National University of Ho Chi Minh City
POLYTECHNIC UNIVERSITY
SCIENCE - COMPUTER ENGINEERING

DESIGN THEORY WITH VERILOG HDL

Topic 5

Door lock

Instructor: Mr. Ngo Duc Minh
Student: Le Khac Minh Dang - MSSV: 1810109
Huynh Phuc Khanh - MSSV: 1810226
Nguyen Tran Quang Minh - MSSV: 1811083

City. HO CHI MINH CITY, DECEMBER 2019

Table of contents

1 Introduction

1.1 Topic 5: Door lock

The topic selected to be implemented in this report is topic 5: Door lock. In general, the requirements of the project are as follows: Realizing a door lock circuit using a state machine has the following functions:

- The password is a three-digit hexadecimal number from 0 to F.

- Digits are entered through the SWs.

- Only one digit is entered at a time, KEY1 is used to confirm the input one digit. The entered digit must appear on the 7-segment LED.

- The circuit operates in two modes: Set and Verify. In Set mode, the user will set the password. In Verify mode, the user enters the password and the system checks it. The two modes can be switched through the KEY2.

- In Verify mode: If the entered password is correct, the green LED will blink at 5 Hz. If the password is wrong, the red LED will blink at 1 Hz. Then the user has 2 more attempts. If in these 2 times, the user enters the correct password, the number of attempts will return to the original.
  Otherwise, the system will be locked and can only work again if the user presses the reset button (KEY3).

1.2 Tools used

Verilog language HDL Verilog
is a hardware description language (Hardware Description Language) used in the design of digital systems, integrated circuits: such as RAM, microprocessors, etc. . . [6]

Verilog was first introduced in 1984 by the company Gateway Design Auto matic. Verilog is not standardized and is edited in most versions later versions from 1984 to 1990. In 1995 Verilog was officially standardized by the IEEE (Institute of Electrical and Electronics Engineers).

Verilog is a language quite similar to C programming language. However, it still has superior advantages over C, especially for hardware design such as Verilog can execute instructions sequentially or in parallel. song.

Hardware: Board DE2i-150 In the
project, the team used Altera's DE2i-150 Board. The board is composed of [2]:

- CPU : Intel R AtomTM Dual Core Processor N2600 (1M Cache, 1.6GHz).
- Altera Cylone IV 4CX150.
- Two sticks of 2MB SSRAM and two sticks of 64MB SDRAM.
- Four buttons (KEY0-KEY3).
- 18 Switches.
- 18 red LEDs and 9 green LEDs.
- The 7 segment LEDs.
- A 50MHz counter for clock calculation.
- Other ingredients.

Software: Quartus, ModelSim and draw.io Quartus is a
software tool developed by Altera that provides an environment for SOPC (System On A Programmable Chip) designs. This is a fully integrated software for logic design with Altera's PLD (Programmable Logic Device) components, including APEX, CyClone, FLEX, MAX, Stratix, . . . [3]

ModelSim is a very powerful and efficient simulation and debugging program for ASIC and FPGA designs. ModelSim provides people with Waveform of FPGA circuits designed on Quartus. From there, it helps the designer to control the output value and detect the errors of the circuit [4].

draw.io is a website that supports a simple, compact, lightweight and free drawing tool for drawing models and charts [5].

2 Design and Reality

2.1 Modules

Door lock design includes 8 modules:

- posedge_detector() is used to detect rising edges for KEYs.
- led7_decoder() is used to decode the hexadecimal signal to the 7-segment LED.
- pass_decoder() is used to display 3-digit password in 7-segment LED.
- LED_blinker() is used to blink the LED according to the frequency provided via pa rameter.
- state_decoder() is used to display the current status of 7-segment LEDs (including S, V, and F states).
- password_getter_sync() is a synchronous circuit, used to receive input digit-by-digit from the user via SWs and KEYs.
- door_lock_FSM() is a synchronous Mealy state machine that handles all the whole logic of the design.
- door_lock_top() is the top module used to connect small modules together.

