Name: _____ ID# _____

Date Submitted: _____ Time Submitted _____

CSE 4356/5356/EE 5315 System on Chip (SoC) Design Fall Semester 2021

Lab Assignment #1 – Digital Lock Controller

Due Date – September 6, 2021 (11:59 PM)

Submit on Canvas Assignments

CSE4356/CSE5356/EE4328/EE5315 **FALL 2021**

LABORATORY ASSIGNMENT #1

DIGITAL LOCK CONTROLLER

(100 POINTS)

PURPOSE/OUTCOMES

To design in Verilog, implement on the DE1-SoC, and test a Moore-type sequential circuit that functions as the controller for a digital lock. Once you complete this assignment, you will have demonstrated an ability to design a sequential circuit in Verilog that meets specified requirements and to implement the design using a Field Programmable Gate Array (FPGA).

BACKGROUND

Sequential circuits can be used as sequence detectors and, hence, can be designed as the controller for a digital lock. Sequences can be considered as block or non-block codes. Block codes have a fixed length and are not overlapping for detection purposes. A non-block code has no fixed length and may contain overlapping segments. For example consider the one-bit code 1-0-1-1. The sequence 1-0-1-1-0-1-1-0 would produce output sequences 0-0-0-1-0-0-0-0 and 0-0-0-1-0-0-1-0 for block and non-block detectors, respectively.

For block codes, the number of possible code sequences of length n is m^n where m is the number of symbols in the code. For the above example, $m=2$ since 0 and 1 are the code characters and n=4. So the number of code sequences is $2^4 = 16$.

Sequential circuits can be designed as Mealy machines or Moore machines. The current output of a Mealy machine is a function of the current input and current state of the machine. Consequently, unwanted spurious outputs, or glitches, may occur if inputs change between state changes. Spurious outputs cannot occur in Moore machines since the current output is a function of only the current state. Mealy machines usually require fewer states than an equivalent Moore machine but that is preferred when spurious outputs must be avoided.

DESIGN REQUIREMENTS

Your assignment is to design, simulate, implement, and test a controller for a digital lock as shown in figure 1 that meets the requirements specified below.

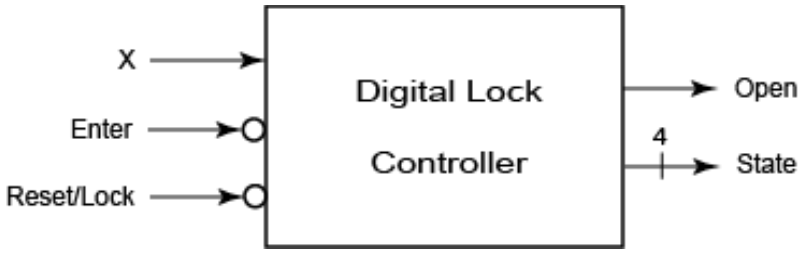

Figure 1. Digital Lock Controller Block Diagram

1. *X is* used to input the binary code sequences for unlocking the lock.
2. *Enter* is used to tell the controller to read the current input value.
3. *Reset/Lock* is used to restart the code entry process or to lock the lock after it's opened.
4. The lock will be opened (*Open = 1*) **if and only if** code sequence *X:1-1-1-0-1-1-1* is entered.
5. The current state of the controller should be displayed on output State.
6. The controller must then stay in the Open = 1 state until locked (Open = 0).

DESIGN PROCESS (30 points)

1. Develop the state diagram and state table for a Moore machine that meets the above design requirements.
2. Write a Verilog model of your design.
3. Compile your Verilog code.
4. Correct any syntax errors.

DESIGN VERIFICATION (20 points)

1. Simulate your design using ModelSim to verify its correctness. Use the following input sequence in your simulation where Λ represents a reset.

   X: 1100101Λ1110111Λ0101010Λ0101011Λ0001000Λ1111111Λ11110111

2. Record the simulation results for your report.

DESIGN ENHANCEMENT (10 POINTS)

1. Add a binary-to-seven-segment decoder to your project. You can use the one we designed in class or one you wrote on your own.
2. Instantiate the decoder in your Verilog code to display the controller state on the HEX0 seven-segment display.
3. Compile your code.
4. Record your Verilog lisitng and compilation summary for your report.

DE1-SoC iMPLEMENTATION (25 points)

1. Implement your design on the DE1-SoC using the following pin assignments. The DE1-SoC pin assignments can be found in the attached table.

   X = SW0
   Enter = KEY1
   Reset/Lock = KEY0
   Open = LEDR0
   State = HEX0

2. Program the DE1-SoC with your design.

DE1-SoC TESTING (15 points)

1. Test your implementation by applying the following input sequence.

   X: 1100101Λ1110111Λ0101010Λ0101011Λ0001000Λ1111111Λ11110111

2. Record the test results for your report.
3. Take a picture of your results showing LEDR0 lighted and HEX0 in the correct state.

