1. Construct a finite state machine for a sequence detector that detects the sequence 1010. The FSM should permit overlapping too. For example if the input sequence is 01101010, the corresponding output sequence is 00000101.

a) Construct the state diagram, Transition and output table, excitation table.

b) Then build the K-map for the same and design the circuitry or logic diagram.

c) Write the Verilog code to implement the sequence detector.

d) Add a test bench to test the sequence detector given fifteen different inputs. Put five-time unit delay between consecutive inputs.

2. Construct a finite state machine for the 3-bit odd parity bit generator. A serial parity-bit generator is a two-terminal circuit that receives coded messages and adds a parity bit to every m bits of the message so that the resulting outcome is an error detecting code. The inputs are assumed to arrive in strings of three symbols(m=3) and the string is spaced apart by single time units (i.e., the fourth place is blank). The parity bits are inserted in the appropriate spaces so that the resulting outcome is a continuous string of symbols without spaces. For even parity, a parity bit 1 is inserted, if and only if the numbers of 1s in the preceding string of three symbols are odd. For odd parity, a parity bit 1 is inserted, if and only if the numbers of 1s in the preceding string of three symbols is even. Here we will focusing on designing only odd parity generator.

a) Construct the state diagram, state table, transition and output table, excitation table.

b) Then build the K-map for the same and design the circuitry or logic diagram.

c) Then write the Verilog code module to implement the 3-bit odd parity bit generator.

d) Now, add a test bench to test the 3-bit odd parity bit generator given 8 different inputs. Put five-time unit delay between consecutive inputs.

The students have to submit a pdf file containing the details of the FSM constructions and the verilog code.