
Inspiring Excellence

# GROUP - 07
## The Numbers Mason

**Name:** Gazi Rehan Rabbi
**ID:** 20101080
**Theory Section:** 09

**Name:** Shakeef Ahmed Rakin
**ID:** 20301046
**Theory Section:** 10

**Name:** Abir Ahammed Bhuiyan
**ID:** 20101197
**Theory Section:** 10

**Name:** Musfiqur Rahman Shourav
**ID:** 19201116
**Theory Section:** 10

# Introduction

In the age of data and technology, our privacy presents itself to be vulnerable to the world. Most of the high end techs and infrastructure have a security system which saves our privacy from exploitation. If we cascade down to the most basic of the security system we might get ideas from where they originated. For instance, a combinational lock has been used as a first line of firewall for years. Old locks and safes use a certain combinatorial pattern that is known to the person who wants to access it and he can lock and unlock his assets according to his will keeping in mind a simple combinational pattern. The analog combinational lock can also be evolved to a digital combinational lock with certain GATEs. Which is quite easy to make. Following the sentences before we present a project where we create a combinational lock system with the help of the most basic gates, Dubbing the name **The Numbers Mason** (Electronic Combination Lock system).

**Objective:**

- To create a lock system using the most basic of Gates.
- Using the concept of Magnitude equator to make the lock.
- Discuss the variations that can be made with the basic idea .

# Proposed Model

We are creating a project with the most basic of the gates. We are taking the inputs in BCD because in a lock we will be taking decimals as input for unit of the machinery. Also, we know that in BCD we can conceive values from 0-9. In our lock system we have a storage unit which will have a BCD code saved. Therefore, when the user will give an input in decimal that decimal number (0-9) will be converted into BCD. Then it will match if the saved code and the input code is same or not. If it is same then a true value is returned otherwise false value is returned. And as there are 4 numbers usually taken in analog combinational lock such as NNNN, we will be assembling total 4 units of the machinery to make the combinational pattern hard. Ergo, only the owner can unlock it.

# Experimental Setup

### Components:

- Proteus Software
- XOR Gate
- AND Gate
- OR Gate
- NOT Gate
- Logic States
- Logic Probes

### Circuit Diagram:

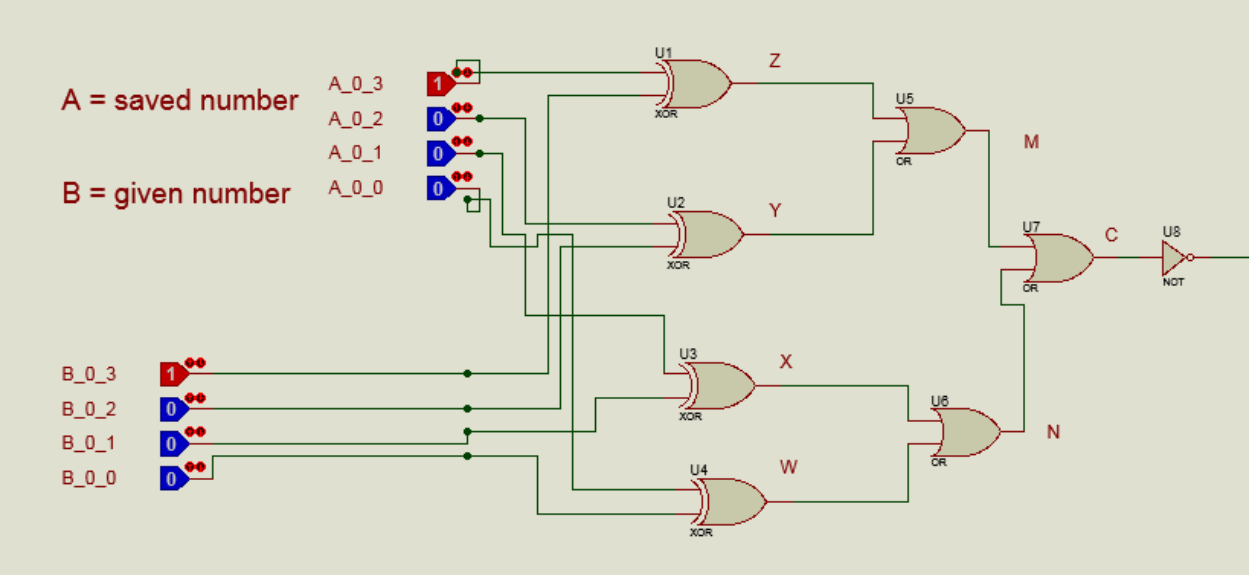


Figure 1: Basic diagram of the lock system

# Results and Analysis

In this case, the stored digits(A) are 0001 and the input(B) given by a user is also 0001. So, practically, this shoudl unlock the combinational lock. The process of figuring out whether the stored and input values match or not is determined like so:

| **Stored Value** | **Value** |
|---|---|
| A_0_0 | 0 |
| A_0_1 | 0 |
| A_0_2 | 0 |
| A_0_3 | 1 |

| **User Input** | **Value** |
|---|---|
| B_0_0 | 0 |
| B_0_1 | 0 |
| B_0_2 | 0 |
| B_0_3 | 1 |

The values for the stored and inputs are fed accordingly into **XOR Gates** where their outputs are as the following:

| **A_0_0** | **B_0_0** | **W** |
|---|---|---|
| 0 | 0 | 0 |

| **A_0_2** | **B_0_2** | **Y** |
|---|---|---|
| 0 | 0 | 0 |

| **A_0_1** | **B_0_1** | **X** |
|---|---|---|
| 0 | 0 | 0 |

| **A_0_3** | **B_0_3** | **Z** |
|---|---|---|
| 1 | 1 | 0 |

The outputs from the XOR gates are then, fed into **OR Gates** like so:

| **W** | **X** | **N** |
|---|---|---|
| 0 | 0 | 0 |

| **Y** | **Z** | **M** |
|---|---|---|
| 0 | 0 | 0 |

Then, the two outpus are input into another **OR Gate**,

| **M** | **N** | **C** |
|---|---|---|
| 0 | 0 | 0 |

Finally, the output from the last **OR Gate** is fed into a **NOT Gate** that determines whether the lock has been unlocked or not where an output of "1" indicates the combinational lock has been unlocked and "0" indicates it has not been unlocked.

| C | NOT Gate |
|---|---|
| 0 | 1 |

Here, the final output from the **NOT Gate** is "1" which indicates that the stored digits and the user input digits are matching. However, if there is any difference between the input and the stored values, one of the **XOR Gates** will have an output of "1" that at the end will change the output of the final **OR Gate** to "1". After this value is fed into the final **NOT Gate**, the final output will become "0" which means the lock has not been unlocked.

The system can be expanded to accomodate even more digits by feeding each output from the **NOT Gates** into several **AND Gates** where the output will remain similar. This is recreated in our Proteus Project file where we combined four individual systems into one single **Combinational Lock**.

# Conclusion

In this project we have tried to illustrate a 4 digit combinational lock system and if someone wants to input more than 4 digits, it is not possible. Therefore, it can be told that this project is not feasible and also should be used only as a prototype.