2.2 Tree of components model

The design elements can be redrawn as a tree diagram as follows:

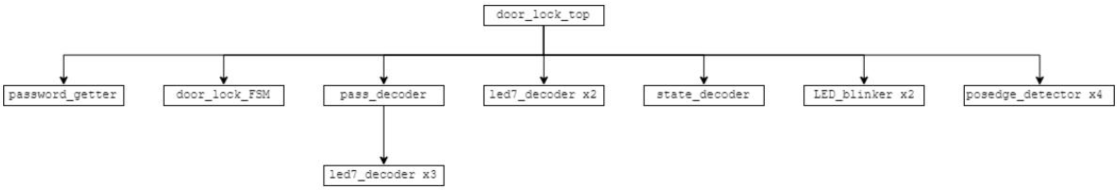


Figure 1: Door lock hierarchy

2.3 Block diagram of connections

The door_lock_top() module has a block diagram of connections as follows:

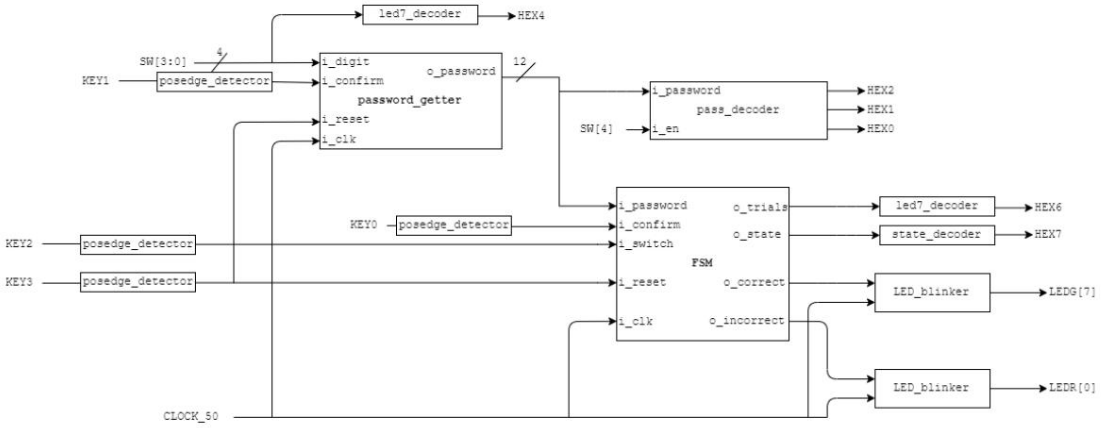
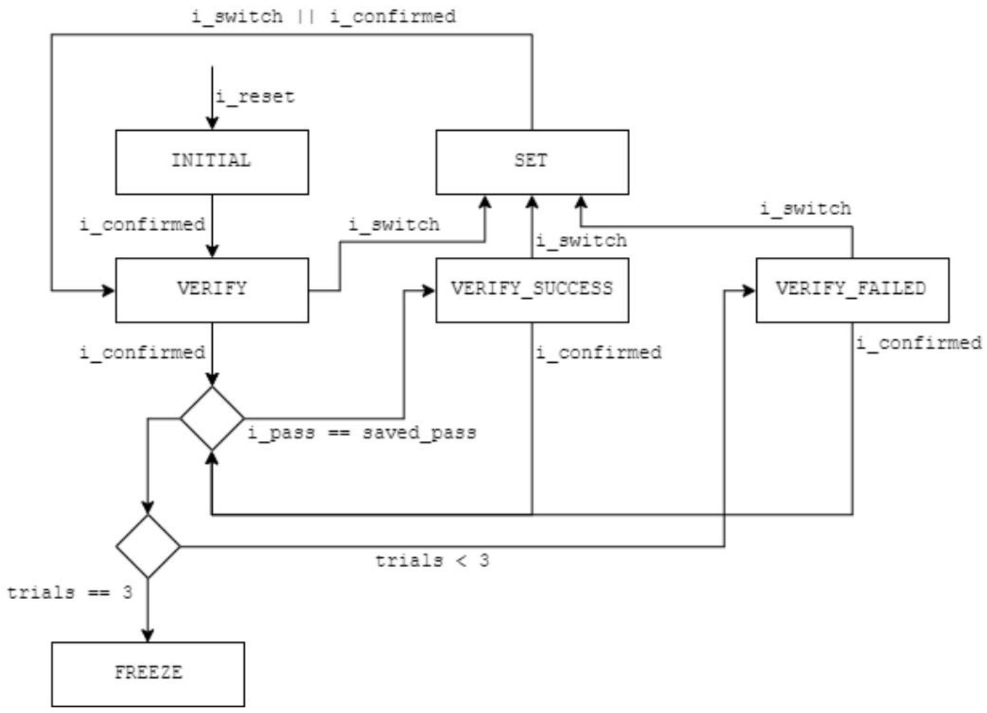


