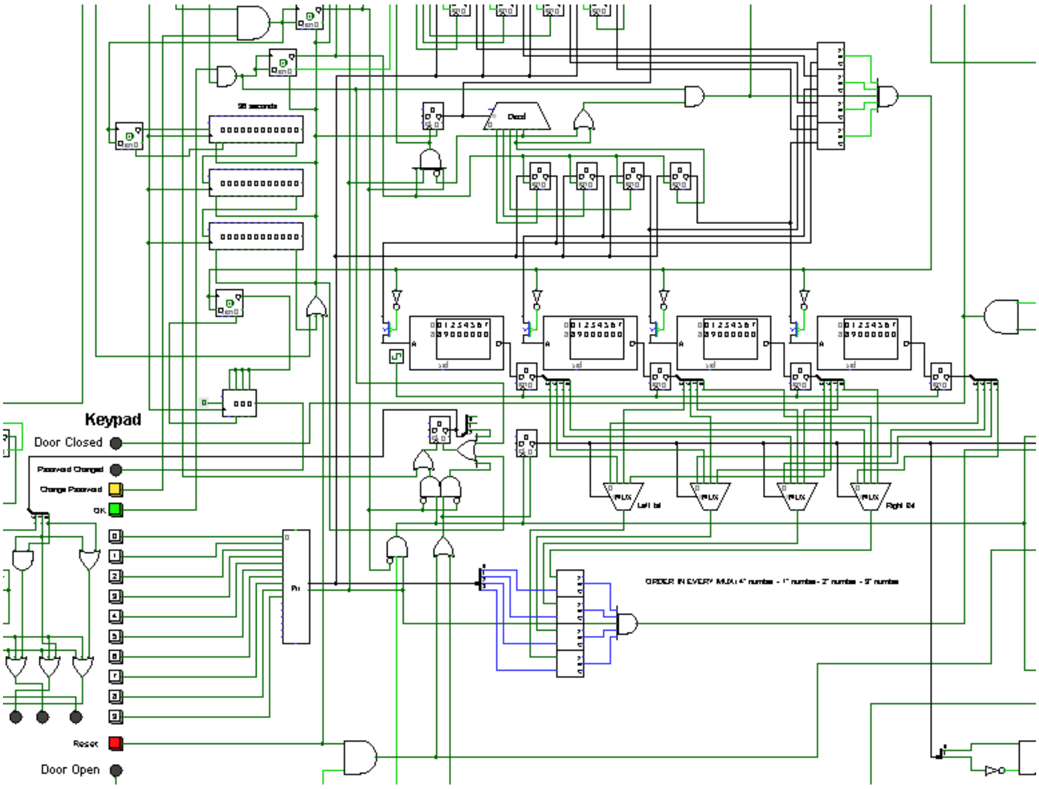
22 NOVEMBRE 2019



# ELECTRONIC SAFETY LOCK

## INNOVATION OF THE DAILY EXPERIENCE

EDOARDO PANICHI 1856156 - GIULIO ROSADI 1856166
TONGJI UNIVERSITY & ALMA MATER STUDIORUM BOLOGNA
edoardo.panichi@studio.unibo.it - giulio.rosadi@studio.unibo.it

## Abstract

The design of a 4-digit electronic locking system for a safer and easier daily life is reported in this work. The project was developed exploiting all the knowledge learned during the Logical Networks course, from simple logic gates to complex modules, such as decoders and encoders, comparators, multiplexers, registers, counters, transistors, etc. We also had to deal with issues concerning the management of synchronous and asynchronous signals.

Unlike a conventional key, the electronic lock accepts a code composed of four digits, compares it to the stored code in its memory via comparators and allows the user to enter a door if the password is right.

The design also provides the opportunity to personalize the secret code in a safe and reliable way, without giving up the quickness and intuitiveness of a user-oriented device. In addition to this, the design of the lock permits to prevent eventual human mistakes, bringing the user experience in a more dynamic and technologic dimension.

