**B38DF** Computer Architecture and Embedded Systems

**Total Marks (100)**

Lab 2 – Verilog Programming using Quartus II

**Part 1: Introduction**

In this lab, we are going to look at how to design and implement different types of combinational and sequential circuits using Verilog language. You will be introduced to design concepts such as design entry, compilation, and simulation. This lab consists of five problems. You will need to do the Verilog programming, compile, and simulate your code using Quartus II software.

**Part 2: Design of Digital Circuits Using Verilog**

For each of the problem, you will have to produce the Verilog code, and the simulation waveform to demonstrate that your Verilog code works correctly.

**Problem 1**

Write a Verilog HDL module called minority. It receives three inputs, **a**, **b**, and **c**. It produces one output, **y**, that is TRUE if at least two of the inputs are FALSE. Use Quartus II functional simulation for simulating your circuit and demonstrate its correct working. **[15 marks]**

**Problem 2**

Given an 8-bit bus, the task is to use Verilog to build an encoder that can identify the location of the first '1' in the bus, starting from the most significant bit (MSB). For example, if the input is [0 0 1 0 1 1], then the logic should return 3. (Hint: you can use a for loop, but probably a simpler solution can be achieved with casez.) **[15 marks]**

**Problem 3**

A parity check is a core component of the Hamming error-correcting codes. An even/odd parity checker can be implemented as an FSM that receives bit sequence as input and generates a 1 if the number of 1's received so far is even, or 0 otherwise. For example, if the input stream is 0110100101...., then the corresponding output stream of an even parity checker is 1011000110.... Based on the parity checker FSM shown in Figure Q3, implement the parity checker in Verilog. **[20 marks]**

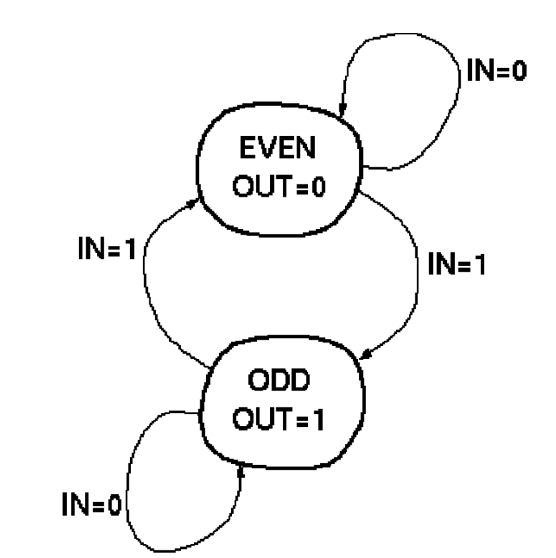
****

Figure Q3 FSM for parity checker

**Problem 4**

Based on the state diagram provided in Figure Q4a, write a Verilog code for a 111 Sequence Detector that outputs 1 when a sequence of three consecutive 1’s is applied to input, and 0 otherwise. As example input and output is given in Figure Q4b.

**[25 marks]**

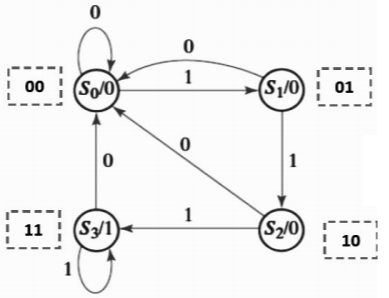


Figure Q4a State diagram

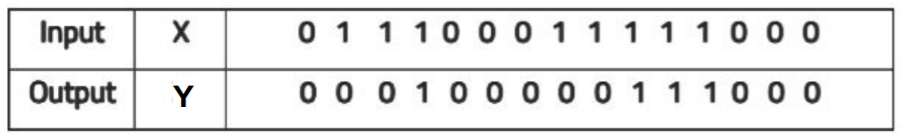


Figure Q4b Example input/output

**Problem 5**

Figure Q5a shows an electronic combination lock with a reset button, two number buttons (0 and 1), and an unlock output. The combination of 01011 will unlock the lock. Based on the state diagram shown in Figure Q5b, write a Verilog code to implement the design. **[25 marks]**

