Lab 1

Exercise:

1. Write Verilog code to describe the following functions

   f1 =ac'+bc+b'c'

   f2 = (a+b'+c)(a+b+c')(a'+b+c')

   Check whether f1 and f2 in question 1 are functionally equivalent or not.
2. Simplify the following functions using K-map and implement the circuit using logic gates. Write Verilog code and simulate the circuit
   a) f(A,B,C,D) = ∑m(1,3,4,9,10,12) + D(0,2,5,11)
   b) f(A,B,C,D) = ∏M(6,9,10,11,12) + D(2,4,7,13)
3. Minimize the following expression using K-map and simulate using only NAND gates.

   f(A,B,C,D)= πM(2,6,8,9,10,11,14)

Additional exercise:

1. Minimize the following expressions using K-map and simulate using only NOR gates.

   f(A,B,C,D)= ∑m(0,1,2,5,8,9,10)

Lab 2

Exercise:

Write behavioral Verilog code to implement the following and simulate

1. Full adder
2. Four-bit adder/ subtractor
3. Single-digit BCD adder using a four-bit adder(s).

Additional exercise:

Write behavioral Verilog code to implement the following and simulate

1. 2-bit multiplier

Lab 3

Exercise:

1. Using **for** loop, write behavioral Verilog code to convert an N bit grey code into equivalent binary code.
2. Write and simulate the Verilog code for a 4-bit comparator using 2-bit comparators.
3. Write behavioral Verilog code for
   - an 8 to 1 multiplexer using **case** statement
   - a 2 to 1 multiplexer using the **if-else** statement.

Using the above modules write the hierarchical code for a 16 to 1 multiplexer.

Additional exercise:

1. Implement F(a,b,c,d) = a’b + ac’ + abd’ + bc’d using 8 to 1 multiplexer and write the Verilog code for the same.

Lab 4

Exercise:

1. Write and simulate the Verilog code for a BCD to Excess 3 code converter using 8 to 1 multiplexers and other necessary gates.
2. Write behavioral Verilog code for a 2 to 4 decoder with active low enable input and active high output using **case** statement. Using this, design a 4 to 16 decoder with active low enable input and active high output and write the Verilog code for the same.
3. Write behavioral Verilog code for 16 to 4 priority encoder using **for** loop.

Additional exercise:

1. Write behavioral Verilog code for a 3 to 8 decoder with active-high enable input and active high output using **for** loop. Using the 3 to 8 decoders above, design a 4 to 16 decoder and write the Verilog code for the same.

Lab 5

Exercise:

1. Design and simulate a combinational circuit with external gates and a 4 to 16 decoder built using a decoder tree of 2 to 4 decoders to implement the functions below.
   F= ab’c + a’cd + bcd’ , G=acd’ + a’b’c and H=a’b’c’ + abc + a’cd
2. Design and implement a full adder using 2 to 4 decoder(s) and other gates.
3. Design and simulate the circuit with 3 to 8 decoder(s) and external gates to implement the functions below.
   F(a, b, c, d)= Σm(2,4,7,9) G (a, b, c, d)= Σm (0,3,15) H(a, b, c, d)= Σm(0,2,10,12)

Additional exercise:

1. Design and implement an 8 to 1 multiplexer using 3 to 8 decoder and external gates.

Lab 6

Exercise:

1. Write behavioral Verilog code for a negative edge triggered T FF with asynchronous active low reset.
2. Write behavioral Verilog code for a positive edge-triggered JK FF with synchronous active high reset
3. Design and simulate the following counters
   a) 4-bit ring counter.
   b) 5 bit Johnson counter.

Lab 7

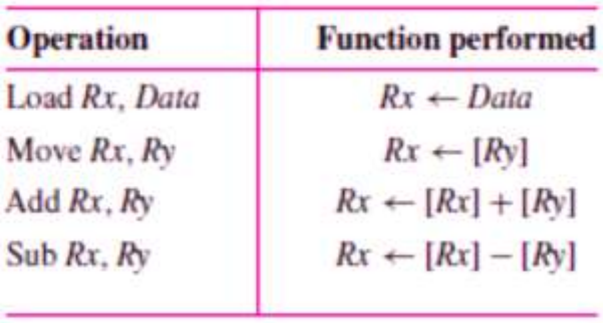
Exercise:

1. Design and simulate the following counters
   a) 4 bit synchronous up counter
   b) 3 bit synchronous up/down counter with a control input up/$\overline{down}$. If up/$\overline{down}$ = 1, then the circuit should behave as an up counter. If up/$\overline{down}$ = 0, then the circuit should behave as a down counter.

Lab 8

Exercise:

1. Write and simulate the Verilog code to swap the contents of two registers using multiplexers.
2. Simulate a simple processor that can perform the following functions:



Operation	**Function performed**
Load *Rx*, *Data*	$Rx \leftarrow Data$
Move *Rx*, *Ry*	$Rx \leftarrow [Ry]$
Add *Rx*, *Ry*	$Rx \leftarrow [Rx] + [Ry]$
Sub *Rx*, *Ry*	$Rx \leftarrow [Rx] - [Ry]$