3C8 Project 2 Report – Combination Lock

Sean Fahey 21360313

Group 3

Introduction

The goal of this project is to create a digital combination lock using basic electronic components on a breadboard. The user will input a code into a keypad and the circuit will use different LEDs to indicate that the correct (or incorrect code) has been entered.
The circuit's logic is made up entirely of CMOS 400 series components, as was indicated in the assignment description. The only other materials used were a breadboard, insulated wire, coloured LEDs, a keypad and the power source from the workbench in the laboratory.

Requirements

Before any of the combination lock's digital logic is implemented, the circuit must be able to reset itself and take keyboard input so that the circuit can re-enter its initial state between uses.

Keyboard debouncing

There must be some way of determining (i) if a key has been pressed, and (ii) which key has been pressed. We can split the 12 keys on the keypad into 3 different groups;

1. Code keys {1, 2, 4, 8}
2. Non-code keys {0, 3, 5, 6, 7, 9}
3. Enter and Clear {#, *}

There are edge cases (unclear circumstances) where they keyboard must decide how to act:

1. 2 keys pressed simultaneously
2. Keys pressed while the circuit is in one of the output states

These will be explained under the *keyboard output* heading in the *Design* section.

Reset D-Flip-Flops

The D-Flip-Flops (DFFs) reset to digital low, which occurs when the user presses the *clear (*)* key or when an output state is engaged.

Reset multivibrator input

Input for multivibrator should be the output of a mealy state machine, where it does not remain in a given state indefinitely. Therefore the keys should directly affect the state of the multivibrators' input wire.

Reset counters

There are 2 counters, 1 for counting the number of keystrokes (*Input Counter*), and 1 for counting the number of incorrect attempts by the user (*Wrong Code Counter*).

The *Wrong Code Counter* should auto-reset upon reaching 3, and can only be manually reset by entering the correct code (8214#).

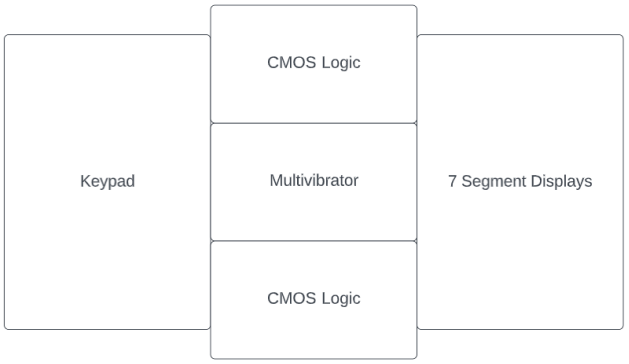
The *Input Counter* resets when an output state (Open/Alarm/Lockout) is engaged.

Design

Breadboard layout

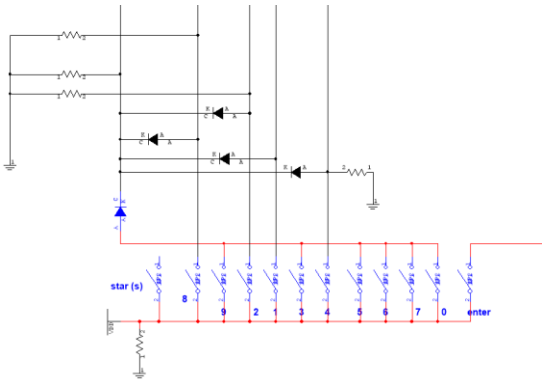
The 4000 series chips used were selected such that only one breadboard was needed, and positioned on the breadboard to minimize the total length of wire used.

The diagram on the right is a rough depiction of how the breadboard is laid out.



Keyboard output

The *Code Keys* and *Non-Code Keys* must all be wired together such that pressing any of them registers that a key has been pressed. The best way to do this with logic gates is to use a 10-input OR gate, however this type of gate is not available, and the alternatives (using several 2-input or 3-input OR gates) are impractical when we accounted for space on the breadboard. Our solution was to use diodes, as they are very easy to use and space-efficient.



