Experiment 2 : Logic Gates

Digital Systems and Microcontrollers | Spring 21

Objective: To familiarize with the logic gates, De Morgan's law and to implement a full adder circuit using the gates

---

Instructions

- There are four parts in this Lab Experiment.
- Submit a separate tinkercad circuit project for each part.
- Make sure to answer the extra question in each part of the experiment in the report.
- Name the output LED's(SUM, CARRY et cetera) appropriately.
- The inputs to the circuit would be of the form `AB` when there are two inputs i.e `01` means $A = 0$ and $B = 1$ and `ABC` when there are three inputs. i.e `100` $\implies A = 1, B = 0, C = 0$.
- Sample circuit for each experiment is given in the Appendix for reference.

In this experiment, you will be introduced to some of the basic logic gates (NOT, AND, OR, NAND, NOR, XOR etc) available commercially in the IC (Integrated Circuit) form. Two families of digital ICs are commonly used: the TTL 74LSxx series and the CMOS CD 40xx series. Many of these ICs have 14 pins, and some have 16 or more. Two pins are used for power supply connections. Thus 12 pins are available in a 14-pin IC for gate inputs and outputs. A 2-input gate requires three pins per gate (two for inputs and one output), and so ICs that implement 2-input logic functions generally have 4 gates per IC. TTL ICs require a fixed d-c power supply voltage VCC having the nominal value of 5V and a tolerance of 5%. Thus these ICs are not guaranteed to function with VCC below 4.75V and VCC in excess of 5.25V can damage the IC. A 0.1μF ceramic capacitor should be connected between the VCC and Ground pins of each TTL IC to suppress spikes that may otherwise be created due to the current drawn from the power supply. Most CMOS ICs can work with 3V $\leq$ VCC $\leq$ 15V and do not require capacitors at each VCC pin.

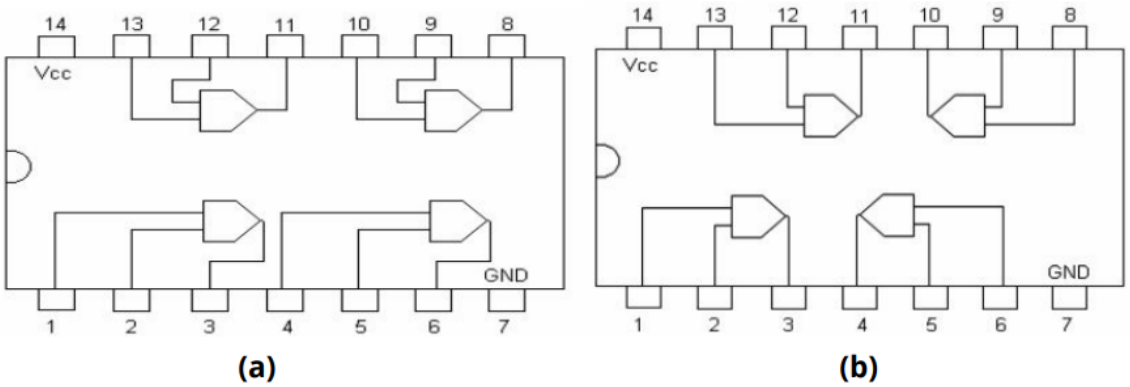

Figure 1: Two types of input connections for Quad-2-Input Gates a). TTL 74xx ICs b). CMOS CDXX ICs

Part A : Logic Levels

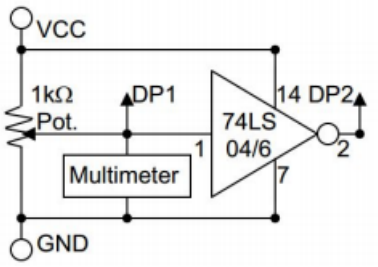

Figure 2: Part A

You may use any `NOT` gate IC available. `74HC04` is an IC which implements NOT gate and is available in Tinkercad.

