## Purpose

This lab is designed for you to demonstrate and validate your understanding of topics learned from chapter 3

You are asked to design and simulate three logic circuits using Logisim.

- **3 or 4 same inputs**
- **Even Parity Generator**
- **BCD to 7 Segment Display**
- **BCD with Error Output – *Extra Credit***

## Before you begin

For this lab work, we expect you have completed Lab 0 and Lab 1 entirely and your lab environment is ready to start this lab assignment. If you haven't, please refer to the Lab 0 instructions. Again, you are allowed to use any working lab environment and in fact, you may store your latest work in a secondary storage (like a thumb drive) or cloud system so that you can work from the latest copy from any environment at your convenience.

Also, you may create a new folder for Lab 3 to store all your circuits there. If you are using Mac, please place that folder in your home directory.

# 1. Part 1: *3 or 4 same inputs* circuit.

1. The circuit will have 4 inputs (`w, x, y, z`) and will output a `1` when at least 3 of the 4 inputs have the same value. Note this does not say if the inputs are a `1` or a `0`, just that they are the same.

2. First, create a truth table showing the relationship between 4 inputs and the outputs.

   - Write your truth table in Excel and save the file as `Lab3Part1.xlsx`. You will be submitting this.

3. Then use a K-Map or algebraic manipulation to achieve the optimal circuit for the truth table.

   - Write the map in the same Excel file you use for the truth table, `Lab3Part1.xlsx`.

4. Create a Logisim project for this circuit and save it as `Lab3Part1.circ`.

5. Draw the circuit obtained in step 3 after using K-Map on paper. Then merge an `AND` gate and an `NOT` gate into a `NAND` gate. I.e., distribute the bubbles (little circles) to get the NAND equivalent circuit.

   - **You may ONLY use 2-input NAND, 3-input NAND, and 4-input NAND for this circuit. NO OTHER GATES, including NOT gates, ARE ALLOWED.** No marks will be given if you don't follow these restrictions.

   - **Note**: `NAND` gates are not cascade-able (not extendable) in the same way as `AND` and `OR` from the last lab, so the extension to multiple inputs you used for the `AND` and the `OR` **will not** work for the `NAND`.

6. Draw on Logisim the circuit you designed and simulate it.

   - **How many Data Bits *(input size)* should you use for the inputs?**

   - Recall, you are trying to validate that all possible combinations of values are validated, so definitely will be more than one. In fact, you will be trying to get the truth table in Logisim.

## 2. Part 2: Even Parity Generator

1. Design a circuit which generates an **even parity bit** based on the values of the 4 inputs to the circuit. *Information on parity generation can be found in section 3.8 of textbook* and the last slide in Chapter3 class lecture notes.

   You must show your Boolean algebraic manipulation work in designing the circuit.

2. First, create a truth table showing the even parity generation output from 4 inputs.

3. Then use a K-Map and algebraic manipulation to achieve the optimal circuit for the truth table.

   - Write the map and algebraic expression in the same Excel file you use for the truth table, `Lab3Part2.xlsx`.

4. Simulate to validate.

   - **How many Data Bits *(input size)* should you use for the inputs?**

   - Recall, you are trying to validate that all possible combinations of values are validated, so definitely will be more than one. In fact, you will be trying to get the truth table in Logisim.

- **Note**: You can simplify the circuit using 4-input `NAND` gates as many as possible.

- Consider an `OR` gate with all its inputs inverted. The `OR` gate and the inverters on the input could be replaced by a single gate. Look at the lecture notes for Chapter 3 for ideas.

- **Hint**: The `OR` gate and its associated inverters on the inputs can be replaced by a single gate

# 3. Part 3: BCD to 7 Segment LED Display

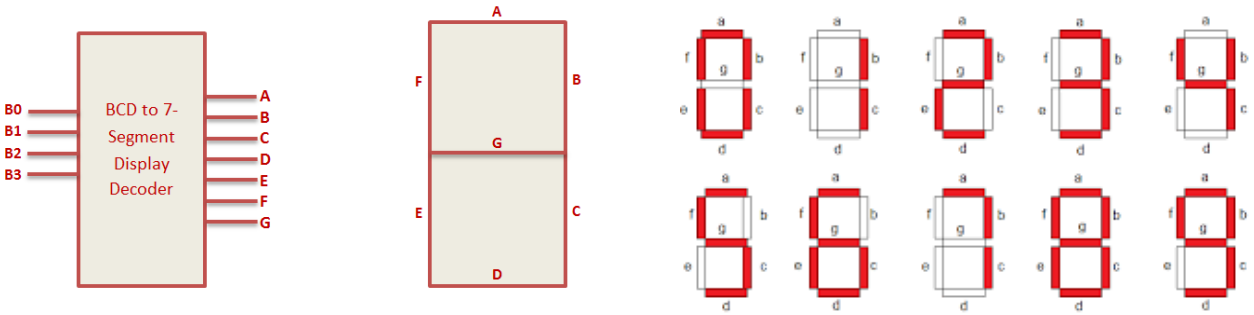
## 3.1 Background information

A **Digital Decoder** is a device which converts one digital format into another and one of the most used digital decoders is called the **Binary Coded Decimal (BCD) to 7-Segment Display Decoder**.