The circuit, realized contemplating all the possible scenarios, was built and tested so that undesirable situations could emerge without affecting the performance of the lock. For example, the erroneous usage of the buttons on the keypad does not cause any trouble to the circuit and keeps the final output unchanged.

The detailed description of the project, provided with all its functionalities, can be found in the following pages.

## Keywords

*Safety* *User-Friendly* *Counters* *Shift Registers* *ROMs*

## Introduction

A lock is a device used to protect information, places or valuables. Even if we do not think about it, locks are omnipresent in our daily life, in different forms but with a common aim. Indeed a lock can be mechanical, magnetic or electronic; while the first type is widespread and it has already been used for at least three thousand years, the magnetic and the electronic ones are relatively new as they were invented during the 20th century.
The mechanical locks are for sure cheaper and easier to realize than the other two, but in many situations an electronic safety lock or a magnetic one can be the best choice as, for instance, the key can be replaced by a simple code or a card.

When we had to decide the project we wanted to design, we searched for something which could have been versatile, daily used and challenging to implement; therefore an electronic safety lock appeared to us to be the perfect choice.
In projecting our security device, we have tried to realize a circuit that suits perfectly for an electronic safety lock, but such a design is easily suitable for many different situations: a safe, an alarm and everything else that needs to be protected by a password.

Once finished the design of our circuit, we compared it to other similar projects[1] found on the internet to see the differences and analyze better the strong points of our work. Studying Project A[2] it is possible to notice that many features are absent if compared with ours. First of all the input combination is a 2-digit code only, instead of a 4-digit code, then the lack of any form of timer, useful to ensure more security to the lock and an easier experience for the user, as well as the possibility to set and change password according to the user wishes are two remarkable lacunas. On the other hand, Project B[3] seems to be more similar to ours even if some important features are not present also here. As in Project A, Project B seems not to have timers, but more important is the lack of ROMs to store the password as the usage of FFs or register does not give the possibility to remember the secret code if the power supply is switched off.

## Description of the Methodology

During the development of the project, we took into account two different sections. The first part, the one we designed at the beginning, intends to analyze the user's activities and to lock or unlock the door consequently, so basically to achieve safety. Speaking about the second part, which was realized at a later stage, we focused on the opportunity to change the password of the lock for a more trustworthy and personalized experience.

To realize the first part, we started from the user's point of view and designed the keypad (Fig.1) to interact with. It is made by ten buttons representing the digits from 0 to 9, a *reset* button, two LEDs to indicate if the door is open or closed, and four lights to show when a number has been selected. We also implemented a blue LED and two buttons (*change password* and *ok*) that are related to the second part of the circuit.
What we need in order to understand the input from the user is some specific device able to interpret the press of a button as a binary number: the most natural choice for us was an encoder. All the wires from the buttons are connected to a priority encoder, which results in two outputs.

[1] The projects are two: *The design and construction of a 2-digit electronic lock system using simple circuit elements and digital electronic logic gates* by F. G. Akinborz, H. K. Yusuf, M. A. Adedeji and *Design of A Keyless Digital Security System* by E.O. Oyetunji and J. Asare.

[2] To simplify the discussion we will refer to *The design and construction of a 2-digit electronic lock system using simple circuit elements and digital electronic logic gates* by F. G. Akinborz, H. K. Yusuf, M. A. Adedeji as Project A.

[3] To simplify the discussion we will refer to *Design of A Keyless Digital Security System* by E.O. Oyetunji and J. Asare as Project B.

