# INTRODUCTION TO DIGITAL SYSTEMS

SYSC2310 (Fall 2020)
Dr. Mostafa Taha

## Lab 3: Gate-Level Minimization

## Objectives

- Get familiarized with simplifying using K-maps.
- Simplify completely specified and incompletely specified functions.
- Build a BCD to 7-Segment decoder.

## Relation to course outcomes

The work in this lab is related to the two following course outcomes:

- Manipulate or design basic combinational operators.
- Apply analysis skills to correctly describe the behavior of given combinational logic circuits.

## Preparation for the lab

1. Complete part (1.A) and part (2.A) in the attached answer sheet. Your answer to these pre-lab parts **will be graded**.
2. Under your ‘sysc2310lab/lab3’ folder, create two folders ‘Ex1’ and ‘Ex2’, to save your work during this lab. Remember, by the end of the term, you are required to have the files for all the exercises in every lab. TAs and lab technicians will not be able to recover any deleted file, or any file that you have overwritten by mistake.

## Exercise 1: Design an Arbitrary Function.

**(1.A)** The circuit has four inputs and two outputs. The inputs $\boldsymbol{A3, A2, A1, A0}$ represent a 4-bit number from 0 to 15. Output $\boldsymbol{P}$ should be TRUE if the number is a prime number. Output $\boldsymbol{D}$ should be TRUE if the number is divisible by 3 or 4, i.e., {0, 3, 4, 6, etc.).

1. Write the truth table for this circuit.
2. Use K-maps to find simplified Boolean equations for each of the outputs.
3. Sketch a circuit that implements the identified equations using AND, OR and NOT gates.
4. Count the number of AND and OR gates used in the circuit. Differentiate between gates with a different number of inputs. A 2-input AND gate has a smaller implementation area than a 3-input AND gate. Please count 2-input AND gates in a different category from 3-input AND gates, 4-input AND gates, and so on.

**(1.B)** Implement and test the functionality of the circuit in Logisim.

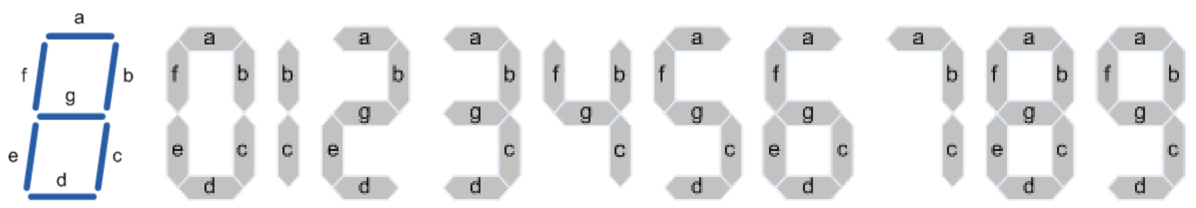
1. Using the Computational Analysis in the Window Tab, input the name of your input variables $\boldsymbol{A3, A2, A1, A0}$ and the corresponding outputs $\boldsymbol{P}$ and $\boldsymbol{D}$.
2. Input the truth table in the Table tab.
3. Verify the Boolean expression for each of the outputs in Expression tab. Compare them to your results. Identify any differences between your results and the ones obtained in Logisim.
4. Visualize the K-Map for each of the segments by clicking on the Minimized tab. Compare them to your results. Identify any differences between your results and the ones obtained in Logisim.

5. Click “Build Circuit”, and then name the circuit. Connect two LEDs from the Input/Output folder in Logisim, a green LED to the output $\boldsymbol{P}$ and blue LED to the output $\boldsymbol{D}$. For using LEDs, refer to: http://www.cburch.com/logisim/docs/2.5.0/en/libs/io/led.html
6. Enable logging to ‘Lab3_Ex1_log.txt’ and use the poke tool ( ) to change the values of the input bits. Notice output LEDs.
7. Save the circuit as ‘Lab3_Ex1.circ’.

