Blackstone Valley Regional Vocational Technical High School
Student Portfolio- Project Reflection

Date	**9/28/23**
Student Name	**Om Patel**
Subject	**DE (11th)**
Instructor's Name	**Mrs. dos Santos**
Instructor's Signature	**Mrs. dos Santos**

In this project, I am challenged to fix a major problem in an election process. In most paper ballot processes, there is a probability that the count can be under or over-counted which leads to a faulty and false election. To fix this we were advised to create an electronic version that counts the votes of 4 members of The Cabinet, the President, the Vice President, the Secretary, and the Treasurer. To win a vote, the poll must be won by a majority, and in the case of a draw the president's vote will serve as the tiebreaker. For the actual design of the machine, we are only allowed to use AOI logic, also known as AND, OR, and inverter gates. These gates are constrained as I am only allowed to use the two-input versions. The final prototype must also be done by September 29th, 2023. With the final prototype, made on a breadboard, we must also submit a truth table, logic expression of the simplified and un-simplified versions, simulated circuits of both in MultiSIM, the Boolean algebra to simplify the SOP logic equation, and a TinkerCAD simulation of the simplified circuit.

In the first part of the project, I created a truth table. To create this truth table, I used binary number counting from zero to thirty-one to fill out all the possible input combinations. To describe the steps of how to make a truth table in simpler terms, one must add a column for each input and for each column, the interval of 0 to 1 would double. As an example, the very right input is always in a pattern of 0 to 1 to 0 to 1. Then when adding a second column the pattern would be 0 to 0 to 1 to 1. This pattern occurs because we are counting in binary, also known as base 2. Based on the input combinations I could infer the positive outcomes, by identifying which combinations had 3 inputs or votes, and 2 inputs with a vote from the president. With this information, I could then find the minterms of all the positive values. Minterms are used to create a logic equation which is the second step of this project. To create the logic equation, I would simply add all the minterms together. With this logic equation, we could then create a sketch of what a circuit would look like using the un-simplified logic equation. To make this circuit one must understand that addition in an expression is an "OR" gate and an "AND" gate is multiplication. But to get things such as the "not" seen on the equation we would use an inverter. This circuit specifically required 24 "AND" gates, 7 "OR" gates, and 4 inverters. This is an exceptionally large circuit however we can make it much smaller. With my sketch completed I could then model my circuit in MultiSIM, allowing me to check whether or not my circuit works. Using MultiSIM is particularly important as it is a tool that can be used to prototype and make ideas, without any repercussions. It also allows me to create a clean schematic to make a working prototype in real life. To check if my circuit works, I can turn my inputs on in binary and refer to my truth table to see when my circuit should turn on and turn off.

To make it smaller we must use Boolean algebra. Boolean algebra is a way to combine like-binary variables to simplify a logic equation. To combine these binary terms there are a certain set of rules called theorems that one must follow. These theorems are very similar to proofs in geometry having similar rules, such as the Commutative Law, Associative Law, and Distributive Law, but with special theorems that apply only to logic equations such as the Boolean Theorems, the basics of Boolean algebra, Consensus Theorems, and DeMorgan's Theorems.

After simplifying the logic equations, we can once more create a circuit using the same methods as before. When creating this new circuit, the final product was much smaller than what I had before. This simplified version only had 5 “AND” gates, 3 “OR” gates, and no inverters. The simplified version has 77% fewer gates than before. This is a huge benefit as the fewer gates there are the cheaper it is to produce these machines. It also means it takes less time to manufacture the counter. Another benefit is that with fewer gates it means the circuit itself is smaller and takes up less space and is much faster to make an output as the signal must pass through much less. After sketching the circuit, I once more used MultiSIM to create a circuit. This allowed me to check my work, just as I did with the un-simplified one. With this schematic made in MultiSIM, I could then create a circuit similar to a breadboard in a digital breadboard software, called TinkerCAD. In TinkerCAD, I could have infinite possibilities to create breadboard prototypes as there are no repercussions. It is better to use a digital breadboard first as there is no waste of material. If I make a mistake in a digital version, I can easily just delete a wire. Using the MultiSIM schematic I could create an organized, color-coded circuit, that I then tested by using the same method I used to check my MultiSIM circuits. To complete this project, I would have to make a working prototype in real life, using a breadboard. However, this was much easier than expected as all I needed to do was follow the image of my TinkerCAD. Using two 74LS08 QUAD AND gates, 74LS32 QUAD OR GATE, four switches, an LED, and a large sum of wires I completed my working prototype.

Ultimately, this project taught me many things, such as how I would organize a project of this size for myself. It also helped me practice many of the skills I recently have learned. Those of which are Boolean algebra, how to create a truth table and derive a logic equation from one. Also, I practiced the ability to form a circuit from a logic equation and then create a simulation in MultiSIM and TinkerCAD and in the end, build it in real life.

Technical competencies and academic skills demonstrated by completing this assignment.

Framework Standard	Description
2.C.03.02	Construct a circuit.
2.C.02.23	Use schematics and symbolic algebra to represent digital gates as part of a solution to a design problem (logic symbols: AND, OR, NOT, NAND, NOR, XOR and X-NOR gates).
2.C.02.24	Create Boolean expressions and truth tables.
2.C.02.27	Formulate and use a Karnaugh Map and/or Boolean algebra to reduce logic equation.
2.C.03.01	Simulate a circuit.
Embedded Academics	**Description**
2.C.02.23	Determine the meaning of symbols, key terms, and other domain specific words and phrases as they are used in a specific scientific or technical context relevant to grades 6–12 texts and topics.
2.C.06.1.5	Interpret plans, diagrams, and working drawings in the construction of prototypes or models
2.C.02.27.2	Create equations in two or more variables to represent relationships between quantities.
2.C.20.9-10	Follow precisely a complex multistep procedure when carrying out experiments, taking measurements, or performing technical tasks, attending to special cases or exceptions defined in the text