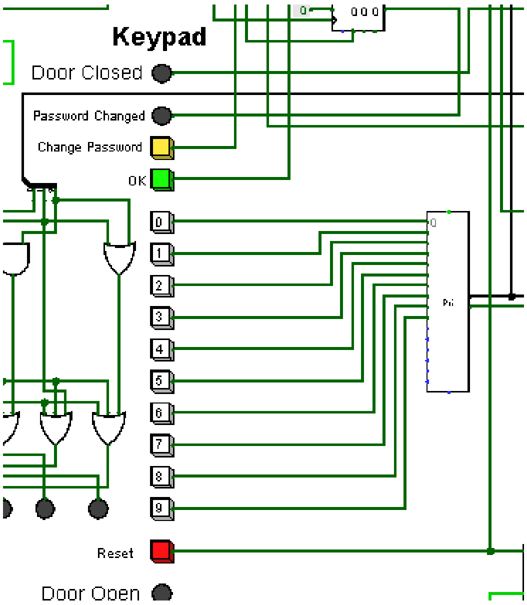


*Fig. 1*

The first output is a 4-bit sequence that identifies a number from 0 to 9 and is sent to a group of four comparators (Fig.2). This structure aims to compare bit-to-bit the entered number with the right password. The results of the comparison are connected with an AND gate because what we need is to have all the four bits simultaneously correct. When the user has not pressed a button yet, we have that the output of the encoder is not defined: it means that we cannot be sure about the result of the comparison. To avoid this problem it is sufficient to add a new input to the AND gate.

This input is the IA, the second output of the encoder, and it is 1 exclusively when the user presses a button, so that at any other time the gate will undoubtedly result in a 0. In fact, the comparison is only relevant when a button is selected.

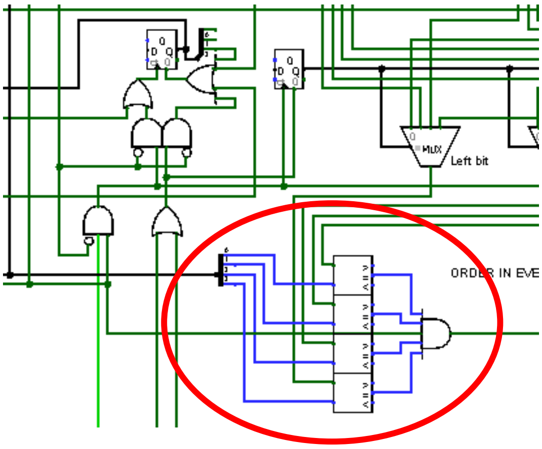


*Fig. 2*

What we have now is a structure that can detect the insertion of a number and compare it with another, already stored in the circuit, but the password is a 4-digit combination. We have to keep track of how many digits the user has chosen so that, at any time, we can know what the right one should be. To achieve that, it is enough to implement a 2-bit counter (Fig.3), which is able to distinguish four different states, each one corresponding to a digit of the password. The clock port is connected to the IA, because the counter needs to rise every time a new button is pressed, but it is also connected to a wire that indicates whether the door is unlocked or not, because if the lock is already open we do not need to register the activity of the user. In addition to this, the clock port is also connected to a third wire, which determines if the user has already pressed the button to change the password, because the counter's purpose is to work exclusively when the user is trying to input the right combination and not when he's typing a new one. The counter may also reset in some situations, for example when we press *reset* on the keypad or when too much time has passed.