1. Set up the circuit shown in Fig 2 on the breadboard and turn the potentiometer shaft to one end so that the multimeter reads 0V.
2. DP1 and DP2 are LEDs connected with appropriate resistors. DP2 must be glowing.
3. Now rotate the potentiometer shaft gradually up to the other end and tabulate the transitions in DP1 and DP2.

We are intending to find the tipping point voltage for the IC which it considers to be HIGH or LOW in both INPUT and OUTPUT pins. You may add another multimeter at the output of the NOT gate to monitor the output voltage as well. Compare these voltages with the specifications for binary logic level for a 0-5 V range.

$$0 \leq V_{OL} \leq 0.4, 0 \leq V_{IL} \leq 0.8, 2.0 \leq V_{IH} \leq 5.0, 2.4 \leq V_{OH} \leq 5.0$$

Part B : Verifying the truth table of Logic Gates

The goal of this part of the experiment is to take input from the serial monitor and verify the truth table of logic gates: (NOT, OR, AND, XOR, NOT, NAND) using TTL 74XX family of ICs.

1. Place the IC on breadboard and give V_{cc} and Gnd connection to it.
2. Take inputs from the Serial Monitor for values of A and B and route them to the input pins of the IC.
3. Connect an LED with appropriate resisitor to the output of the GATE.
4. Note the output of the chosen gate for different values of input in a truth table.

You can take an input of 0 to 10 from the Serial Monitor to a variable `x` the following way:

```
int x;
void setup() {
    Serial.begin(9600); //opens serial port, sets data rate to 9600 bps
}
void loop() {
    if(Serial.available() > 0) {
        x = Serial.read(); // x would be an integer between 0 and 255
        // depending on the ascii value of the character read
        x = x - '0'; // Subtracting ascii value of 0 from x.

    }

    //Use the value of x in code.
}
```

Part C : De Morgan's Law

De Morgan's theorems state that $(A + B)' = A' \cdot B'$ and $(A \cdot B)' = A' + B'$. Verify these theorems by proceeding step by step as follows:

1. Set up a circuit consisting of two NOT gates and one AND gate to perform function $Y = A' \cdot B'$,
2. Obtain the truth table of this circuit by noting the output of the function for different values of A and B. Verify that the output of the function is same as that of the NOR gate.
3. Repeat steps 1 and 2 using an OR gate instead of an AND gate to verify that the truth table is same as that of the NAND gate.

How would you realise the above circuit if you have only NAND gates instead of NOT gates? i.e How would you use NAND gates to perform function of NOT gates?

Part D : Binary Full Adder

A binary Full Adder adds two bits A and B along with a carry in C to generate SUM and CARRY bits as output. The first step to achieve this is to make a binary Half Adder, which adds two binary inputs A and B to give a sum S1 and a carry C1 according to the following Boolean expressions for the outputs S1 and C1:

$$S1 = A' \cdot B + A \cdot B' = A \oplus B \text{ and } C1 = A \cdot B$$

Another Half Adder is then used to generate the final SUM by adding the third binary input C to the S1 bit generated by the first Half Adder:

$$SUM = S1 \oplus C$$

The carry bit generated by this Half Adder is given by

$$C2 = S1 \cdot C$$

.

Write down the complete truth table of a Full Adder, including columns for the intermediate outputs S1, C1 and C2. Find out the logic for generating the final CARRY output from C1 and C2. As XOR and AND gates are going to be used for the Half Adders, try to obtain a logic for CARRY using the same type of gates, so that the complete realisation of the Full Adder is possible without necessitating a third IC.

1. Set up the circuit of a Half Adder using an XOR gate and an AND gate. Apply the inputs A and B from two input pins and observe the outputs S1 and C1 on two LED displays for all combinations of the inputs. Tabulate these values and verify the operation of the Half Adder.
2. Set up another Half Adder using another XOR and another AND gate out of the same ICs used in step 1, and connect the C input and the S1 output generated by the first Half Adder as its inputs to generate the final SUM output and the C2 output.
3. Generate the final CARRY output from the intermediate carry outputs C1 and C2, using the unused gates in the XOR and AND ICs deployed so far
4. Verify the truth table experimentally by applying the inputs A, B and C through three input pins and displaying the S1, C1, C2, SUM and CARRY outputs.

