JOINT INSTITUTE
交大密西根学院

Ve 270 Introduction to Logic Design

Lab 6
Design of a Keypad Scanner

UM-SJTU Joint Institute
Shanghai Jiao Tong University
July 2020

1. Objective

To design a Finite State Machine that reads the keys from a 4-by-4 keypad and displays the corresponding hexadecimal value on an SSD.

2. Background

Keypad scanners are used to interpret data entered from keypads for digital devices such as phones, calculators, digital lock, etc. A keypad scanner decodes a pressed key and outputs the corresponding binary code. Figure 1 shows the connection between a hexadecimal keypad and the keypad scanner circuit.

When a button is pressed, it connects a row and a column at the location of the button. By providing a 1 to a column of the keypad, a 1 will be received from the connected row. In other words, the pressed key creates a loop for the signal 1 to travel from the keypad scanner, through the loop created by the pressed key, and back to the keypad scanner.

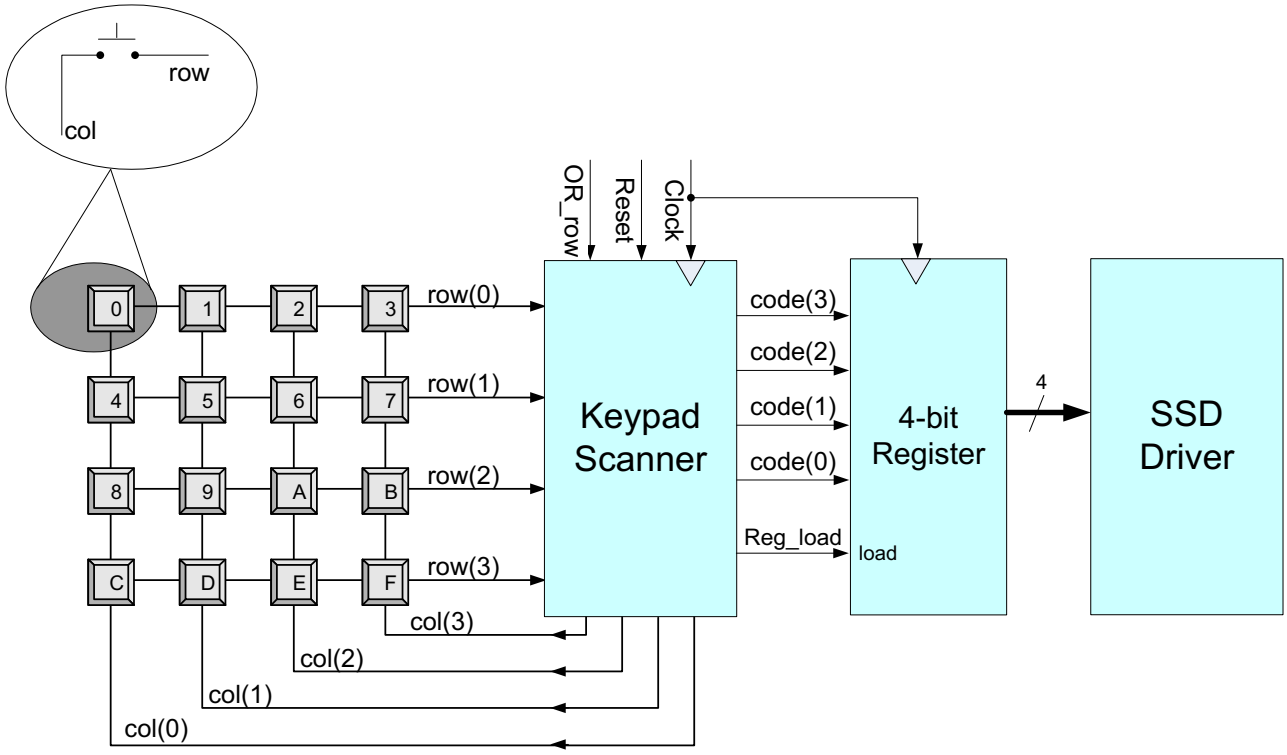

Figure 1. Keypad Scanner

In order to avoid the floating input condition on the rows, all the rows should be securely pulled up to the power supply or pulled down to the ground when they are not connected to anything. In this lab, the design described below assumes all the inputs are pulled down to the ground, as shown in the following figure.

