1. Design the following in Verilog:
   A circuit that can be used as an inverter/buffer depending on input signal INV. if INV=1, the circuit acts as an inverter. If INV=0, the circuit acts as Buffer.

```
module inv_buffer(
   input wire A,   // Input signal
   input wire INV, // Control signal: 1 for inverter, 0 for buffer
   output wire Y   // Output signal
);

   assign Y = (INV) ? ~A : A;

endmodule
```

2. Design a module 4bit_adder_subtractor_dut which has a function 4b_add_sub which takes an input signal add if add=1 the function performs addition of two 4-bit inputs ‘A’ and ‘B’, else it performs subtraction of ‘A’ and ‘B’.

```
module four_bit_adder_subtractor_dut(
   input [3:0] A, B,
   input add,
   output [3:0] result,
   output carry_out
);
   reg [3:0] B_complement;
   reg carry_in;
   wire [4:0] sum;

   always @(*) begin
      if (add) begin
         B_complement = B;  // Perform addition
         carry_in = 1'b0;
      end else begin
         B_complement = ~B;  // Perform subtraction using 2's complement
         carry_in = 1'b1;
      end
   end

   assign sum = A + B_complement + carry_in;
   assign result = sum[3:0];
   assign carry_out = sum[4];  // Carry out
endmodule
```

3. Design following using Verilog
   a. 3-bit Johnson Counter

```
module johnson_counter_3bit(
   input clk,
   input reset,
   output reg [2:0] Q
);

   always @(posedge clk or posedge reset) begin
```

```
        if (reset)
            Q <= 3'b000;
        else
            Q <= {~Q[2], Q[2:1]};
    end
endmodule
```

b. SR latch

```
module sr_latch(
    input S, R,
    output Q, Q_bar
);
    nor(Q, R, Q_bar);
    nor(Q_bar, S, Q);
endmodule
```

4. Design an FSM in Verilog with below states:
   a. IDLE - This is the default state on reset.
      IDLE state can transition to 2 states GRNT0 and GRNT1:
      GRNT0 - if req0-1 GRNT1 - if req1=1
   b. GRNT0 -This state is achieved from IDLE state when req0=1
      If req0 remains '1'. The state GRNT0 remains unchanged.
      If req0=0, then GRNT) transitions to IDL state.
   c. GRNT1 - This state is achieved from IDLE state when req1=1
      If req1 remains '1'. The state GRNT0 remains unchanged.
      If req1=0, then GRNT) transitions to IDL state.

```
module fsm_grant(
    input clk,
    input reset,
    input req0,
    input req1,
    output reg grant0,
    output reg grant1
);

    typedef enum reg [1:0] {IDLE, GRNT0, GRNT1} state_t;
    state_t state, next_state;

    always @(posedge clk or posedge reset) begin
        if (reset)
            state <= IDLE;
        else
            state <= next_state;
    end

    always @(*) begin
        case (state)
            IDLE: begin
                if (req0)
                    next_state = GRNT0;
```

```
            else if (req1)
               next_state = GRNT1;
            else
               next_state = IDLE;
         end
         GRNT0: begin
            if (req0)
               next_state = GRNT0;
            else
               next_state = IDLE;
         end
         GRNT1: begin
            if (req1)
               next_state = GRNT1;
            else
               next_state = IDLE;
         end
         default: next_state = IDLE;
      endcase
   end

   always @(state) begin
      case (state)
         IDLE: begin
            grant0 = 0;
            grant1 = 0;
         end
         GRNT0: begin
            grant0 = 1;
            grant1 = 0;
         end
         GRNT1: begin
            grant0 = 0;
            grant1 = 1;
         end
      endcase
   end
endmodule
```