Figure 2: Block diagram of door_lock_top() and inputs from DE2i-150 . circuit

2.4 State machine

The state machine used in this design is a synchronous Mealy state machine. The state machine operates according to the flow graph as follows:



Hình 3: FSM flow graph

3 Simulation and testing

3.1 Emulation

The software used to test the modules is ModelSim. For each module the testcases are as follows (there are 7 modules tested):

- led7_decoder(): first, signal enable = 0, then enable = 1 and all 4-bit numeric values from 0 to F are tested.

- pass_decoder(): first, signal enable = 0, then enable = 1 and 16 values Random value of the password to be tested.

- LED_blinker(): with frequency 5Hz, first, enable signal = 0, then signal enable is held to 1 for a period of time.

- state_decoder(): all 6 states of the state machine are tested.

- password_getter_sync(): first, the circuit is reset, then 5 random digits are passed in.

- door_lock_FSM(): state machine is tested as follows: reset -> INITIAL state -> VERIFY state with correct password -> VER IFY state with wrong password 3 times -> FREEZE state -> reset -> SET status -> VERIFY status with new password.

- door_lock_top(): test similar to state machine.

The results when running the testbenches are as follows:

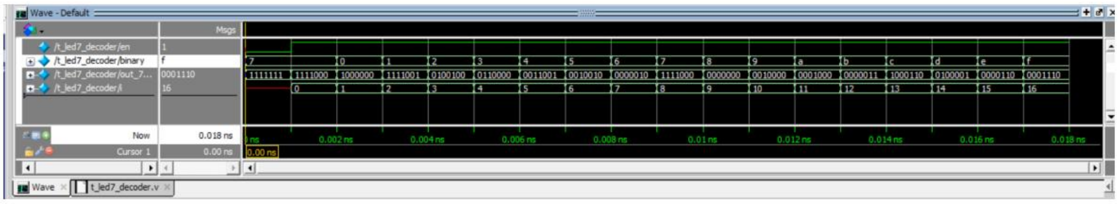


Figure 4: Test result of led7_decoder() module

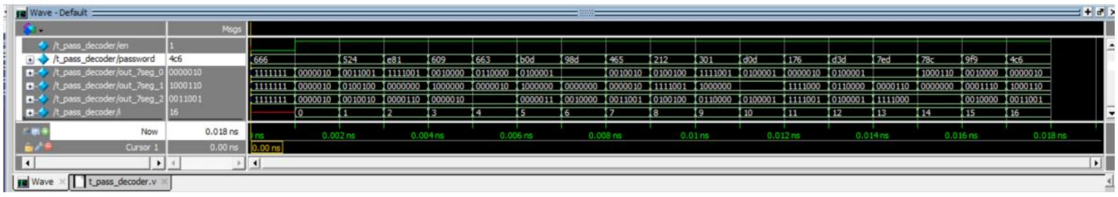

Wave - Default
Msgs
/t_pass_decoder/en 1
/t_pass_decoder/password 4c6
/t_pass_decoder/out_7seg_0 0000010
/t_pass_decoder/out_7seg_1 1000110
/t_pass_decoder/out_7seg_2 0011001
/t_pass_decoder/i 16
Now 0.018 ns
Cursor 1 0.00 ns
Wave
t_pass_decoder.v