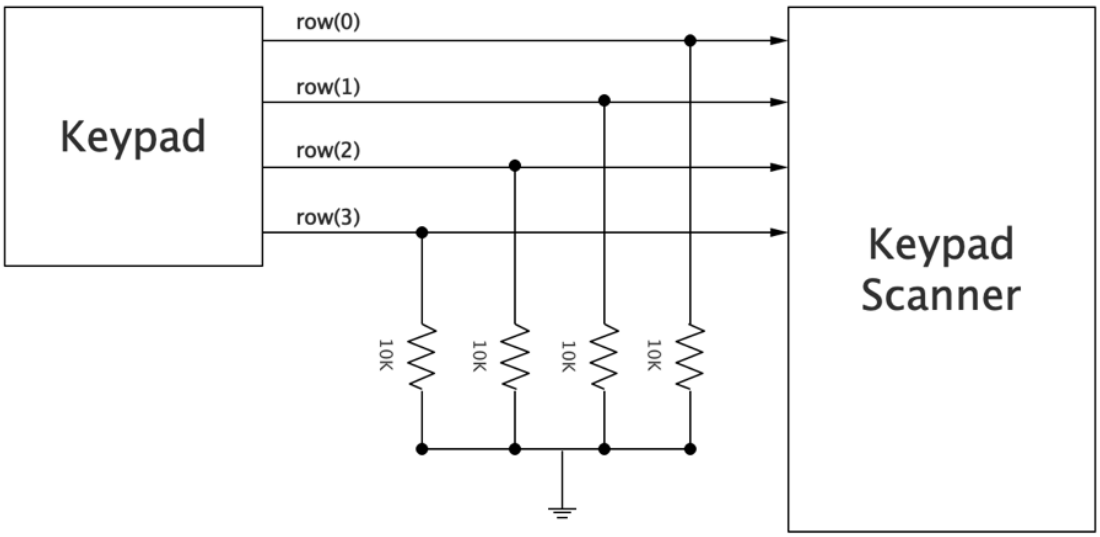

Figure 2. Pull-down circuit

The keypad scanner must be able to fire a column line to detect the location of a pressed button. A keypad scanner detects a pressed key in three steps:
(1) Detects whether a key is pressed;
(2) Identifies which key is pressed;
(3) Generates the corresponding 4-bit hexadecimal code for the pressed key.

Step (1) can be achieved by outputting "1111" on the *col* output, so that one of the *row* inputs will be asserted whenever there is a key pressed. An OR logic of all the *row* inputs (*OR_row*) will be used to indicate whether any row is high. If the keypad scanner detects a high on the *OR_row* input, it performs step (2) in which the *col* outputs will be turned on sequentially. This will tell which row and column are connected. Thus, the pressed key can be located. Step (3) will be done by looking it up in Table 1. After the 4-bit code is generated, it will be stored in a 4-bit register for display.

Table 1. Keypad code for the Hex keypad

Key	row (3 : 0)	col (3 : 0)	code (3 : 0)
0	0001	0001	0000
1	0001	0010	0001
2	0001	0100	0010
3	0001	1000	0011
4	0010	0001	0100
5	0010	0010	0101
6	0010	0100	0110
7	0010	1000	0111
8	0100	0001	1000
9	0100	0010	1001
A	0100	0100	1010
B	0100	1000	1011
C	1000	0001	1100
D	1000	0010	1101
E	1000	0100	1110
F	1000	1000	1111

3. Design of the Keypad Scanner

This keypad scanner can be designed as a Finite State Machine. The key detecting algorithm is captured in Figure 3. The machine begins in State0, with all of the *col* outputs asserted, until one of the *row* inputs is asserted. From State1 to State4, only one of the *col* outputs is asserted. If one of the *row* inputs is also asserted, the pressed key is located. An output code can be determined by the combination of *col* outputs and *row* inputs. The FSM then moves to State5 to State8, respectively, where the corresponding *code* is generated as well as a signal *Reg_load* that controls load of the 4-bit register connected to the outputs of the keypad scanner. After one clock cycle, the FSM moves to State9 with all the *col* outputs asserted again until the *OR_row* is de-asserted which indicates the key is released.