Appendix

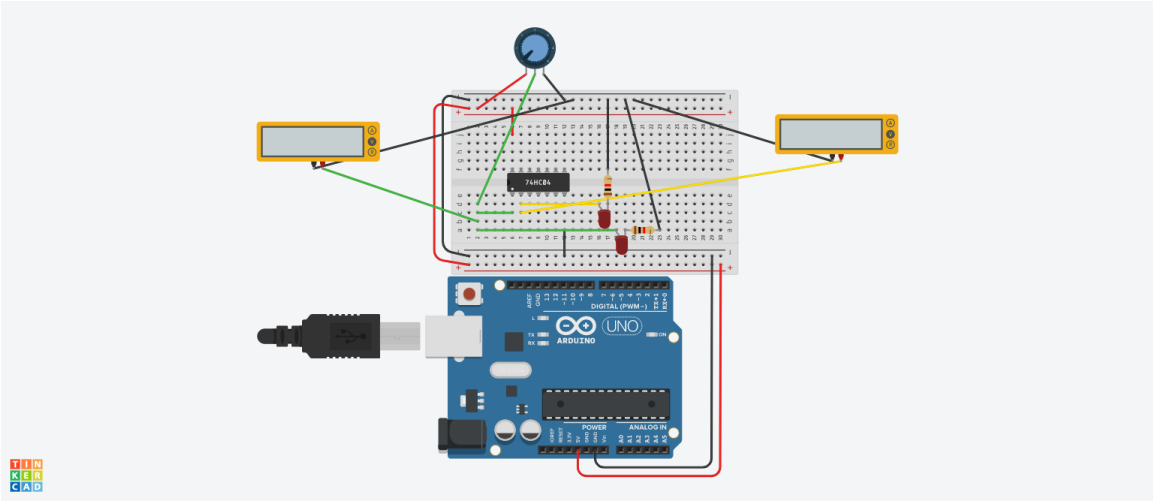

Figure 3: Part A: Finding Logic Levels

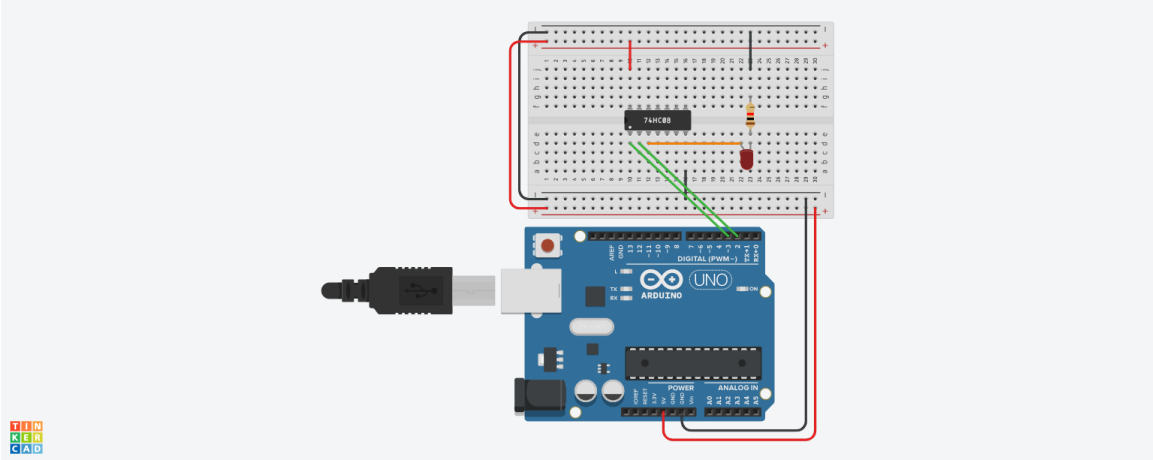

Figure 4: Part B: Verifying AND Gate