Figure 5: Test results of the pass_decoder() module

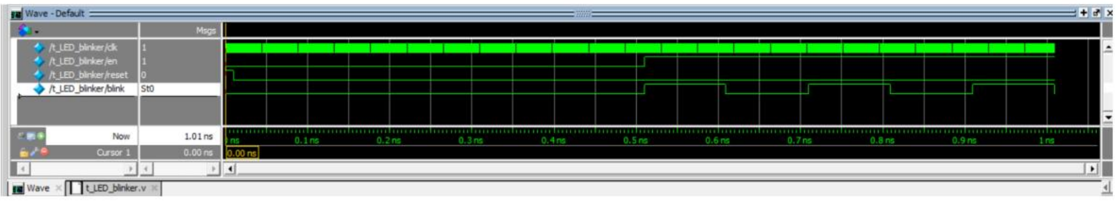


Figure 6: Test results of LED_blinker() module

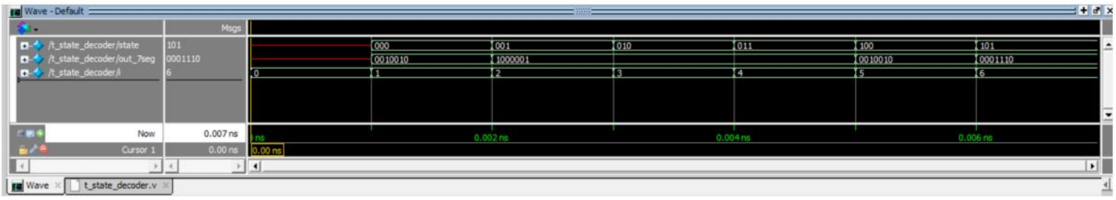


Figure 7: Test results of the state_decoder() module

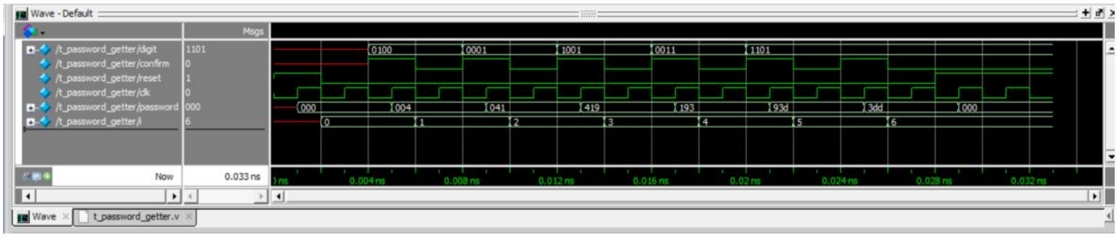


Figure 8: Test result of password_getter() module

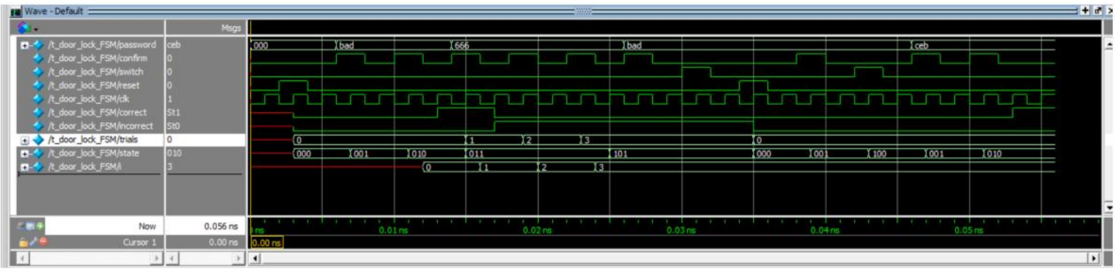
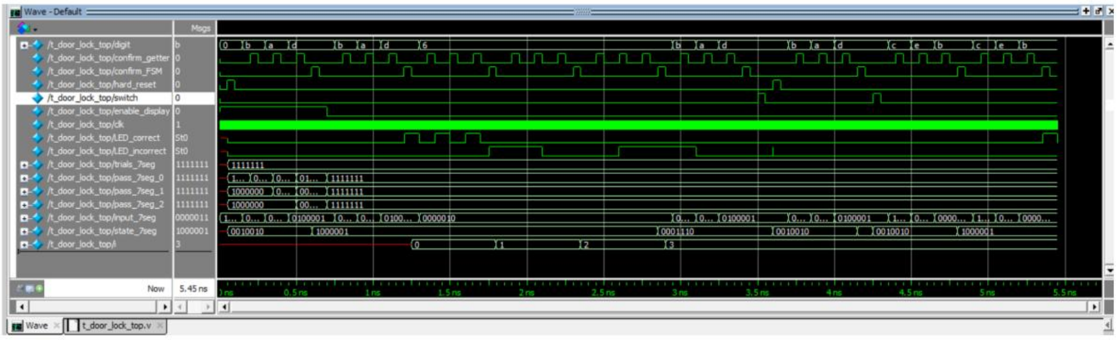