The keypads in the lab have 12 output pins, numbers 0-9, * and #. The 13th pin is the only input pin, ground. This means current would flow from the output pins through the key being pressed then to ground. We wanted to have the keypad pins output digital high when pressed, and digital low otherwise. To implement this we attached the ground pin to a junction with Vdd and a resistor, which was connected to ground. That way, when a key is pressed, a path from Vdd to the desired output pin is short-circuited. Each of the 5 outputs (*Any*, *8*, *2*, *1*, *4*) are connected to a pull-down resistor in order to set them to digital low when the key is not pressed.

Edge cases:

If 2 keys are simultaneously pressed one key will always get priority, and the other will not register at all according to these rules:

- If a *Code Key* is pressed with a *Non-code Key*, the *Code Key* will get priority.
- If * or # are pressed with any other key, they will take priority.
- If * and # are both pressed, # will get priority.

Caveat:

> If 4 keys have been pressed, and then 2 keys are simultaneously pressed as the 5th entry, there is no way to prioritize a key. So whichever key is pressed first will be the 5th key, and the other key will be registered during the next attempt.

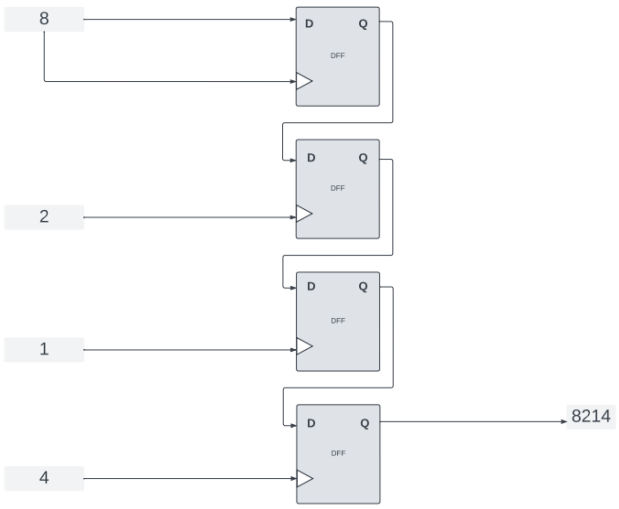
Detect correct/wrong code

An attempt is registered as *wrong* if either:

- 5 keys (0-9) have been pressed
- 4 keys not matching the code (in order) then #

On the right is a simplified version of the DFFs. As we can see the only way to set the output labeled 8214 to digital high is by entering 8 then 2 then 1 then 4. (Note that in reality the first clock input from 8 is buffered such that the clock receives the signal after the D input receives it).

An attempt is registered correct when output 8214 is set to digital high, and then enter (#) is pressed.



Multivibrators

3 multivibrators are used. 1 for engaging the alarm state (5 sec.), 1 for the open state (5 sec.) and one for the lockout state (60 sec.). Each of these are indicated by an LED on the breadboard. The LEDs are always on, until the associated state is engaged, during which the LED(s) will be turned off.