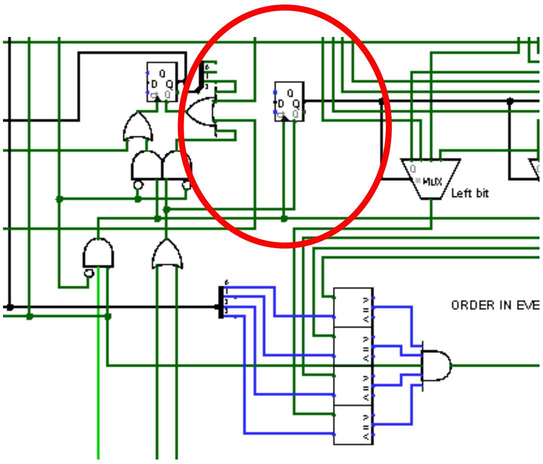


*Fig. 3*

At this point, the output of the counter reaches four multiplexers (Fig.4). The inputs of the multiplexers are the bits representing the numbers of the correct password. The bits selected by every mux are connected to the comparators, where they can be used to check if the input of the user is right. For example, if the password is 4099 and we have to enter the number 9, we will have the bits "1001" as output of the multiplexers. These bits will be compared with the number inserted by the user, which in this case may be 2, "0010" in the decimal system. Since they are different, the signal coming from this part of the circuit will be 0.

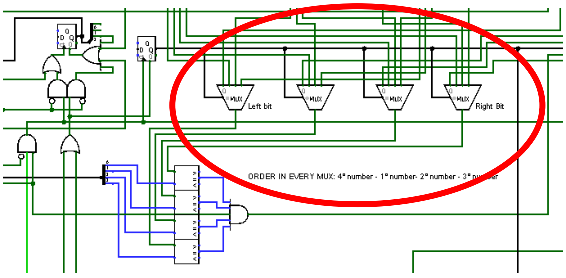

MUX
MUX
MUX
MUX
Left bit
Right Bit
ORDER IN EVERY MUX: 4° number - 1° number- 2° number - 3° number

*Fig. 4*

This last signal, whether it is 0 or 1, is sent into a 4-bit shift register (Fig. 5). In this way, after four buttons pressed there are going to be four relevant bits in the shift register. Only if all these four bits are 1 the combination entered by the user is correct. To detect that, all the bits exit from the register and merge into an AND gate.

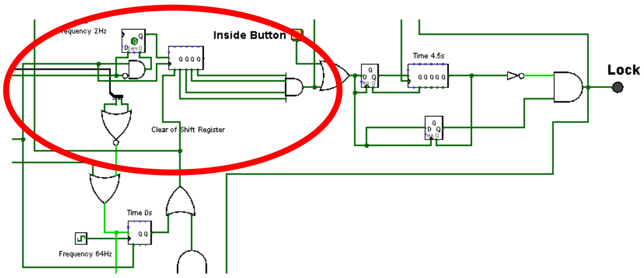


*Fig. 5*

The shift register resets through the clear port in three conditions. Firstly, if too much time passes: there is a structure, mainly composed of a shift register and a delayed clock, which counts up to 11 seconds from the moment that the user starts to insert the password.

Secondly, if the user presses *reset*.

Thirdly, if the user has finished entering the four numbers: there is a structure, made up with a clock and a 2-bit shift register, which detects from the counter if the last number has been inserted and sends a signal to reset after a negligible time.

The last elements of this part of the circuit have the purpose of leaving the door unlocked for a reasonable amount of time (Fig.6). The bit from the AND gate we mentioned before, used to indicate if the password is correct, is sent to the lock, which potentially opens.

If the door is unlocked, i.e. the bit is a 1, we have that the 1 is also sent to another shift register that "carries" it for about 4.5 seconds. After that time, the 1 exits from the register, gets inverted, reaches the lock and closes it.

The same result can be obtained by simply pressing the inside button, which is not located on the keypad but on the other side of the door.

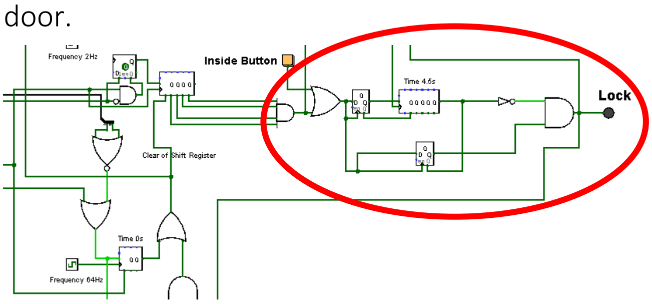


*Fig. 6*

The second significant part of the project is the circuit necessary to change the secret code used to open the door (Fig.7). This feature is fundamental to ensure safety and practicality to the lock.