Figure 9: Test results of the door_lock_FSM() module



Wave - Default

	Msgs
/t_door_lock_top/digit	b
/t_door_lock_top/confirm_getter	0
/t_door_lock_top/confirm_FSM	0
/t_door_lock_top/hard_reset	0
/t_door_lock_top/switch	0
/t_door_lock_top/enable_display	0
/t_door_lock_top/clk	1
/t_door_lock_top/LED_correct	St0
/t_door_lock_top/LED_incorrect	St0
/t_door_lock_top/trials_7seg	1111111
/t_door_lock_top/pass_7seg_0	1111111
/t_door_lock_top/pass_7seg_1	1111111
/t_door_lock_top/pass_7seg_2	1111111
/t_door_lock_top/input_7seg	0000011
/t_door_lock_top/state_7seg	1000001
/t_door_lock_top/i	3

Now 5.45 ns

Figure 10: Test result of module door_lock_top()

3.2 Onboard test results

Here are some test results on the DE2i-150 board:

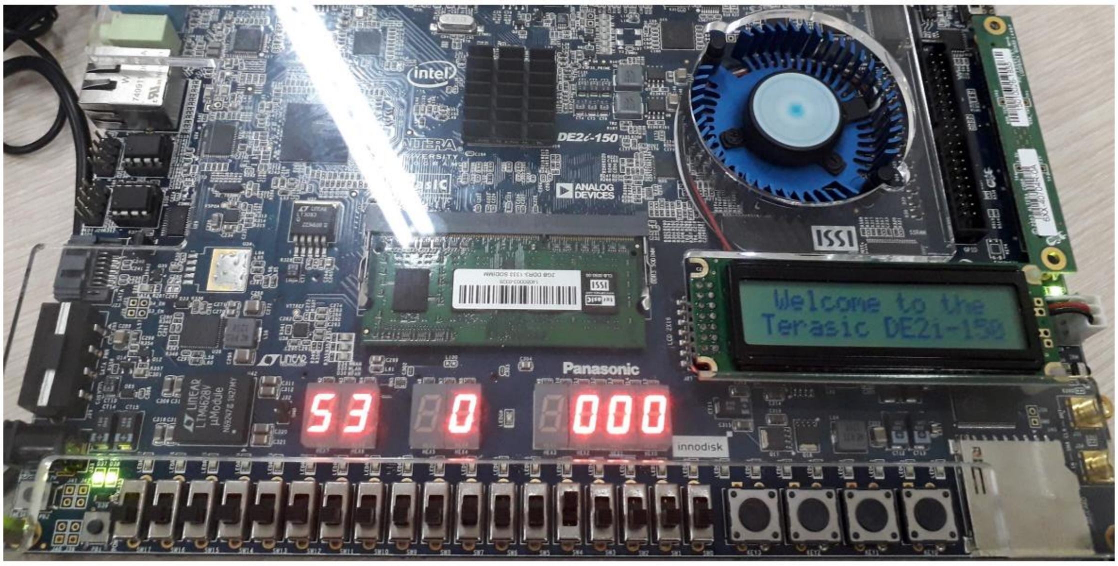


Figure 11: Circuit in SET state, all output = 0 after reset

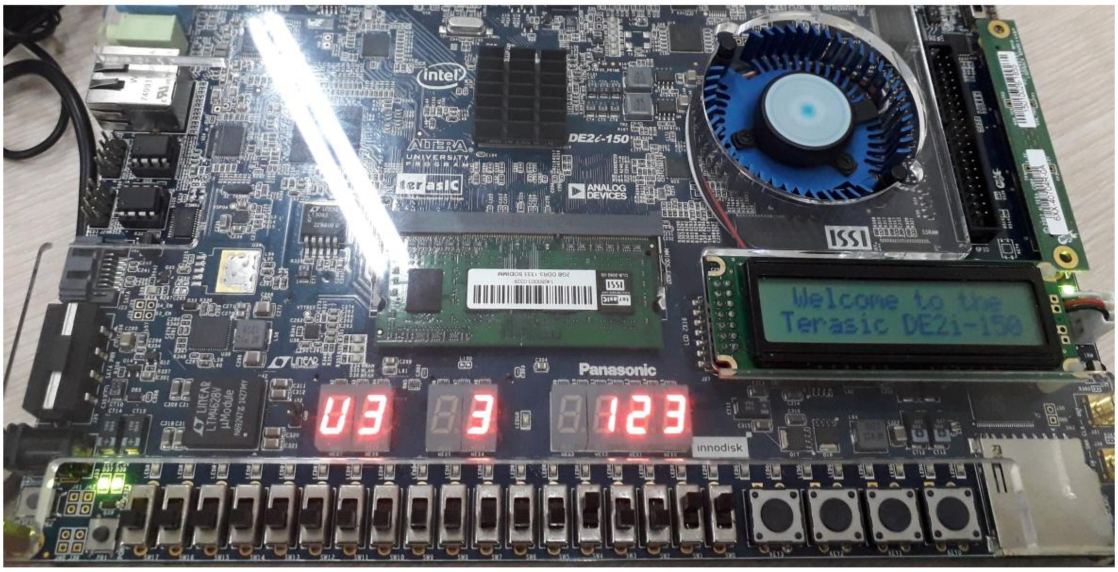


Figure 12: The circuit is in the VERIFY state, the password has just been set to 123

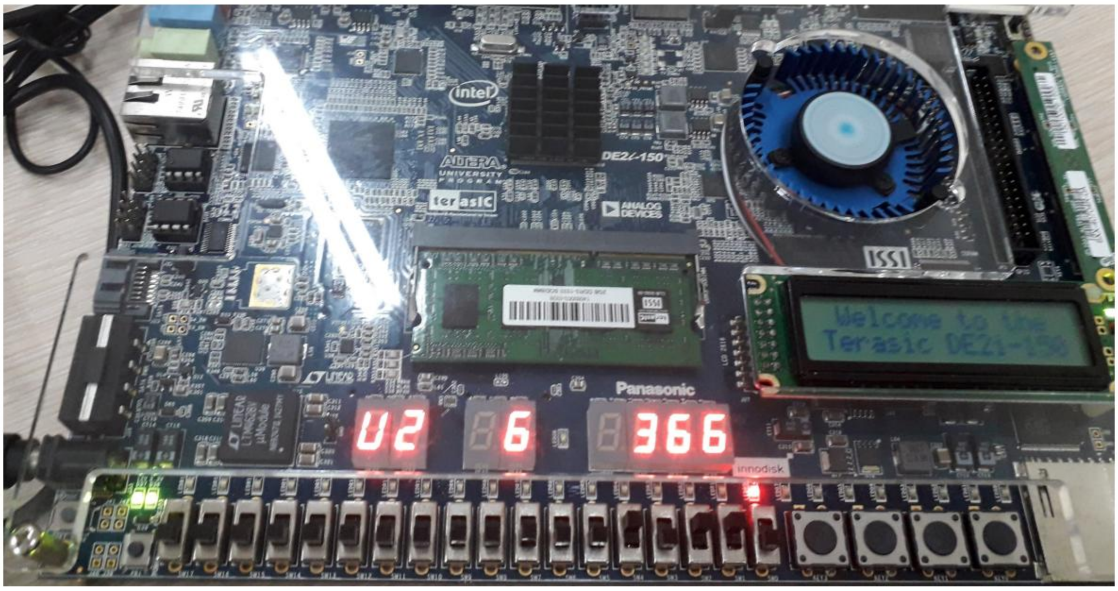


Figure 13: The circuit is in VERIFY state, 366 is the wrong password, so LEDR is on