Simulation

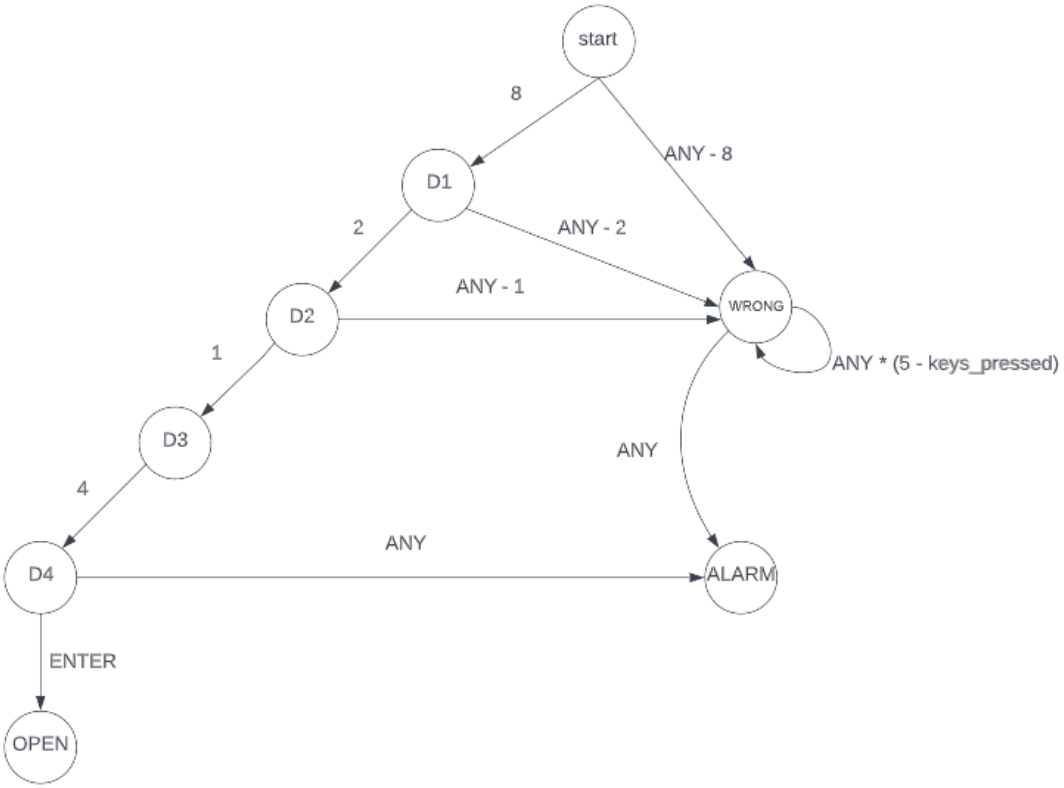
The simulation was extremely useful for building the circuit. We went through several iterations to arrive at the final design, which was made easy with the combination of Multisim and GitLab.

Build

We tried to colour code the sections as best we could with the limited colours of wires available in the lab. We built the circuit on the breadboard piece by piece. The keypad and multivibrators were built first as we could test them easily, since they operate independently of any CMOS logic or state of the combination lock.

We then built the required circuitry for the correct operation of a particular circumstance (e.g 5 keys (0-9) pressed triggers *alarm multivibrator*), as we could test these functionalities one by one as we went along.