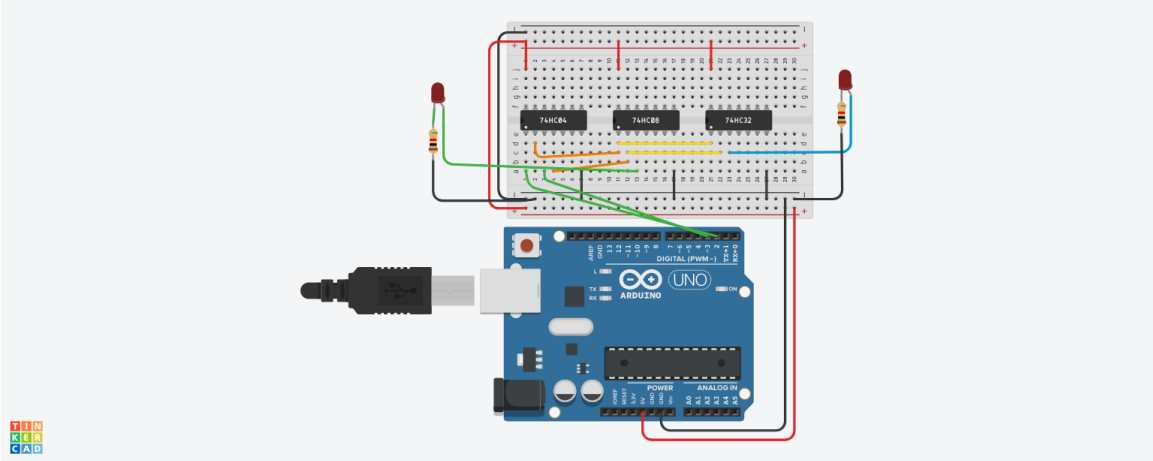

Figure 5: Part C: De Morgan's Law

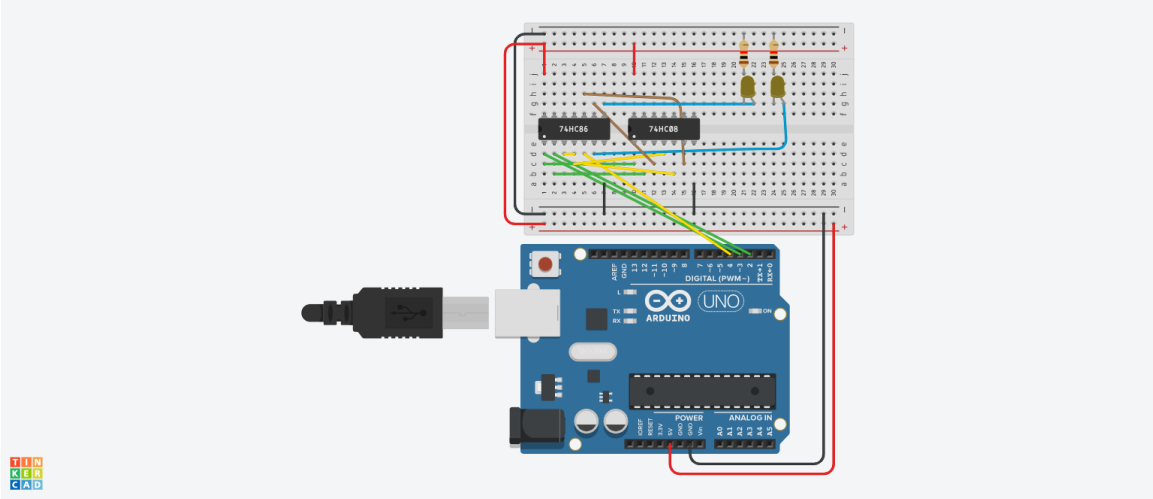

Figure 6: Part D: Binary Full Adder