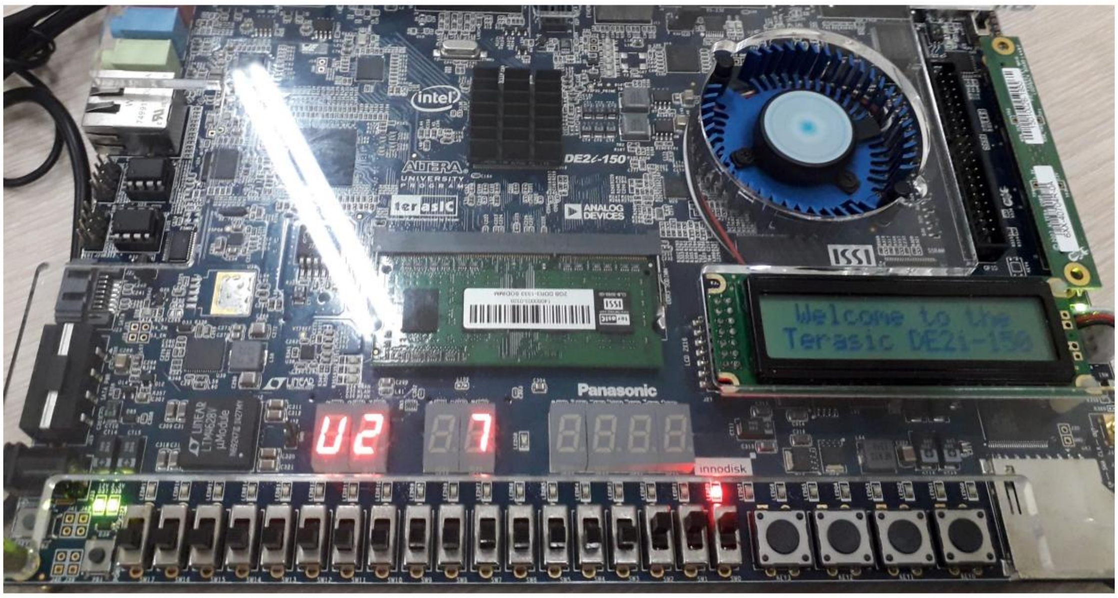


Figure 14: Circuit in VERIFY state, SW[4] = 0, turn off enable signal of pass_decoder() so password is not displayed

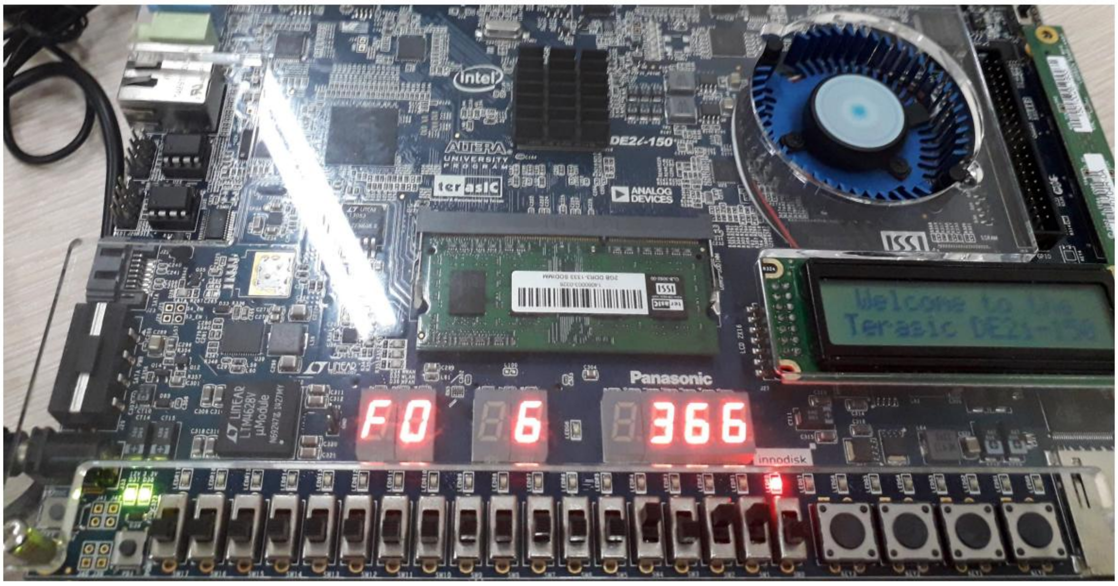


Figure 15: The circuit is in the FREEZE state after entering the wrong password 3 times