LAB REPORT REQUIREMENTS

1. Cover sheet
2. Design requirements
3. State diagram
4. Verilog code
5. Simulation results
6. Pin assignments
7. Test results

DE1-SoC I/O Pin Assignments

Input Switches	***LED Outputs***	***Seven-Segment***
SW0 – PIN_AB12	LEDR0 – PIN_V16	HEX0(0) – PIN_AE26
SW1 – PIN_AC12	LEDR1 – PIN_W16	HEX0(1) – PIN_AE27
SW2 – PIN_AF9	LEDR2 – PIN_V17	HEX0(2) – PIN_AE28
SW3 – PIN_AF10	LEDR3 – PIN_V18	HEX0(3) – PIN_AG27
SW4 – PIN_AD11	LEDR4 – PIN_W17	HEX0(4) – PIN_AF28
SW5 – PIN_AD12	LEDR5 – PIN_W19	HEX0(5) – PIN_AG28
SW6 – PIN_AE11	LEDR6 – PIN_Y19	HEX0(6) – PIN_AH28
SW7 – PIN_AC9	LEDR7 – PIN_W20	
SW8 – PIN_AD10	LEDR8 – PIN_W21	HEX1(0) – PIN_AJ29
SW9 – PIN_AE12	LEDR9 – PIN_Y21	HEX1(1) – PIN_AH29
		HEX1(2) – PIN_AH30
KEY0 – PIN_AA14	***Clocks***	HEX1(3) – PIN_AG30
KEY1 – PIN_AA15	CLOCK_50 – PIN_AF14	HEX1(4) – PIN_AF29
KEY2 – PIN_W15	CLOCK2_50 – PIN_AA16	HEX1(5) – PIN_AF30
KEY3 – PIN_Y16	CLOCK3_50 – PIN_Y26	HEX1(6) – PIN_AD27
	CLOCK4_50 – PIN_K14	
		HEX2(0) – PIN_AB23
Expansion Header 1	***Expansion Header 1***	HEX2(1) – PIN_AE29
GPIO_1(0) – PIN_AB17	GPIO_1(18) – PIN_AK26	HEX2(2) – PIN_AD29
GPIO_1(1) – PIN_AA21	GPIO_1(19) – PIN_AH25	HEX2(3) – PIN_AC28
GPIO_1(2) – PIN_AB21	GPIO_1(20) – PIN_AJ25	HEX2(4) – PIN_AD30
GPIO_1(3) – PIN_AC23	GPIO_1(21) – PIN_AJ24	HEX2(5) – PIN_AC29
GPIO_1(4) – PIN_AD24	GPIO_1(22) – PIN_AK24	HEX2(6) – PIN_AC30
GPIO_1(5) – PIN_AE23	GPIO_1(23) – PIN_AG23	
GPIO_1(6) – PIN_AE24	GPIO_1(24) – PIN_AK23	HEX3(0) – PIN_AD26
GPIO_1(7) – PIN_AF25	GPIO_1(25) – PIN_AH23	HEX3(1) – PIN_AC27
GPIO_1(8) – PIN_AF26	GPIO_1(26) – PIN_AK22	HEX3(2) – PIN_AD25
GPIO_1(9) – PIN_AG25	GPIO_1(27) – PIN_AJ22	HEX3(3) – PIN_AC25
GPIO_1(10) – PIN_AG26	GPIO_1(28) – PIN_AH22	HEX3(4) – PIN_AB28
GPIO_1(11) – PIN_AH24	GPIO_1(29) – PIN_AG22	HEX3(5) – PIN_AB25
GPIO_1(12) – PIN_AH27	GPIO_1(30) – PIN_AF24	HEX3(6) – PIN_AB22
GPIO_1(13) – PIN_AJ27	GPIO_1(31) – PIN_AF23	
GPIO_1(14) – PIN_AK29	GPIO_1(32) – PIN_AE22	HEX4(0) – PIN_AA24
GPIO_1(15) – PIN_AK28	GPIO_1(33) – PIN_AD21	HEX4(1) – PIN_Y23
GPIO_1(16) – PIN_AK27	GPIO_1(34) – PIN_AA20	HEX4(2) – PIN_Y24
GPIO_1(17) – PIN_AJ26	GPIO_1(35) – PIN_AC22	HEX4(3) – PIN_W22
		HEX4(4) – PIN_W24
		HEX4(5) – PIN_V23
		HEX4(6) – PIN_W25
		HEX5(0) – PIN_V25
		HEX5(1) – PIN_AA28
		HEX5(2) – PIN_Y27
		HEX5(3) – PIN_AB27
		HEX5(4) – PIN_AB26
		HEX5(5) – PIN_AA26
		HEX5(6) – PIN_AA25