Flow Chart



Start: No keys have been pressed yet

DX: X correct keys have been pressed so far

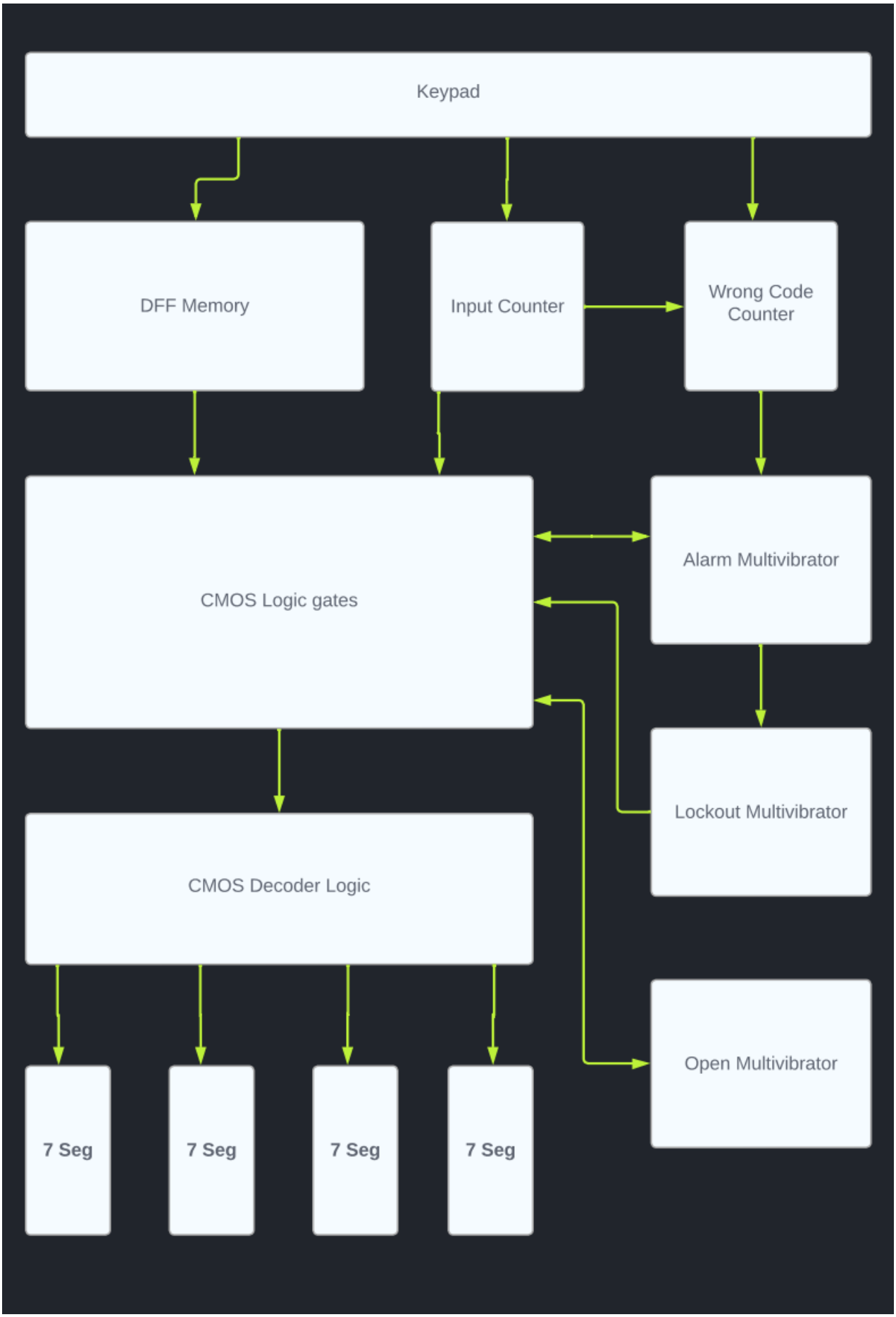
ANY – {N}: any numerical key can be pressed except for *N*

For any state, if * (CLEAR) is pressed, the present state immediately becomes start

For any state except for D4, if # (ENTER) is pressed, the present state becomes ALARM