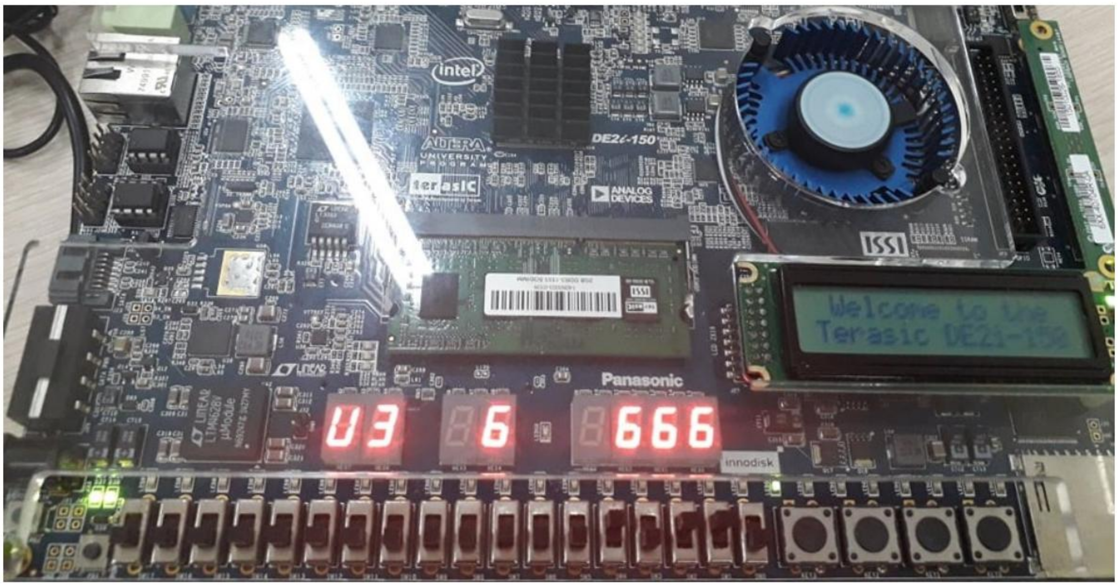


Figure 16: The circuit is in VERIFY state, 666 is the correct password, so LEDG is on

In addition, the circuit operation can also be viewed through the following video: https://youtu.be/6LMBHBavZH4

4 Conclusion

To summarize, this Door lock design has the following functions:

- Reset button KEY[3] is used to return the circuit to the initial password set state.
- The confirm_getter KEY[0] button together with the four SW[3:0] keys are used to enter a single digit in the password.
- The confirm_FSM KEY[1] button is used to enter the password into the processing state machine.
- Switch button KEY[2] is used to switch between the two states SET and VERIFY.
- When in the set state, if there is a confirm_FSM signal, the new password will be given save and the status will change to VERIFY.

- In the VERIFY state, if there is a confirm_FSM signal, the password will be checked. If true, LEDG will blink at 5Hz, if false, LEDR will blink at 1Hz. If wrong up to 3 times, the circuit will go to the FREEZE state.
- In the FREEZE state, the machine is completely inactive.
- There is also a SW[4] key to choose whether to display the 7-segment LED output password or not.
- The parameters displayed on the 7-segment LED are: current status, number of attempts remaining again, current value on SW[3:0], current password.

Design advantages:

- Using the Mealy state machine saves the number of states.
- Receiving input digit by digit reduces the amount of SW needed.
- SW[4] can choose whether to display the output password or not.
- Display current status to 7-segment LED to help d6e4 easily know the operation circuit motion.

Disadvantages of the design:

- The Mealy state machine is synchronous, so the output will be delayed by 1 clock cycle.
- Buttons can be noisy when pressed indiscriminately.

Document

[1] M. Morris Mano, Michael D. Ciletti, Digital System with an Introduction to the Verilog HDL, VHDL, and SystemVerilog, Pearson Education, Inc, 2017.

[2] https://www.intel.com/content/www/us/en/programmable/solutions/partners/partner profile/terasic-inc-/board/de2i-150-fpga-development-kit.html

[3] https://www.element14.com/community/docs/DOC-40098/l/altera introduction-to-the-quartus-ii-software

[4] https://www.mentor.com/products/fpga/model/

[5] https://snvn.net/4rum/viewtopic.php?t=80

[6] Slide lecture on Logical Design with verilog HDL by Mr. Pham Quoc Cuong, Edited by Master Kieu Do Nguyen Binh.

Last visit of websites: 7/12/2019.

Work assignment:

Le Khac Minh Dang	Design, main code, write reports part 2, 3, synthesis report
Nguyen Tran Quang Minh	Support code, write testbenches, write report part 1
Huynh Phuc Khanh	Support code, write testbenches, write report part 4