## Exercise 2: BCD to 7-Segment Decoder

**(2.A)** A BCD to 7-Segment display decoder takes a 4-bit data input, $[\boldsymbol{D3}, \boldsymbol{D2}, \boldsymbol{D1}, \boldsymbol{D0}]$, and produces seven outputs to control the LEDs of the 7-segment display to show a digit from 0 to 9. The seven outputs are often called segments $\boldsymbol{a}$ through $\boldsymbol{g}$, or $\boldsymbol{Sa}$ to $\boldsymbol{Sg}$.

https://www.electronics-tutorials.ws/combination/comb_6.html



1. Write the truth table for the decoder. Mark with a “1” the segments that are activated and with “0” those who are not activated for each output. Your aim is to display the digits from 0 to 9, so all the combinations after 1001 should be Don’t Care Conditions.
2. Use K-maps to find simplified Boolean equations for each of the outputs.
3. Sketch a circuit that implements the identified equations using AND, OR and NOT gates.
4. Count the number of AND and OR gates used in your circuits. Differentiate between gates with a different number of inputs.

**(2.B)** Implement and test the functionality of the 7 segment decoder in Logisim.

1. Using the Computational Analysis in the Window Tab, input the name of your input variables $[\boldsymbol{D3}, \boldsymbol{D2}, \boldsymbol{D1}, \boldsymbol{D0}]$ and of the corresponding outputs $\boldsymbol{Sa}$ to $\boldsymbol{Sg}$.
2. Input the truth table in the Table tab.
3. Verify the Boolean expression for each of the outputs in Expression tab. Compare them to your results.
4. Visualize the K-Map for each of the segments by clicking on the Minimized tab. Compare them to your results.
5. Click “Build Circuit”, and then name the circuit. Compare the circuit with your sketch implementation in part (2.A) and identify any differences.
6. Add a 7-Segment display from the Input/Output folder and connect the ports of your circuit. To properly connect the 7-Segment, refer to the help: http://www.cburch.com/logisim/docs/2.3.0/libs/io/7seg.html
7. Enable logging of the input $(\boldsymbol{D3}, \boldsymbol{D2}, \boldsymbol{D1}, \boldsymbol{D0})$ and the outputs $\boldsymbol{Sa}$ to $\boldsymbol{Sg}$ to a file named ‘Lab3_Ex2_log.txt’, while using radix-10.
8. Simulate the circuit by using the poke tool, to test all the possible inputs.
9. **What do you expect on the 7-segment display when you enter decimal numbers above 9?**
10. Save the circuit as ‘Lab3_Ex2.circ’.

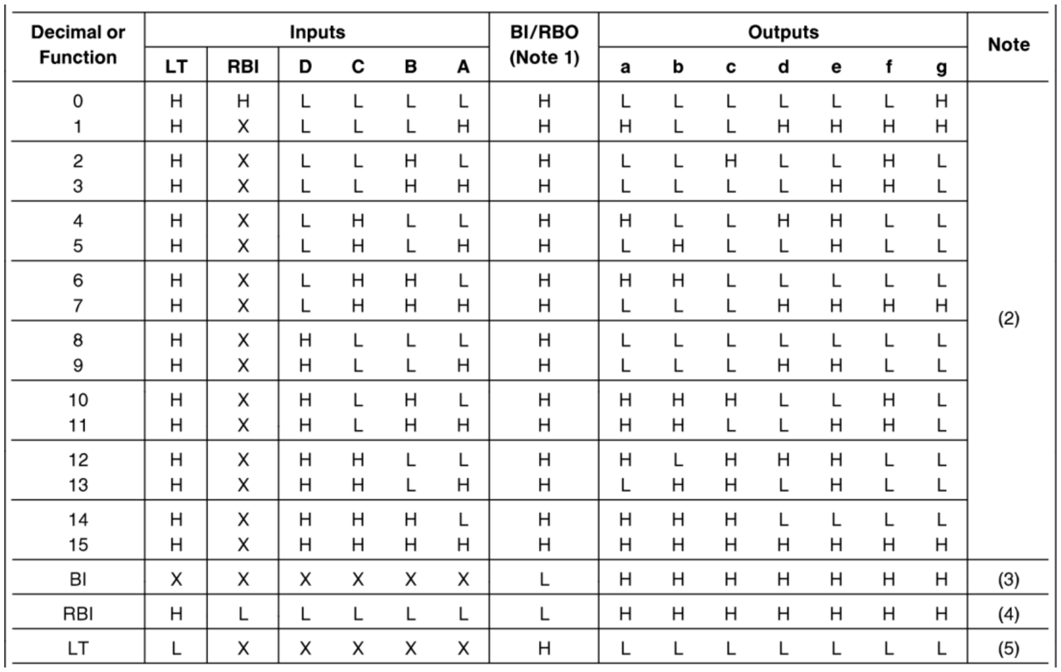
## Exercise 3: Analyzing the IC 7447 BCD to 7-Segment Decoder.

