**Score: \_\_\_\_\_**

**Lab02 – Integrated Circuits and Computation**

COMP256 – Computing Abstractions

Dickinson College

Spring 2023

Prof. Grant Braught

**Names:**

**Introduction:**

In the first lab you learned about using breadboards to build circuits, how to use a voltmeter to measure logic levels and you built a CMOS transistor circuit that performed a basic computation (the NOT gate). In this lab we will move up the hardware abstraction hierarchy to build circuits using logic gates contained in integrated circuits (i.e. *chips*). Using these chips, you’ll design and build a circuit that can do perform a more complex computations – it will add unsigned binary numbers.

**Logic Gates and Integrated Circuits:**

Circuits that use logic gates are usually built using *integrated circuits* (IC), each of which contains multiple logic gates. These ICs come in the form of a *dual inline package* (DIP) or what we commonly just call *a chip*. The figure below shows a picture of a DIP chip and a schematic diagram for a CMOS hex inverter (i.e. 6 logical inverters or NOT gates) packaged onto a single IC.

A picture containing text, electronics, circuit

Description automatically generated

A few important points about the IC chips that we will be using:

* 74HC04 is just the part number for the CMOS hex inverter chip. You could use your favorite search engine to find this part number if you wanted to buy one. The chips for different gates have different numbers.
* The metal contacts on the chip (14 of them, 7 on each side) are called *pins*.
* The schematic diagram is often called a *pin-out* for the chip, because it shows how each of the pins is connected to the logic gates.
* The little notch on the chip (on the left side above) lets you know which end is which so that you can match it up with the pin-out diagram. Some chips and TINKERCAD will use a small white dot instead of a notch.
* The pin labeled Vcc must be connected to the + terminal of the power source (i.e. 1), and the pin labeled GND (ground, i.e. 0) must be connected to the – terminal of the power source.

🔑 1. Use the pin-out diagram for the 74HC04 CMOS hex inverter to fill in the table below with the pin numbers that correspond to the statement:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | **Description** | **Pin Number(s)** |  |
|  | Connected to the + terminal (1) of the battery. |  |  |
|  | Connected to the – terminal (0) of the battery. |  |  |
|  | Is an input to a NOT gate. |  |  |
|  | Is an output of a NOT gate. |  |  |
|  |  |  |  |

**Integrated DIP Circuits and Bread Boards:**

The bread board that you learned about in last week’s lab is specially designed to work with DIP chips, like the 74HC04 chip. It is not required viewing, but you might find it useful to watch the end (starting at 11:00) of Ben Finio’s *How to Use a Breadboard* video that shows how to place a DIP on a breadboard.

* <https://youtu.be/6WReFkfrUIk?t=661> (1:20)

**Wiring a NOT Gate:**

🔑 2. Create the following circuit in TINKERCAD and name it Lab02-NotGate. *Your answer for this question will be your TINKERCAD circuit.*

A picture containing clock, meter, game

Description automatically generated

Paste a link to your TINKERCAD circuit here.

🔑 3. Complete the above circuit by connecting:

* the output of the slider switch (the middle pin) to the input of any one of the not gates on the 74HC04 chip.
  + Recall that when wired as above, with the slider switch to the left its center contact is connected to ground (i.e. logical 0). Conversely, with the switch to the right its center contact is connected to +3V (i.e. logical 1).
* the negative lead of the voltmeter to ground (as always).
* the positive lead of the voltmeter to the pin that is the output of the NOT gate that you used.

*Your answer for this question will be your TINKERCAD circuit linked above.*

🔑 4. Simulate your circuit. Using the slide switch, change the input to the NOT gate use the voltmeter to complete the table below. You can refer back to Lab01 if you need a refresher on how voltages are converted to logic values.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | **Switch  Position** | **Logic Value at NOT Input** | **Voltage at NOT Output** | **Logic Value at NOT Output** |  |
|  | Left |  |  |  |  |
|  | Right |  |  |  |  |
|  |  |  |  |  |  |

5. Check whether your results in the table above agree with the behavior of a NOT gate. If they do great, time to move on. If not, revisit questions #3 and #4 to resolve any disagreement.

No answer is required here. Just ensure that your circuit works correctly before proceeding.