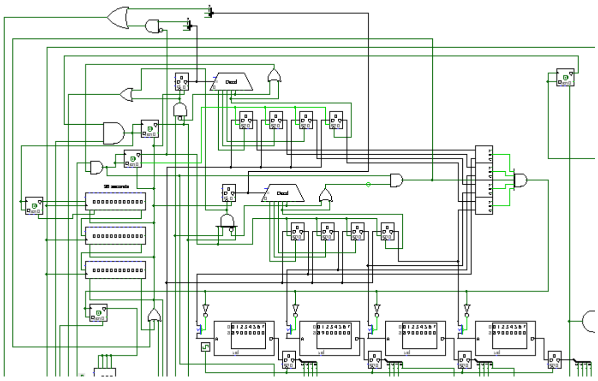


*Fig. 7*

To achieve this result we added two buttons, *ok* and *change password,* to the keypad which are used by the user to express his will to change the code.

*Change password* can be pressed at any time, but it will have an effective result only if the right code has been already inserted. This process is achieved easily using an AND gate

(Fig.8, red circle) that computes the signal coming from the button and a wire coming from the result of the first part of the circuit where the password is checked to be correct. This is important to be sure that, only who has the authority to open the door, has also the possibility to change the entering code.

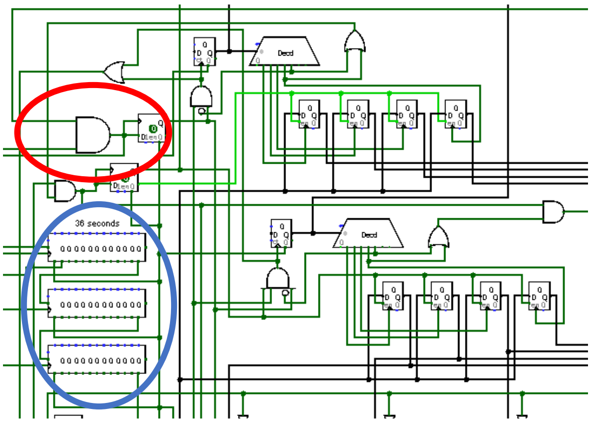


Fig. 8

When *change password* is correctly pressed a timer (Fig.8, blue circle) of 36 seconds starts: this is the time that the user has to insert the new combination, press *ok*, insert the new combination again and press *ok* one more time. The timer is necessary to ensure that, if for any reason the user cannot complete the procedure and he forgets to push *reset,* in less than a minute the lock will be usable as always. The timer is simply composed of three shift registers in series, that act just like a longer shift register. When the timer starts the signal from *change password* is also stored in a D-FF[4] (Fig.9, red circle) that activates a new part of the circuit composed mainly by four D-FFs, a decoder and a 3-bit counter (Fig.9, blue circle).

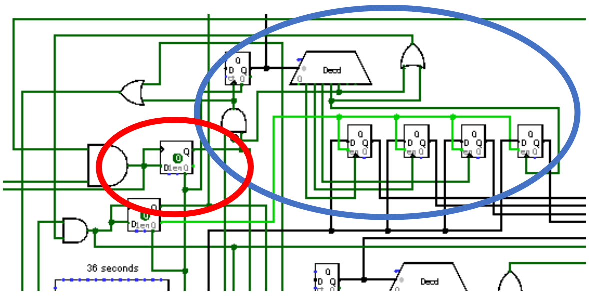


Fig. 9

If CP D-FF[4] is 1, then every time we press a numbered button on the keypad the 3-bit counter will increase by one, and using the output of the 3-bit counter as input of the decoder it is possible to select the first, the second, the third or the fourth D-FF to store the input number. Even if after that four numbers have been saved the user keeps pressing other numbers, these are not memorized as the 3-bit counter is disabled if more than five[5] numbers have already been inserted.