The truth table and logic diagram of an actual BCD to 7-segment decoder that you can purchase is shown below. The IC is called 7447 and can be purchased from: https://www.digikey.ca/en/products/detail/texas-instruments/SN7447AN/1575197 (or any similar electronic outlets). This design was first introduced in early 1970's.

The 7447 features active low outputs designed for deriving common-anode LEDs. In common-anode LEDs, the anode is shared among all the LEDs and is connected to the power source. The control signals are connected to the cathode of each LED. The LED will be ON whenever the control signal is Low. In other words, whenever the output $a$ in the circuit below is Low, the LED segment $Sa$ is expected to be ON. Whenever the output $a$ is High, the LED segment $Sa$ is expected to be OFF.

- On the attached logic diagram of the 7447, find the logic output **of each gate** in the design (not just the final output mentioned in the truth table) in the following cases:
    1. [LT RBI D C B A] = [H H L L L L]
    2. [LT RBI D C B A] = [H H L L L H]
    3. [LT RBI D C B A] = [H L L L L L]
    4. [LT RBI D C B A] = [L H L L L L]

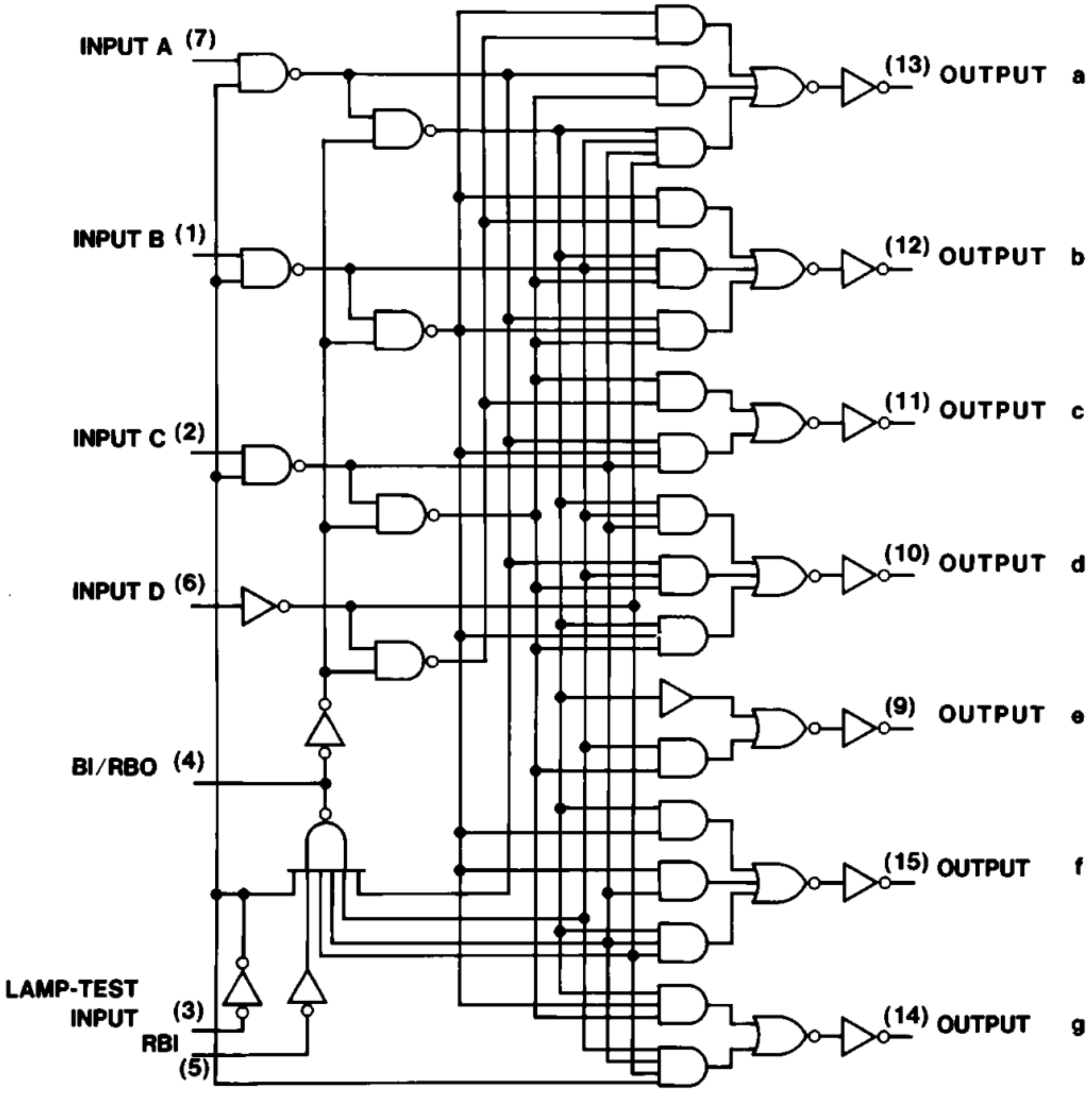
  Note that the circuit may be designed as 'Don't Care' in certain cases, but as an actual digital circuit, all the inputs must be either High or Low.



