in the 1960s and how much more efficient circuits have become through the digitalization of the logic gates, while also showing how the creation of the logic gates was a necessary foundation for modern technology.

Figure 11.1

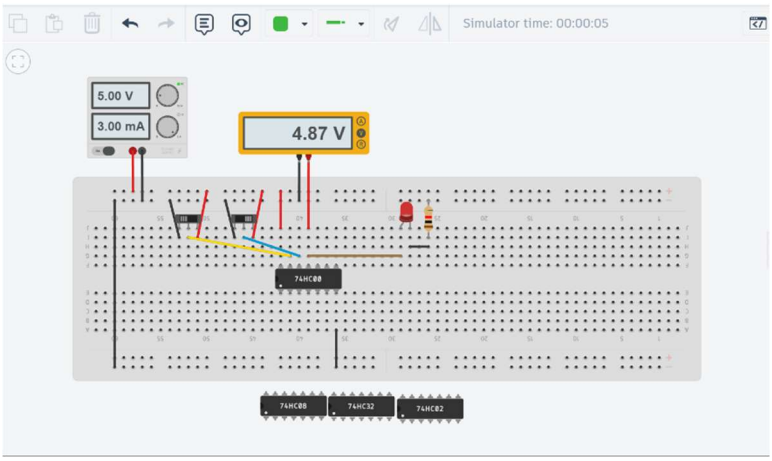

Figure 1.12 NAND gate physical circuit

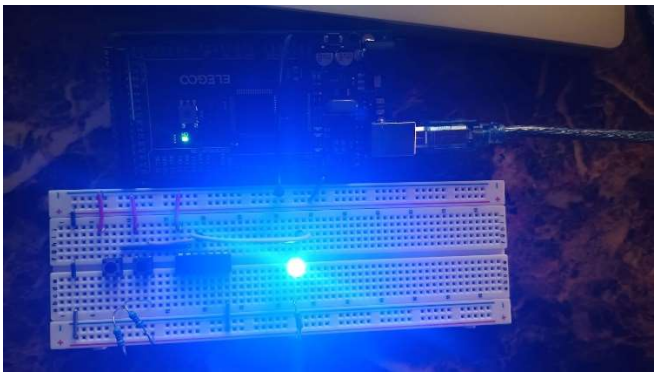

Figure 2.0 Arduino project code

LAB1C.ino

```
void setup() {
// Each digital pin used must be declared as
// INPUT or OUTPUT before being used
pinMode(2, INPUT); // Button A input
pinMode(3, INPUT); // Button B input
pinMode(8, OUTPUT); // AND pin
pinMode(9, OUTPUT); // OR pin
pinMode(10, OUTPUT); // NAND pin
pinMode(11, OUTPUT); // NOR pin
}

void loop() {
// Read in both buttons
int buttonA_value = digitalRead(2);
int buttonB_value = digitalRead(3);
digitalWrite(8, buttonA_value && buttonB_value);
digitalWrite(9, buttonA_value || buttonB_value);
digitalWrite(10, !(buttonA_value && buttonB_value));
digitalWrite(11, !(buttonA_value || buttonB_value));
}
```

Output

Fig 2.2 Arduino project circuit

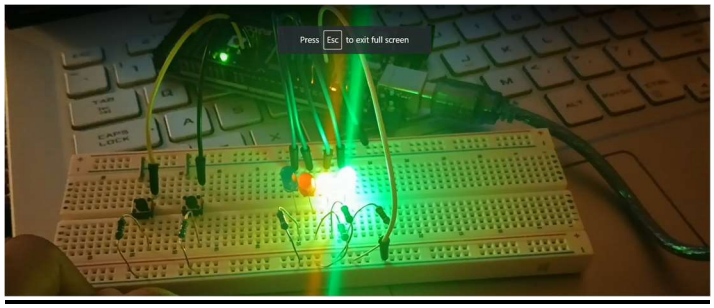

Fig 3 (Tinkercad) results

Both slide switches OFF (to the left)

Logic gate IC	Power supply current/**µA**	Multimeter Voltage/**V**	LED Status
74HC08	16.9	0	OFF
74HC00	3000	4.87	ON
74HC02	3000	4.87	ON
74HC32	16.9	5	OFF

Left slide switch OFF, right slide switch ON

Logic gate IC	Power supply current/**µA**	Multimeter Voltage/**V**	LED Status
74HC08	17.9	0	OFF
74HC00	3000	4.87	ON
74HC02	17.9	-5	OFF
74HC32	3000	4.87	ON

Left slide switch ON, right slide switch OFF

Logic gate IC	Power supply current/**µA**	Multimeter Voltage/**V**	LED Status
74HC08	17.9	0	OFF
74HC00	3000	4.87	ON
74HC02	17.9	0	OFF
74HC32	3000	4.87	ON

Both slide switches ON

Logic gate IC	Power supply current/**µA**	Multimeter Voltage/**V**	LED Status
74HC08	3000	4.87	ON
74HC00	18.9	0	OFF
74HC02	18.9	-5	OFF
74HC32	3000	4.87	ON

FIG 4 hardware lab results

7HC08 AND gate

A	**B**	**F**
0	0	0 *(OFF)*
0	1	0 *(OFF)*
1	0	0 *(OFF)*
1	1	1 (ON)

7HC00 NAND gate

A	**B**	**F**
0	0	1 (ON)
0	1	1 (ON)
1	0	1 (ON)
1	1	0 *(OFF)*

7HC02 NOR gate

A	**B**	**F**
0	0	1 (*ON*)
0	1	0 *(OFF)*
1	0	0 *(OFF)*
1	1	0 *(OFF)*

7HC32 OR gate

A	**B**	**F**
0	0	0 *(OFF)*
0	1	1 (ON)
1	0	1 (ON)
1	1	1 (ON)