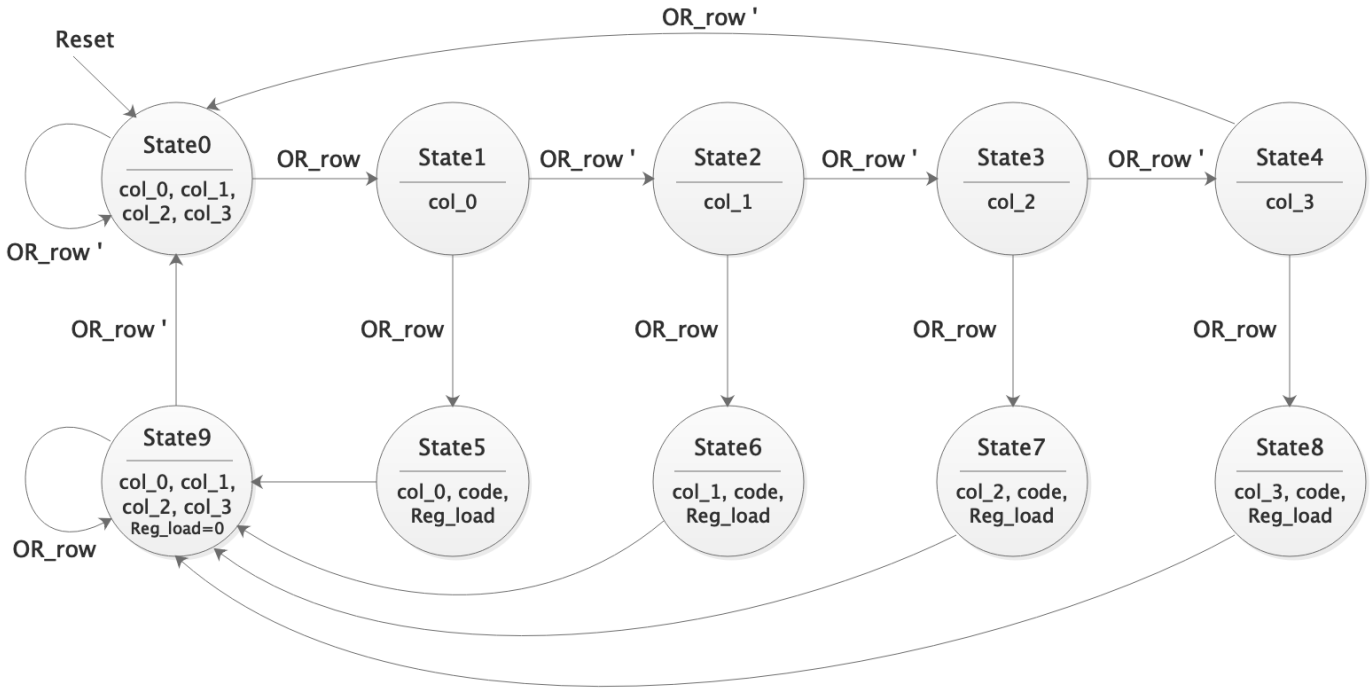

Figure 3. State diagram of the keypad scanner

4. Verilog Coding

Write a Verilog module for the Kaypad Scanner as an FSM following the state diagram shown in Figure 3. The FSM should have an asynchronous reset. Modify the Verilog module for SSD Driver developed in Lab 5 to display the interpreted hexadecimal value.

5. Simulation, Synthesis, and FPGA Implementation

Simulate the top-level integrated circuit using HDL Bencher. Synthesize and implement your design on the Basys 3 FPGA board.

6. Deliverable

This is a 2-week lab. The full score for this lab is 200 points.

1) Demonstrate your circuits to the TAs before your lab session ends.
2) Upload source files on Canvas by **10pm, July 18, 2020**.