Introduction

This is a group assignment on using digital *logic* gates and design more complex digital *circuits*.

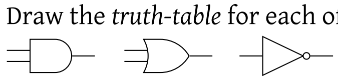
Assignment Goals

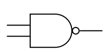
Learning Outcomes After completing this group assignment, each student is expected to be able to

- **Draw** truth tables for a given logic circuit or expression.
- **Draw** a logic circuit for a given truth table.
- **Create** different "simple" gates from other "simple" gates.

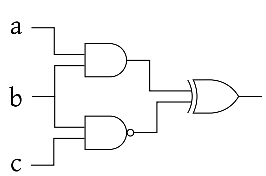
Procedure

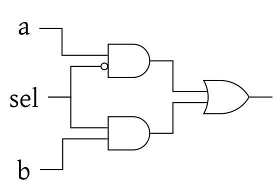
Get out paper for a *single* turn-in at the end of class. Copy enough of each question so that the paper could stand alone as a study guide.
Assign (Least-recently Held) Roles: *Manager*, *Recorder*, *Reflector*, *Speaker*. Everyone should help the whole team contribute and manage time.
Answer these questions:

1. Draw the *truth-table* for each of the circuits below.


2. Using just the gate below, construct circuits that are logically equivalent to **AND**, **OR**, and **NOT** (you may want to do them in a different order).


3. Using only two-input **AND** gates, construct a *three-input* **AND** gate. Before starting, make sure you understand the desired truth-table (no need to draw it).

4. Draw the truth table for the following circuit:


5. Draw the truth table for the following circuit. Put the `sel` column on the left of the table.


6. Add the *binary* numbers `0101` and `0111`; interpreting them (and their sum) as *unsigned* values, what sum did you just calculate, in decimal?

7. Consider a **full-adder** that could add the two numbers from above. How many *carry-in* wires would you need? How many *carry-out*? How many wires of input and how many for output?

8. Draw a *three-input* circuit that determines if exactly two of its inputs are `1`. You may only use two-input **AND**, **OR**, and **NOT** gates.