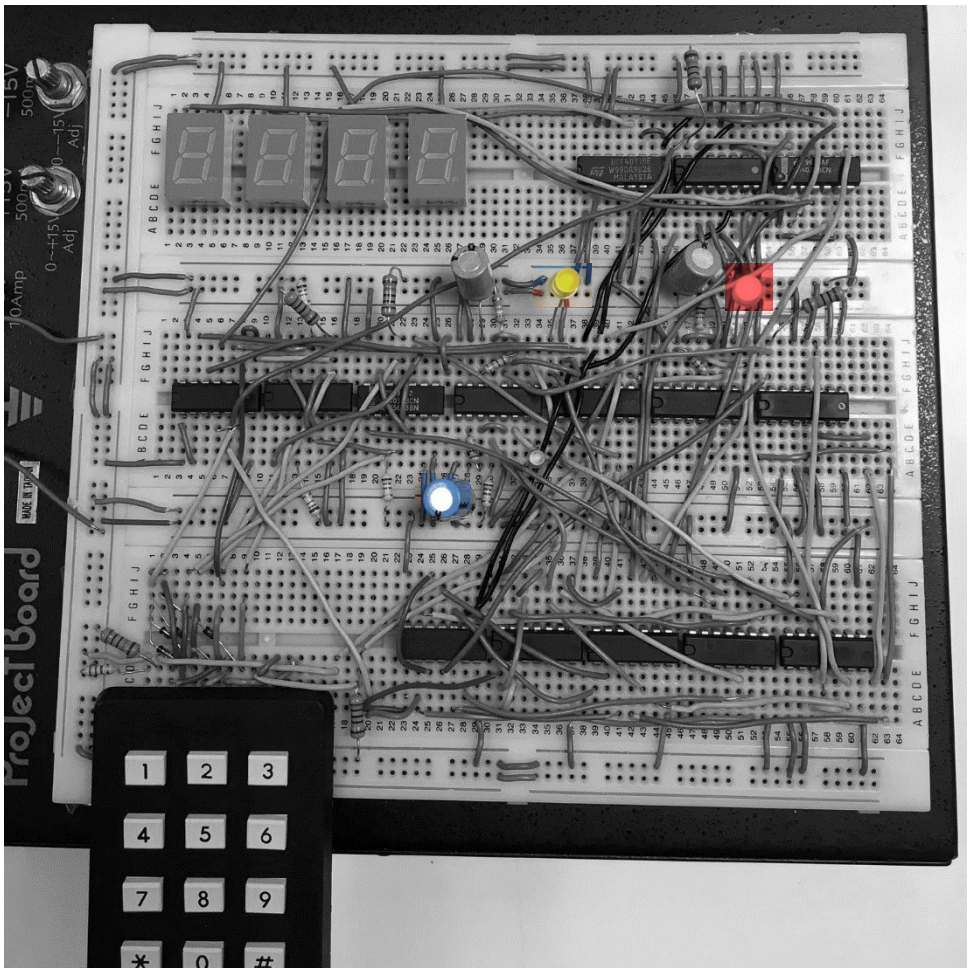
The circuit enters the *WRONG* state, until it either (i) returns to *start* by pressing clear (*), or it goes to *ALARM* when the number of keys pressed is 5 or the enter (#) key.

Block Diagram

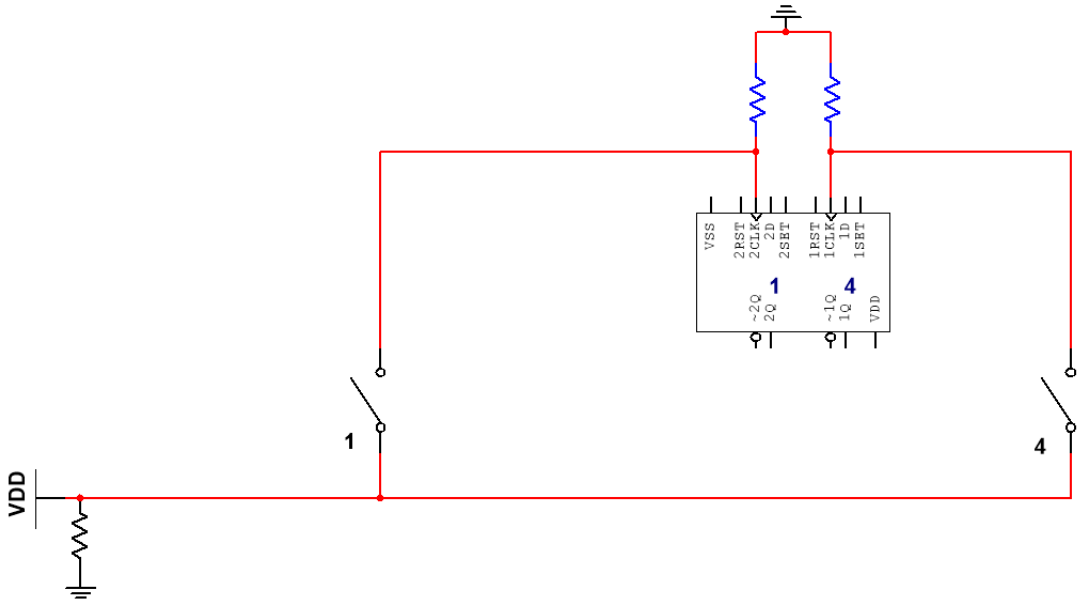


Circuit

- The LED highlighted in red is the lockout indicator. While the circuit is in lockout mode, this LED turns off for 60 seconds.
- The yellow LED is the alarm indicator. This LED turns off for 5 seconds while the circuit is in the alarm state.
- The blue LED is the open state indicator, and it stays off for 5 seconds while the circuit is in the open state.