At this point we have saved the new combination in D-FFs. But to be sure that the user is conscious of the new code, that he is going to set it as new password, it is necessary that he inputs the same code again; but first it is requested to the user to press *ok* in order to confirm the end of the first part of the process of changing the password.

As when *ok* is pressed an important change to circuit happens, it is important that this signal is accepted only if all the four new digits have been input. To achieve this goal we use again an AND gate (Fig. 10) which connects the *ok* wire and a wire that detects if the output of decoder corresponds to the fourth or fifth status of the 3-bit counter.

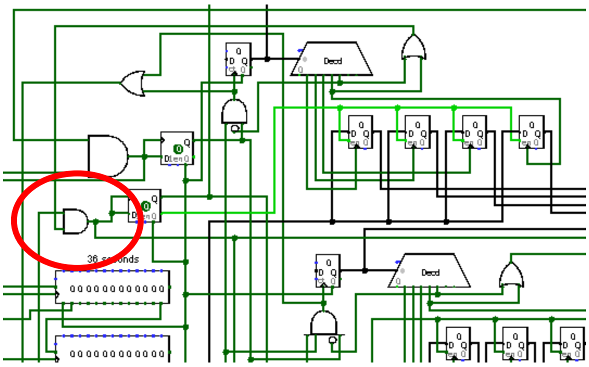


Fig. 10

If *ok* is correctly pressed, then a 1 is stored to the OK D-FF[6] (Fig. 10). This information is used to switch off the previous part of the circuit, in order to switch on another section of the

[4] I will refer to this D-FF calling it the CP D-FF (Change Password D-FF) (Fig.9, red circle).

[5] Note that even if the fifth number affects the output of the 3-bit counter, this does not cause any trouble, because the fifth output of the decoder is not connected to the D-FFs.

[6] OK D-FF is the flip flop which stores the information that *ok* has been pressed correctly, exactly the same function of CP D-FF for *change password.*

circuit, constituted by the same components with the aim of analyzing and storing the new combination inserted for the second time by the user.

Then the user presses *ok* again to indicate that even this part of the process is completed.

The output of the eight D-FFs, used to store the numbers that the user has input, are constantly compared by four 4-bit comparators (Fig.11) in order to understand if the content of the FFs is equal or not.

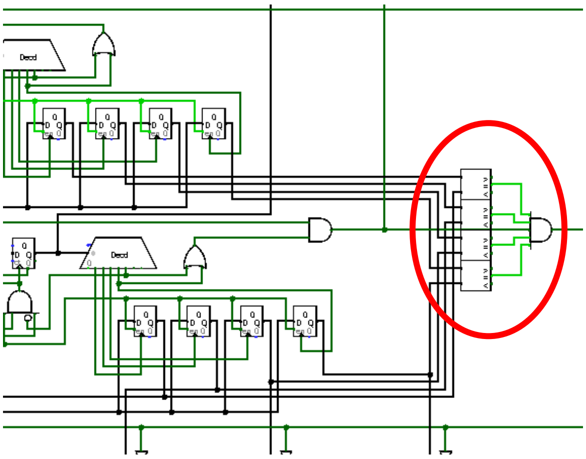


*Fig. 11*

Obviously, the content of the FFs must be equal, but it is likewise important that all eight numbers have been inserted. To ensure this, the outputs of the comparators are computed in an AND gate with a fifth wire, which is high only if we press *ok* and the second 3-bit counter is set on the fourth or the fifth state.

So if *ok* is pressed twice, and the user has input the new combination correctly both times, a high signal can go through the AND gate. This wire is then connected to four transistors studied to switch on a part of the circuit only if the output of the AND gate is high. The part of the circuit that is turned on is composed of four ROMs (Fig.12), whose select ports are connected with the output of the last four D-FFs.

This means that, if the user has inserted the new combination rightly twice and he has pressed *ok* for the second time, the transistors will connect for an instant the output of the D-FFs to the select ports of the ROMs. These connections will select in the ROMs the new values chosen by the user and the procedure of changing password will end.

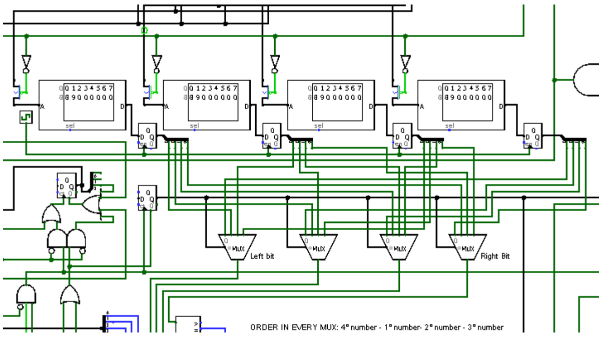

MUX
MUX
MUX
MUX
Left bit
Right Bit
ORDER IN EVERY MUX: 4° number - 1° number- 2° number - 3° number

*Fig. 12*

If this process is correctly completed, the result is signaled to the user thanks to a blue LED, which is turned on for a few seconds.

ROMs[7] are used to store the password because they are not volatile memories, so if the power supply is switched off, the password will be stored anyway.

Obviously, the content of the ROMs is used as input of MUXs used in the first part of the circuit.

It is also important to notice that the entire process can be interrupted instantaneously by pressing *reset* on the keypad: this button indeed will clear the timer, the 3-bit counters, the CP D-FF and the OK D-FF.

Other relevant parts of the circuit are the four red LEDs (Fig.13, blue circle) present on the keypad, used to show how many numbers have already been input.

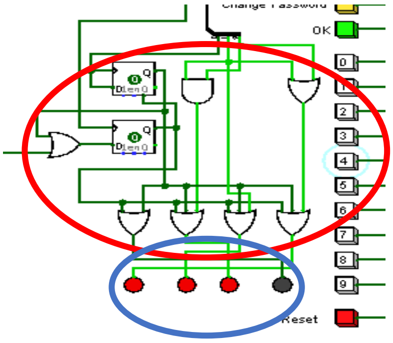


*Fig. 13*

[7] Actually, the outputs of ROMs are connected to other four D-FFs, this thing is necessary only to show better the output of ROMs. Indeed, if you close the project on Logisim and you open it again the content of the ROMs is erased, and the output is set on zero even if it is not shown graphically.

When the user inserts the first number, the first LED turns on, when the second number is inserted, also the second lits up, and so on. When the fourth number is inserted, two possible situations can happen: if the user is typing the code to open the door, as soon as he inserts the last digit, all the four LEDs become red and after something like a second they turn off automatically. This because the user can immediately try another code, if the first one was wrong, or simply open the door and move on if the password was right.

Otherwise, if the user is trying to change the secret code, when the fourth digit is pressed the four LEDs will stay on until *ok* is pressed. This to indicate to the user that all the numbers have already been input and he must press either *ok* or *reset* to keep using the keypad.

To achieve this behaviour of the LEDs we added a 3-bit counter[8] (Fig.14), whose clock port and clear port are preceded by an OR and AND gate. With these two gates the counter recognizes if the user is using the keypad to open the door or to change password. According to this information, the LED counter will follow the behaviour of the 2-bit counter[9] or it will follow one of the two 3-bit counters that are situated close to the eight D-FFs storing the new combination.

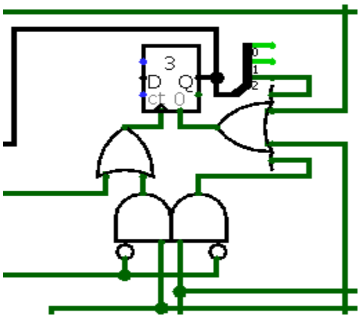


*Fig. 14*

Then the output of the LEDs counter is used to distinguish how many buttons have been pressed, in order to turn on the right number of LEDs. The part of the circuit that makes this possible is shown in Fig.13 (the red circle). This is also the final section of the project.

