NAME + ROLL #

Laboratory 5 – Introduction to Finite State Machines

In this laboratory you will learn about a Finite State Machines (FSM), an especially useful application of Sequential Logic. You will design a simple FSM's and implement it using gates and flip-flops.

Procedure:

You are asked to design and build an FSM for solving the stated problem. As we talked about the design process in class, we expect you to design the FSM in a logical manner. So here are a few hints – Do the following steps and discuss your write-up with a TA before proceeding to the next step:

1. Describe the FSM in words – put down in words what you expect it to do.
2. Draw state transition diagram of the FSM which shows the allowed states of the machine and the logic that causes transitions between the states.
3. Once you have figured out the allowed states, you should also understand the amount of memory you need to store the states.
4. Work out the truth table for the FSM by specifying the "Current State", "Input" and "Next State".
5. Then use the general principle of how a sequential logic system works to design your FSM.

Each of the above steps is necessary to get full marks for this assignment.

Exercise: Fun in an elevator – FSM controller for an elevator

Design a controller for a simplified elevator that goes between just two floors ground and first. Go out into the hallway and observe the operation of the building elevator carefully to determine what control inputs and outputs you need.

Note that you can use the following simplifications for a two-floor elevator:

a) The indicator button outside is merely an output of the FSM – it is not a control input
b) The control inputs are provided by the user *inside* the elevator. He/She also has a copy of the output of the FSM to indicate the present floor location.

At the end of the design process, you can implement your controller using the usually available IC's we've been working with and demonstrate its operation with momentary switches and LED's. Use a Red LED to indicate the elevator is on the Ground Floor and a Green one for First floor.

1) FSM description in words – how it works

NAME + ROLL #

2) FSM State Diagram

3) FSM Truth Table

4) Implementation:

A) Comment on how would you expand your design to control multiple floors? How many states are required for a 3 floor elevator? 7 floors? 14 floors?

B) Note that such a state machine does not scale up smoothly if the building has ~ 25 floors and four separate elevators – the waiting time is too long. A sophisticated microprocessor based system is used. A waiting person enters the desired destination floor on a keypad and a control algorithm optimizes which of the elevators is nearest to him/her and get them to the destination floor fastest.

/1	/1	/2	/3	/7
(Part 1) Description	(Part 2) State Diag	(Part 3) Truth Table	(Part 4) Lmplementation	+Exapansion to multiple floors

Reference:
Boolean Algebra Rules to remember:

In the following, the symbols:
(•) represents an AND operation
(+) represents an OR operation
(′) represents the NOT operation

$$
\begin{aligned}
&(1a) && x \cdot y &&= && y \cdot x \\
&(1b) && x + y &&= && y + x \\
&(2a) && x \cdot (y \cdot z) &&= && (x \cdot y) \cdot z \\
&(2b) && x + (y + z) &&= && (x + y) + z \\
&(3a) && x \cdot (y + z) &&= && (x \cdot y) + (x \cdot z) \\
&(3b) && x + (y \cdot z) &&= && (x + y) \cdot (x + z) \\
&(4a) && x \cdot x &&= && x \\
&(4b) && x + x &&= && x \\
&(5a) && x \cdot (x + y) &&= && x \\
&(5b) && x + (x \cdot y) &&= && x \\
&(6a) && x \cdot x' &&= && 0 \\
&(6b) && x + x' &&= && 1 \\
&(7) && (x')' &&= && x \\
&(8a) && (x \cdot y)' &&= && x' + y' \\
&(8b) && (x + y)' &&= && x' \cdot y'
\end{aligned}
$$

Truth tables for the basic gate operations are:

Input 1	Input 2	Output
0	0	0
0	1	0
1	0	0
1	1	1

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	1

Input = 1 Output = 0
Input = 0 Output = 1

XOR (different = high, same = low)

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

NAND = (AND)′
NOR = (OR)′
XNOR = (XOR)′