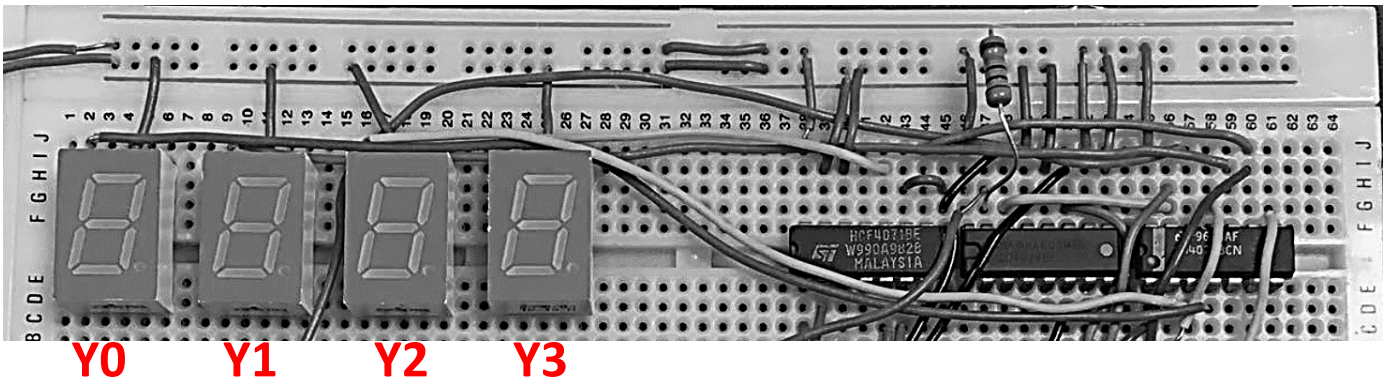
4538 Multivibrator	Open	Alarm	Lockout
RC Value	R = 120k Ω C = 42µ F RC = 5.04 seconds	R = 120k Ω C = 42µ F RC = 5.04 seconds	R = 1.5M Ω C = 42µ F RC = 63 seconds



Above is a representation of how floating connections are dealt with. The component shown is a 4000 series D-Flip-Flop, with only the clock inputs considered. When neither 4 or 1 are pressed on the keypad, the clock inputs are left floating in an undefined state, which causes unwanted behavior. The solution is to add pull down resistors, so that the inputs are always in either a digital high or low state. The same is done for 4029 counter clock inputs and the other 4013 D-Flip-Flop clocks.

CMOS Decoder Logic

This group of logic gates are used for taking the 4029 input counter and creating the output for the 7-segment displays correctly.



Q0, Q1, Q2, Q3 are the 4029 counter outputs, and Y0, Y1, Y2, Y3 are the inputs for the 7segment displays.

Truth table

Q0	Q1	Q2	Q3	Y0	Y1	Y2	Y3
0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0
0	1	0	0	1	1	0	0
1	1	0	0	1	1	1	0
0	0	1	0	1	1	1	1

Karnaugh Maps

Y0

Q_0Q_1 \ Q_2	0	1
00	0	1
10	1	X
01	1	X
11	1	X

Y1

Q_0Q_1 \ Q_2	0	1
00	0	1
10	0	X
01	1	X
11	1	X

Y2

Q_0Q_1 \ Q_2	0	1
00	0	1
10	0	X
01	0	X
11	1	X

Y3

Q_0Q_1 \ Q_2	0	1
00	0	1
10	0	X
01	0	X
11	0	X

Y0 = Q0 + Q1 + Q2

Y1 = Q1 + Q2

Y2 = Q0.Q1 + Q2

Y3 = Q2