Tasks

1. Generate a 2:1 mux using verilog code. Verify its operation.  
   Implement a T flip flop using 2:1 mux as a sub module.
2. Design a D-flip flop with asynchronous clear using verilog code. Verify its operation.  
   Using this as submodule create a D flip flop with asynchronous set using minimal hardware.
3. Design FSM for a pattern matching block : Output is asserted 1 if pattern “101” is detected in last 4 inputs.

Eg : I/P 0 1 0 1 0 0 1 1 0 1 0 1 0

O/P 0 0 0 1 1 0 0 0 0 1 1 1 1

1. Write a Verilog code to swap 2 numbers using:
2. Blocking Assignment
3. Non-Blocking Assignment

Try to do this with and without a temporary variable.

* Build a 4-bit shift register (right shift), with asynchronous reset, synchronous load, and enable.

1. areset: Resets shift register to zero.
2. load: Loads shift register with data[3:0] instead of shifting.
3. ena: Shift right (q[3] becomes zero, q[0] is shifted out and disappears).
4. q: The contents of the shift register.

If both the load and ena inputs are asserted (1), the load input has higher priority.

* You have two 8-bit 2's complement numbers, a[7:0] and b[7:0]. These numbers are added to produce s[7:0]. Also compute whether a (signed) overflow has occurred.*(A signed overflow occurs when adding two positive numbers produces a negative result, or adding two negative numbers produces a positive result. )*