| Decimal or Function | Inputs | | | | | | BI/RBO (Note 1) | Outputs | | | | | | | Note |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | LT | RBI | D | C | B | A | | a | b | c | d | e | f | g | |
| 0 | H | H | L | L | L | L | H | L | L | L | L | L | L | H | (2) |
| 1 | H | X | L | L | L | H | H | H | L | L | H | H | H | H | |
| 2 | H | X | L | L | H | L | H | L | L | H | L | L | H | L | |
| 3 | H | X | L | L | H | H | H | L | L | L | L | H | H | L | |
| 4 | H | X | L | H | L | L | H | H | L | L | H | H | L | L | |
| 5 | H | X | L | H | L | H | H | L | H | L | L | H | L | L | |
| 6 | H | X | L | H | H | L | H | H | H | L | L | L | L | L | |
| 7 | H | X | L | H | H | H | H | L | L | L | H | H | H | H | |
| 8 | H | X | H | L | L | L | H | L | L | L | L | L | L | L | |
| 9 | H | X | H | L | L | H | H | L | L | L | H | H | L | L | |
| 10 | H | X | H | L | H | L | H | H | H | H | L | L | H | L | |
| 11 | H | X | H | L | H | H | H | H | H | L | L | H | H | L | |
| 12 | H | X | H | H | L | L | H | H | L | H | H | H | L | L | |
| 13 | H | X | H | H | L | H | H | L | H | H | L | H | L | L | |
| 14 | H | X | H | H | H | L | H | H | H | H | L | L | L | L | |
| 15 | H | X | H | H | H | H | H | H | H | H | H | H | H | H | |
| BI | X | X | X | X | X | X | L | H | H | H | H | H | H | H | (3) |
| RBI | H | L | L | L | L | L | L | H | H | H | H | H | H | H | (4) |
| LT | L | X | X | X | X | X | H | L | L | L | L | L | L | L | (5) |

Truth Table of the IC 7447 (National Semiconductor Corporation)

* BI (blanking input): When forced to Low, all the outputs are set to High (LEDs are off). Used to entirely blank the LEDs or control their intensity by using a high frequency pulse.
* RBI (ripple-blanking input): Does not affect any input symbol other than the 0. High → 0 is displayed normally (like displaying a simple 0 on your calculator). Low has two effects → a) 0 not displayed (like displaying '0005' on your calculator as '5', wihout the leadign zeros), b) reset the RBO (ripple-blanking output) to Low, which shall be connected to the RBI of the next 7-segment to subress the display of all the following zeros.
* LT (Lamp Test): as the name implies, reset all the outputs to Low (all LEDs are ON).



Logic Diagram of the IC 7447 (National Semiconductor Corporation)

## Demonstration:

- In order to get full credit, complete all exercises and be prepared to answer questions asked by the TA during the one-on-one meeting.
- Before you sign out of the lab, even if you could not complete the lab, make sure you sign-out with the TA.
- If you could not complete the lab during the scheduled lab time, finish up the lab on your own time. In this case, the demonstration points will be reduced in accordance to the percentage of work completed.
- If the student is absent or does not show the TA the work completed, 0 points are attributed for this part.

# INTRODUCTION TO DIGITAL SYSTEMS

SYSC2310 (Fall 2020)
Dr. Mostafa Taha

## Lab 3: Answer Sheet

## Exercise 1: Design an Arbitrary Function.

$P =$

$D =$

The total number of AND and OR gates is:

## Exercise 2: BCD to 7-Segment Decoder (IC 7447)

$S_a =$

$S_b =$

$S_c =$

$S_d =$

$S_e =$

$S_f =$

$S_g =$

The total number of AND and OR gates is: