**Xiaodong Song**

**4/25/2020**

**Identify**

For lab 11, the experiment requires students to come up with an ide for a system design to be implemented on the Nexys4 DDR board that will perform some application. However, COVID-19 causes that the verification and project test would only be able to do by Vivado simulation. For this individual project design, a Cipher/lock machine design would be developed. This machine can be used on so many places such as door’s lock and locker’s lock. It can improve and protect user’s property security.

**Formulate**

For this Project Design, a 4-digit Cipher/lock project would be modelled. This Cipher/lock would allow users to set 4-digit passwords they want. Then the program would check the input passwords whether they are correct. This design would be as following:

1. The default password is 1234(hexadecimal number) at beginning;
2. Every time users input any numbers, there would be an out signal which can display input numbers;
3. The numbers which can be input as passwords are 0,1,2,3,4,5,6,7,8,9(hexadecimal number);
4. There are two input hexadecimal number representing “finishing inputting”, “start inputting passwords”, and “start setting new passwords” such as ‘D’,’E’,’F’.
5. There is an output representing the ‘AN’ port displaying.
6. There will be some output signals: Close, Error and Open which are representing “still not get correct passwords”, “input 4-digit passwords is wrong” and “input 4-digit passwords is correct”, respectively. (i.e. If password correct, Open = ‘1’. If passwords correct, Error = ‘1’. Else Close = ‘1’);
7. Display state’s output hexadecimal number ‘f’ represents no showing/no numbers.

**Solve the Problem**

1. **Detailed Design**

Before introducing the system’s work process, it is necessary to introduce the project’s signals because there are many signals, the following **Table 1** is the introduction of signals:

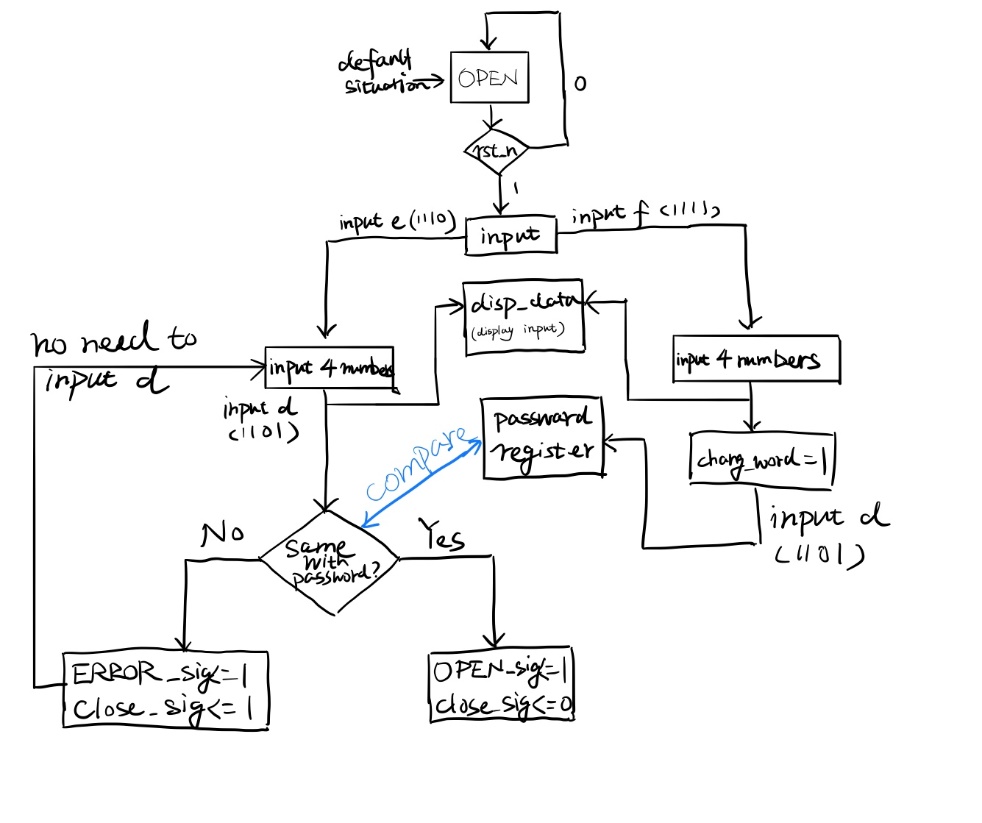
|  |  |
| --- | --- |
| Signal | Use |
| Clk | System’s clock which controls when the output changes as input changes. |
| Rst\_n | Reset, if Reset = 0, the cipher machine will not work correctly due to some other register such as counter have to work by relying on reset. |
| Key\_in | If any number is inputted, key\_in have to be 1 |
| Sw[3:0] | Input switches, it represents the input numbers by users. |
| Close\_sig | Close signal. When passwords is not correct or passwords inputting has not been finished, it is always equal to 1. |
| Open\_sig | Open signal. Only if user finishes a correct password input, the open\_sig will be equal to 1. |
| Error\_sig | Error signal. Only if user finishes an incorrect password input, the error\_sig will be equal to 1. |
| Change\_word | Change signal. Only if user input hexadecimal number ‘f’(change passwords command), the change\_sig will be equal to 1. |
| Seg[6:0] | Segment display ‘AN’ port of Nexys4 Board. |
| Disp\_data[15:0] | Display data, this output signal works for showing user inputting number. |
| Password\_num1[3:0] | This output signal represents the set passwords’ first digital number, ‘f’ represent not yet set or no numbers |
| Password\_num2[3:0] | This output signal represents the set passwords’ second digital number, ‘f’ represent not yet set or no numbers |
| Password\_num3[3:0] | This output signal represents the set passwords’ third digital number, ‘f’ represent not yet set or no numbers |
| Password\_num4[3:0] | This output signal represents the set passwords’ fourth digital number, ‘f’ represent not yet set or no numbers |

**Table 1:** Signals introduction

For this design, there would be an initial password which should be 1234 in hexadecimal number. This also means that the initial password should be 0001, 0010, 0011, 0100 in binary number. Next, if users want to start inputting some numbers, they have to input a hexadecimal number ‘e’ (1110 in binary) first because the project’s system will let user start inputting numbers only if system recognize an input number ‘e’. And if user input hexadecimal number ‘d’, the system would know that user has finished the passwords input then start checking the passwords. For setting password command, users need to input hexadecimal number ’f’, then the system will allow user to set a new 4-digit password. And for the situations of error input, correct input and passwords changing, they all had been introduced in **Table 1.** Besides, if user inputted an incorrect password, then next try will not need users to input ‘e’ again for inputting numbers, users can input directly.

1. **Implement**

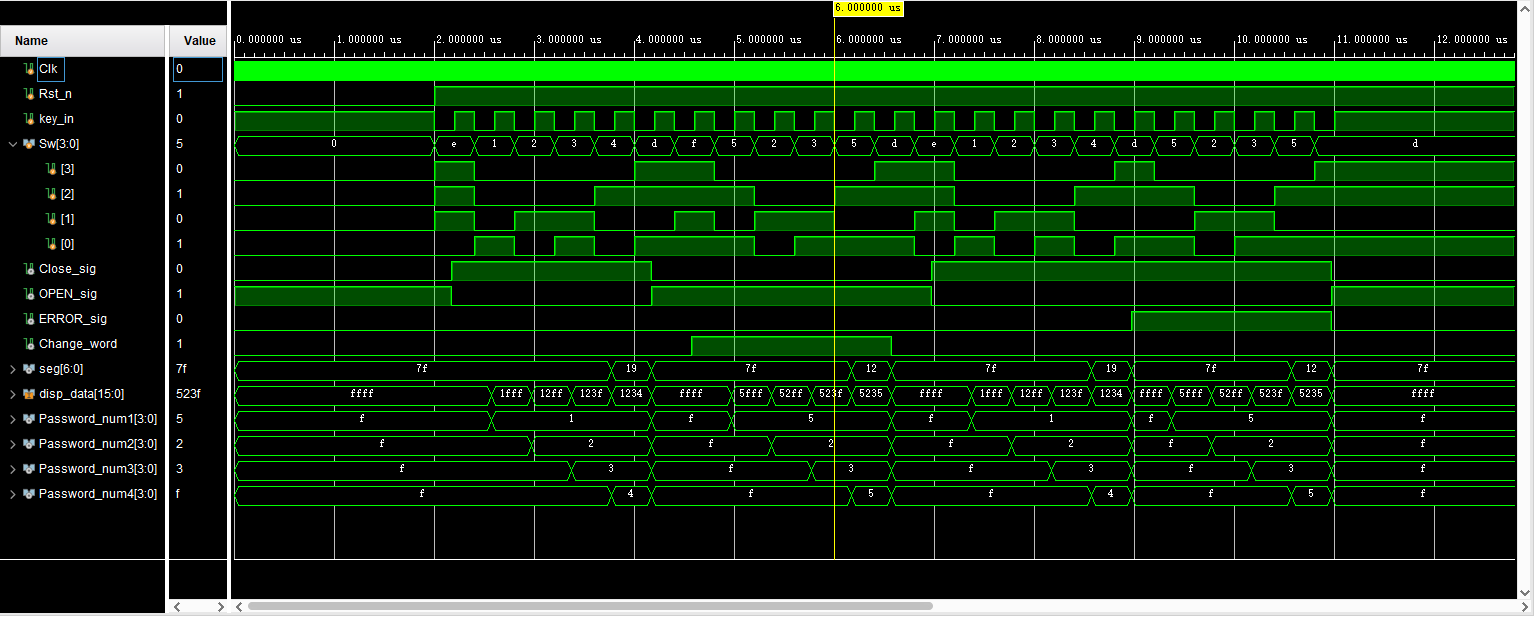
For the implement of this design, the first thing was to determine what signals need to be assign. The basic input signals are absolutely Clk, reset, key\_in, and Sw[3:0]. After that, students need to figure output what would system do if user input correct passwords, input incorrect passwords or want to change passwords. Then the signals Close\_sig, OPEN\_sig, ERROR\_sig, and Change\_word were assigned out. The next thing was to do the remain design – how to let system display the user inputting numbers. After all of that, the most significant thing was to figure out the relationship of every signals. Making a flow chart is the best way to figure out and the best way to explain a project which is shown as following:



**Figure 1:** Flow Chart for Design

1. **Verification**

For this design’s verification, the following **Figure 2** is the simulation result of this lock machine. For details for this simulation, they would be illustrated one part by one part. For the whole process, signals ‘Clk’, ‘Rst\_n’ and ‘key\_in’ are still working as the **Table 1** introduction. Besides, in the whole process of inputting numbers, signals ‘Password\_num1[3:0]’, ‘Password\_num2[3:0]’, ‘Password\_num3[3:0]’, and ‘Password\_num4[3:0]’ were keeping showing user’s inputting 1st, 2nd, 3rd, and 4th numbers. These numbers were transmitted to the output signal ‘disp\_data[15:0]’ for letting user be able to watch what numbers he has inputted. Output ‘seg[6:0]’ expressed the ‘AN’ port displaying situation if the program was downloaded into a Nexy4 Board.



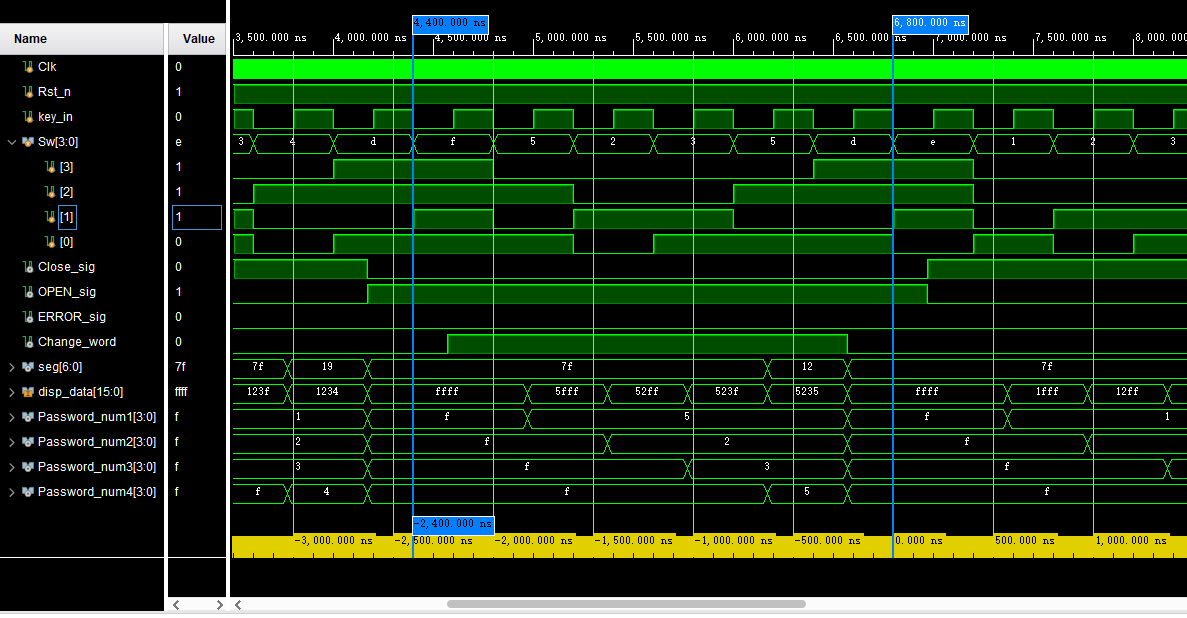
**Figure 2**: Design’s Verification

For first part(between 2,000ns and 4,400ns) of the simulation (**Figure 3),** this part would show the system’s function of checking inputting passwords. For input signal Sw[3:0], it has a hexadecimal input ‘e’ (1110 in binary) which means that user would start inputting numbers. As before introduction, the default password is ‘1234’. At the beginning, the default situation is open before signal ‘Reset\_n’ was inputted as ‘1’ And for this part of simulation, it shows that user inputs a ‘e’ to let system know user want to input numbers, then the user inputs a 4-digit number ‘1234’ which is same with the default password. After that, the user inputs ‘d’ for letting system know user has finished inputting. Then the system checked the passwords and found the password is correct, thus, the signal ‘OPEN\_sig’ became ‘1’ after 4,500 ns a little. And ‘Close\_sig’ became 0. As the figure showing, the ‘disp\_data’ has outputting hexadecimal numbers ‘1fff’, ‘12ff’, ‘123f’, and ‘1234’(hexadecimal number ‘f’ represents no number inputted/no number showing).



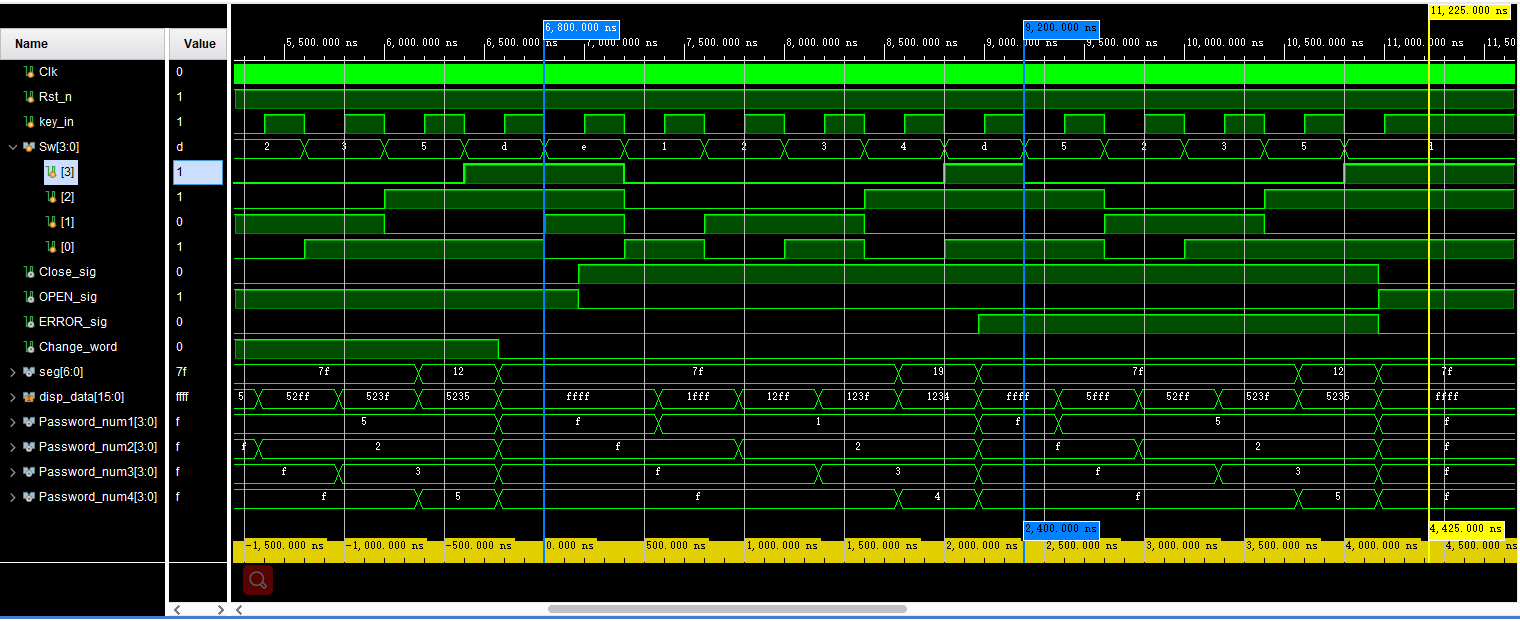
**Figure 3:** First Part of Simulation

For the second part (between 4,400ns and 6,800ns) of the simulation(**Figure 4**), this part would show how the system works for letting user be able to set a new password. For input signal Sw[3:0], it shows that the first input is ‘f’ which means that user want to set a new password. And after a key\_in was input as ‘1’(Press button), the output signal ‘Change\_word’ became ‘1’ which means that system allows user to change a new password by inputting a 4-digit number. Then the user inputs a 4-digtal number ‘5235’ which means that the new password will be ‘5235’(0101 0010 0011 0101). After finishing inputting, user inputs a hexadecimal number ‘d’ to let system know user has finished inputting. Then signal ‘Change\_word’ became back to 0. And as the figure showing, the ‘disp\_data’ has outputting hexadecimal numbers ‘5fff’, ‘52ff’, ‘523f’, and ‘5235’(hexadecimal number ‘f’ represents no number inputted/no number showing).



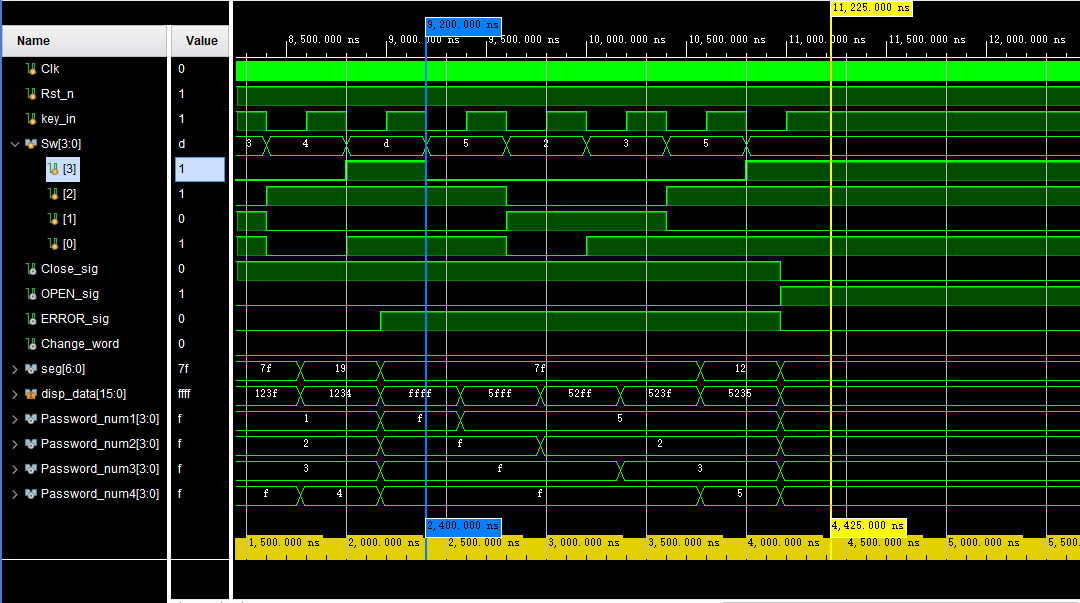
**Figure 4:** Second Part of Simulation

For the third part(between 6,800ns and 9,200ns) of simulation(**Figure 5**), this part would show what system will output if user input an incorrect password. For input signal Sw[3:0], user inputs a hexadecimal number ‘e’ to let system allow user to input numbers. Then the user input a 4-digit number ‘1234’, but this is not the correct password now. The correct password should be 5235 now. Thus, after user finishes inputting by inputting ‘d’ and want system to check the inputting number, the output signal ‘ERROR\_sig’ became ‘1’ and ‘Close\_sig’ was always equal to ‘1’. It means that the lock cannot open because the user inputted a wrong password.



**Figure 5:** Third Part of Simulation

For the fourth part (between 9,200ns and 11,225ns) of simulation (**Figure 6**),this part would show the system will recognize the new set password which was set as ‘5235’ at second part. As before mentioned, if user inputs an incorrect password, then the next try does not need user input ‘e’ again. Then the user inputted a correct password ‘5235’. After finishing inputting, user still needs to input a hexadecimal number ‘d’. Then system found that the inputting number is correct password. Thus ,the output signals ‘ERROR\_sig’ and ‘Close\_sig’ became ‘0’. And ‘OPEN\_sig’ became ‘1’. And as the figure showing, the ‘disp\_data’ has outputting hexadecimal numbers ‘5fff’, ‘52ff’, ‘523f’, and ‘5235’(hexadecimal number ‘f’ represents no number inputted/no number showing).



**Figure 6:** Fourth Part of Simulation

**Conclusion**

For this lab design, this can help students to review everything learnt in this semester in this process of design. Students would think about what they have ability to do and cannot do. And for this individual design, the project had used the many knowledge such as FSM, blocking/unblocking and debouncing etc. Besides, the difficult part to design a project was to figure out the whole project’s structure. And most difficult thing was to find where is the problem when results were wrong. Because the project has many register, input and output.