**Another Logic Gate IC:**

The 74HC is a series of chips. This means that there are other chips, with the 74HC prefix, but different suffixes. Each 74HC chips contains a different type of logic gates (e.g. AND/OR/NAND/NOR/XOR). For example, the 74HC08 is a quad AND gate - 4 AND gates packaged into a single DIP IC. The pin-out for the 74HC08 is shown here:



🔑 6. Create a new circuit (i.e. do not modify your previous one) in TINKERCAD named

Lab02-AndGate to test the behavior of the 74HC08. This circuit should be similar to the one you created above for the NOT gate. However, because the AND gate has 2 inputs you’ll need to use 2 slider switches, one for each input, as shown below:



Paste a link to your TINKERCAD circuit here.

🔑 7. Experiment with your circuit and fill in the truth table below.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
|  | **A Switch  Position** | **B Switch  Position** | **A Input Logic Value** | **B Input Logic Value** | **Voltage at AND Output** | **Logic Value at AND Output** |  |
|  | Left | Left |  |  |  |  |  |
|  | Left | Right |  |  |  |  |  |
|  | Right | Left |  |  |  |  |  |
|  | Right | Right |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

8. Check that the results in the table above agree with the expected behavior of an AND gate. If so great, time to move on. If not, revisit questions #6 and #7 to resolve any disagreement.

No answer is required here. Just ensure that your circuit works correctly before proceeding.

**Building a Circuit:**

🔑 9. Imagine we want to build a circuit that computes the following Boolean function:

a. Draw a logic circuit using NOT gates, AND gates, and a NOR gate that directly implements the above logic expression. Do not simplify the expression before implementing it.

b. To be able to test your circuit you will need to know what its expected behavior is. Complete the truth table below for this logic expression so that you know the expected value of Z for each of the possible input combinations. Appropriate intermediate values have been added to the table for you.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  | 0 | 0 | 0 |  |  |  |  |  |  |
|  | 0 | 0 | 1 |  |  |  |  |  |  |
|  | 0 | 1 | 0 |  |  |  |  |  |  |
|  | 0 | 1 | 1 |  |  |  |  |  |  |
|  | 1 | 0 | 0 |  |  |  |  |  |  |
|  | 1 | 0 | 1 |  |  |  |  |  |  |
|  | 1 | 1 | 0 |  |  |  |  |  |  |
|  | 1 | 1 | 1 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

To build the circuit you drew above you will need a NOT, AND and NOR gates. The pin-out diagrams for all of the 74HC series two input logic gates are shown below. **Notice that the orientation of the NOR gates are reversed with respect to all of the other chips**, so be careful if when you use that one!



🔑 10. Create a new circuit in TINKERCAD and name it Lab02-LogicCircuit. Build the circuit that you drew for question #9a using the appropriate IC chips. Use three slider switches for the inputs A, B and C. Please arrange these switches as A, B, C from left to right in your circuit so that I know which is which when I test your circuit. **I recommend using a full-size Breadboard** (i.e. not the Small Breadboard) to give you plenty of room for all of the components.

Paste a link to your TINKERCAD circuit here.

🔑 11. Now test your circuit to ensure that it works correctly. Connect a voltmeter to measure the voltage at the output of your circuit. Simulate the circuit and then use the slide switches to set the inputs to each possible value. For each one, fill in the logic value of Z (0 or 1) in the table below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | **A** | **B** | **C** | **Z** |  |
|  | 0 | 0 | 0 |  |  |
|  | 0 | 0 | 1 |  |  |
|  | 0 | 1 | 0 |  |  |
|  | 0 | 1 | 1 |  |  |
|  | 1 | 0 | 0 |  |  |
|  | 1 | 0 | 1 |  |  |
|  | 1 | 1 | 0 |  |  |
|  | 1 | 1 | 1 |  |  |
|  |  |  |  |  |  |

12. Check that the truth table produced from your circuit matches what you expect based on the truth table you generated earlier. If so great, time to move on. If not, revisit questions #9, #10 and #11 to resolve any differences. *No answer is required here.*

No answer is required here. Just ensure that your circuit works correctly before proceeding.

**Designing a Circuit - Adding 1-bit Binary Numbers:**

In the last section you started from a logic expression to produce a circuit. However, the design process usually starts from a truth table. From that truth table a logic expression in Sum-of-Products (SOP) form is generated, that expression is simplified to reduce the cost or propagation delay of the circuit, and then from the simplified expression a circuit is built.

In this section, you will use this process to design and build a circuit that is able to add three one-bit unsigned binary numbers.