A **7-Segment LED Display** provides a very convenient way of displaying information or digital data in the form of numbers, letters, or even alpha-numerical characters. LED stands for Light Emitting Diode and you can find more information about LED here.

Typically, 7-Segment Displays consist of seven individual colored LED's called the segments, within one single display package as shown in the image on the right. The package has 8 inputs, one for each corresponding segment and one for a small dot. The wiring is also shown in the figure.

A BCD to 7-Segmend Display Decoder is a circuit that takes 4 inputs (the binary coded number) and has 7 outputs (A, B, … G). Each output corresponds with one segment in the 7-Segment Display, in such way that if a particular output is set to 1, then the corresponding segment in the display will be lighted up. The following figure shows on the left the correspondence between segments and BCD decoder output and on the right the segments needed to turn on to produce each decimal digit (0…9)




## 3.2 Lab To-Do

Design a circuit which decodes 4 inputs to 7 outputs which will be connected to a 7-segment LED output display to display a number added by 5 to the input number. For example, the input is 0 ($0000_2$), it displays 5; if 1 ($0001_2$), displays 6; if 5 ($0101_2$), displays 0; and if 9 ($1001_2$), displays 4.

You may assume that the input will be always between 0 – 9 and any value larger than the input range will display an undefined shape. Handling the input out of the range will be a part of Extra Credit assignment of part 4.

The circuit will have 4 inputs (w, x, y, z, or B0, B1, B2, B3) and 7 outputs, one for each segment in 7-Segment LED Display (A, B, C, E, E, F, G).

1. First, create a truth table showing the relationship between the 4 inputs and the 7 outputs described above. Use Excel to write the table and save it as `Lab3Part3.xlsx`.

2. Then use a K-map or algebraic manipulation to achieve the optimal circuit for each output (A through G). Use same Excel file as before (`Lab3Part3.xlsx`).

3. Create a new Logisim project and save it as `Lab3Part3.circ`.

- Add 4 inputs name (label) each one as B0, B1, and so on. You may want to place the inputs horizontally distributed so you can easily read the binary number you are trying.
- Add one 7-Segment Display (you can find it on input/output folder)
- Draw the logic circuit you obtained using K-Map, connecting the 4 inputs with the 7 segments in the display as it corresponds.
- From the Logisim Help/User's Guide, search for "7-Segment Display" to get the detailed information of Pins connection.

## 4. Part 4 – Extra Credit (20 marks)

1. Create a new Logisim project and save it as `Lab3Part4.circ`.
2. Do the same you did for part 3 but make the needed changes to output and "**E**" (segments a, f, g, e, d) in case of the input being a value larger than 9 (i.e., 10 through 15).
3. Save truth tables and K-Map simplification in `Lab3Part4.xlsx`.

## 5. Deliverables

You need to submit **at least 12 files.** And a total of 14 if you do the extra credit.

**Include your name in your PDF files**, as a header or in the first page.

1. **Part 1:**
   - **a.** `Lab3Part1.xlsx` (Truth table and K-Map simplification)
   - **b.** `Lab3Part1.circ` (Circuit)
   - **c.** `Lab3Part1.pdf` (Include: truth table, K-Map and Circuit screenshots)
2. **Part 2:**
   - **a.** `Lab3Part2.circ` (Circuit)
   - **b.** `Lab3Part2.xlsx` (**T**ruth table, K-Map simplification, and Math manipulation)
   - **c.** `Lab3Part2.pdf` (Include : truth table, K-Map, math manipulation, your circuit screenshot and any other details)
3. **Part 3:**
   - a. `Lab3Part3.circ` (Circuit file)
   - b. `Lab3Part3.xlsx.` (Truth table and K-Map)
   - **c.** `Lab3Part3.pdf` (Include: truth table, K-Map and Circuit screenshots)
4. **Part 4: extra credit**
   - a. `Lab3Part4.circ` (Circuit file)
   - b. `Lab3Part4.xlsx.` (Truth table and K-Map)
   - **c.** `Lab3Part4.pdf` (Include: truth table, K-Map and Circuit screenshots)