## Results and Discussion

The project has been simulated multiple times, resulting in some interesting outcomes.

First of all, the simulation confirmed that the press of the inside button instantaneously unlocks the door but does not allow any other operation. The part of the circuit that can access the ROMs (where the correct combination is stored) is isolated, and the only way to switch it on is after typing the right password.

The user's fundamental approach to the lock is actually through the numeric keypad because it is necessary to input the correct sequence. In the first place, the different simulations show the proper working of the lights: the red one is on if the lock is blocked, while the green one switches on when the door is unlocked. In addition to this, every time a number is inserted, a new light goes on: it means that the user can correctly keep track of how many numbers he has previously entered. If we press *reset*, the password insertion is immediately terminated (and all the lights are switched off) and the user is able to re-input the sequence.

Let us suppose that the numeric password is wrong. It happens that the comparison with the correct combination is not successful and the circuit does not allow the opening of the lock, therefore the user can try to insert the combination again. On the other side, the simulation of a correct-input situation leads to interesting consequences: the selected numbers are successfully compared with the right combination, a positive signal is sent to the lock, which opens for around 4.5 seconds; after that, everything goes back to the original conditions.

We have also tested the circuit behaviour if the user starts entering the password but does not finish in a reasonable amount of time. After 11 seconds, the circuit resets itself, and the user is requested to digit the combination again.

While the lock is open, a 1 is correctly stored inside a flip-flop until the door closes. As a consequence, if we press *change password*

[8] I will refer to this counter calling it LEDs counter.

[9] It is the counter of Fig.3

during this time interval, the 1 is transferred to another flip-flop, which does not reset in a short time. The user has about 36 seconds to change the password before a signal is generated to reset the whole circuit. At this stage, we can press the numeric buttons on the keypad in order to form a new 4-digit password, which is to be temporarily stored in four flip-flops. When we try to select five or more numbers, the simulation confirms that only the first four digits are relevant, while the other inputs are ignored. The user is now allowed to press the *ok* button (if we press it before entering a complete sequence there will be no effect), then he can confirm the combination by merely entering it again.

After *ok* has been pressed for the second time, the two passwords entered by the user are compared, and the simulation may show two opposite results. If the two 4-digit sequences are the same, the transistors give access to the ROMs and the right combination gets replaced. In addition to this, a blue light switches on to indicate the successfulness of the operation. Otherwise, if the two sequences do not match, no transistor is activated and the circuit resets itself, keeping the old password.

## Conclusion

The electronic safety lock we designed is studied to be user-friendly in each its aspect, starting from the LED indicators and the impossibility to damage the circuit in any way, concluding with the chance to reset any process with the same button and the easiness of inserting the code or changing it.

The circuit is thought to be sheltered by any wrong input of data and to be safe on any occasion.

Its versatility gives the opportunity of rearranging it in order to use it in many different situations. Indeed the circuit, if adjusted, can be employed wherever a password is needed to protect any type of valuable object.

In conclusion, our lock does not want to change the method of opening a door, neither wants to introduce a super innovative feature, but it is realized in order to ensure safety and, at the same time, the easiest experience possible for the user, because unlocking a door does not have to be something revolutionary but must be easy and quick.

## References

Digital Design - Fourth Edition by M. Morris Mano and Michael D. Ciletti - Pearson Education, 2007

Logisim, educational tool by Carl Burch

https://en.wikipedia.org/wiki/Lock_and_key
https://en.wikipedia.org/wiki/Electronic_lock
https://en.wikipedia.org/wiki/Electromagnetic_lock

https://www.cannonsafe.com/digital-vs-mechanical-locks