Given three one-bit unsigned binary numbers, let’s call them X, Y and Z, if we add them together, we will get a two-bit result where one bit is the sum (S) and the other bit is the

carry (C):

X

Y

+ Z

C S

For example, if X=0, Y=1 and Z=1 then the addition would be:

0

1

+ 1

1 0

When we add 0 + 1 + 1 we get a sum (S) of 0, and a carry (C) of 1. Taking the C and S together we have the result 102 = 210, in unsigned binary, which is what we should expect from adding 0 + 1 + 1. Obviously, using different values of X, Y and Z will give different results for S and C.

🔑 13. Complete the truth tables below to show what the value of S and the value of C should be for all possible values of X, Y and Z. **Have the instructor check your truth tables before going on.**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  | **X** | **Y** | **Z** |  | **C** |  | **S** |  |
|  | 0 | 0 | 0 |  |  |  |  |  |
|  | 0 | 0 | 1 |  |  |  |  |  |
|  | 0 | 1 | 0 |  |  |  |  |  |
|  | 0 | 1 | 1 |  |  |  |  |  |
|  | 1 | 0 | 0 |  |  |  |  |  |
|  | 1 | 0 | 1 |  |  |  |  |  |
|  | 1 | 1 | 0 |  |  |  |  |  |
|  | 1 | 1 | 1 |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

🔑 14. Write an SOP expression for C.

🔑 15. Write an SOP expression for S.

16. **Have the instructor check your SOP expressions at this point.**

No answer is required here. But have your instructor check your expressions so that you do not continue with an incorrect expression.

17. Using the Boolean identities, simplify the expression for C as much as you can. Show the steps you use and label the identity used at each step.

18. Using the Boolean identities, simplify the expression for S as much as you can. Show the steps you use and label the identity used at each step.

🏆 As an extra challenge, simplify the expression for S so that it uses only two XOR gates.

19. Draw a schematic diagram using logic gate symbols for a logic circuit that computes C.

20. Draw a schematic diagram using logic gate symbols for a logic circuit that computes S.

21. Create a new circuit in TINKERCAD and name it Lab02-BinaryAdder. Again, **I recommend using a full-size Breadboard.** Setup three slider switches as the inputs X, Y, Z. Please label these switches using labels as X, Y, Z in TINKERCAD so that I know which is which when testing your circuit. Then build the circuits you drew for both C and S.

**Note that you can, and should, use the output of each switch as an input to both the S and C circuits.** For example, you can connect two wires to the center pin of the X switch and route one to each circuit as shown below. That way the same value of X will be provided to both the C and the S circuit.



Paste a link to your TINKERCAD circuit here.

22. Now we could use voltmeters to observe the outputs C and S but it’s kind of fun to see the results of your additions light up. So instead of using a voltmeter for testing we will use a *Light Emitting Diode (LED)*. Connect an LED to each of the outputs as shown in the image below. The green wires come in from the C output and the S output and connect to the LEDs. **Note: The orientation of the LED is important and is indicated by the long (bent) leg or short (straight) leg.** Label your C and S outputs in TINKERCAD so that it is clear which is which.

Each LED will light up if the value connected to long leg is a 1. So, for example in the images below the C is a 1 and the S is a 0. This image shows the binary number 102 = 210.



*Your answer for this question will be your TINKERCAD circuit linked above.*

23. Using the switches for X, Y and Z, test that your circuit correctly produces the binary sum of X + Y + Z for all of the possible input combinations. If it does, congratulations! You have just built a circuit that performs addition!! I know… pretty cool! If you aren’t quite getting the right results yet, revisit your work from questions #17 to 22 to track down the mistake.

No answer is required here. But you should ensure that your circuit is functioning correctly before going on.

**Optional Extra: Adding More Bits:**

🏆🏆 24. If you are enjoying building circuits and are looking for an extra challenge, try this. Create a new circuit called LAB02-TwoBitAdder and build a circuit that can add two 2-bit unsigned numbers producing a 3-bit result. For example:

1

1 1 310

+ 1 0 + 210

1 0 1 510

You could use the design process we used above, but with 4 inputs instead of 3. Or, if you are clever about it, you can do this addition using two copies of the circuits you built earlier. Puzzling out what is meant by the following generalization of the above addition problem and the example below will point you in the right direction:

C0 0 0 0

Y1 Y0  1 1 3

+  Z1 Z0 + 1 0 + 2

C1 S1 S0  1 0 1 5

Hint: The subscripts indicate which copy of the circuit is being used. E.g. Y0, Z0, S0 and C0 are in one copy, and Y1, Z1, S1 and C1 are in the other.

Paste a link to your TINKERCAD circuit here.

Optional: To help me improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this Lab (include the 2-hour lab period in your total)?

b. Please comment on any particular challenges you faced in completing this Lab.