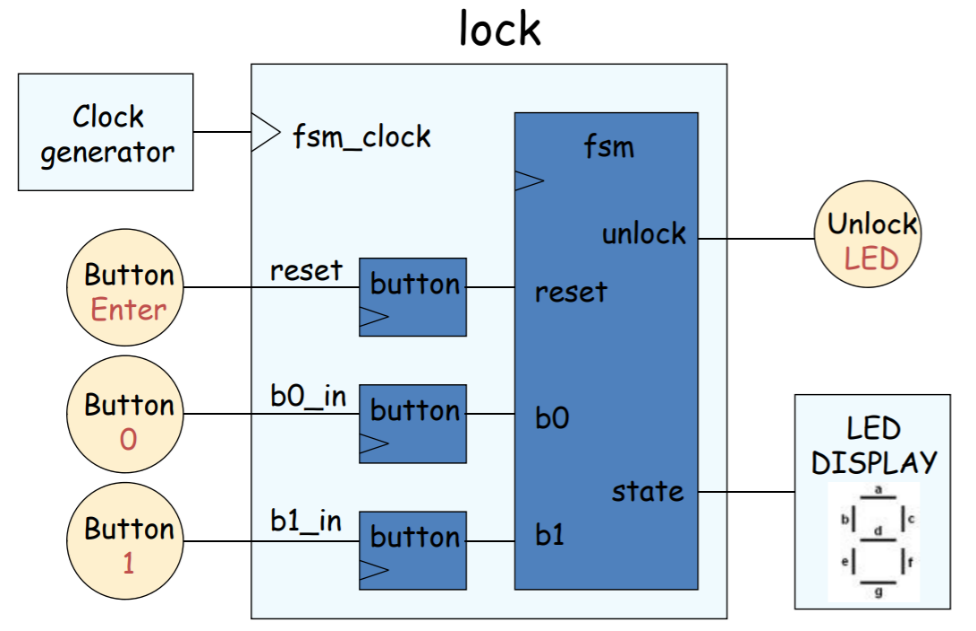


Figure Q5a Electronic combination lock

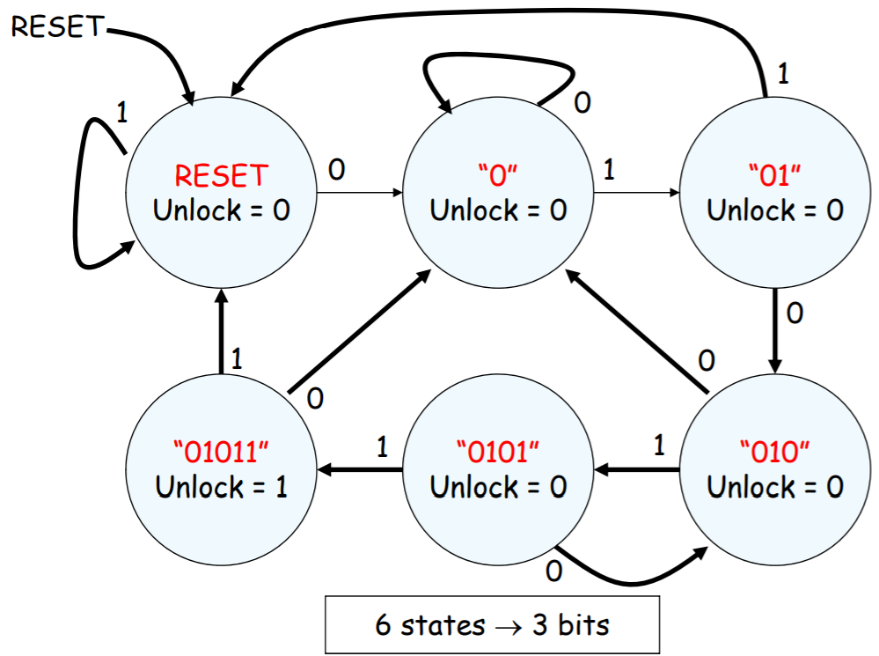


Figure Q5b State diagram for the electronic combination lock