Quiz 1

Name: ________________
GTID: ________________

1. Design an ALU that performs the following operations. Please show the block diagram of the ALU first and then implement the ALU with Verilog. A and B are 8-bits 2’s complement signed values:

A + B
A - B
-A
!A
A & B
A | B
A ^ B
A >> 1

Answer:

module Alu (sel, A, B, Cout, Z);

input [2:0] sel;
input signed [7:0] A, B;
output reg Cout;
output reg [7:0] Z;

always @(sel, A, B)
begin
    case(sel)
    3'b000: {Cout, Z} <= A + B;
    3'b001: {Cout, Z} <= A - B;
    3'b010: Z <= ~A + 1;
    3'b011: Z <= ~A;
    3'b100: Z <= A & B;
    3'b101: Z <= A | B;
    3'b110: Z <= A ^ B;
    3'b111: Z <= A >> 1;
    endcase
endmodule
2. Please design a circuit that detects the 1011 sequence. Show the state machine and implement the sequence detector in Verilog.

**Answer:**

```verilog
module detector(clk, rst, x, z);
    input clk;
    input rst;
    input x;
    output z;
    reg [2:0] state;

    parameter start = 3'b000;
    parameter got1 = 3'b001;
    parameter got10 = 3'b010;
    parameter got101 = 3'b011;
    parameter got1011 = 3'b100;

    always @(posedge clk)
        begin
            if(rst)
                begin
                    state <= start;
                end
            else
                begin
                    case (state)
                        start: begin
                            if(x) state <= got1;
                            else state <= start;
                        end
                        got1: begin
                            if(x) state <= got1;
                            else state <= got10;
                        end
                        got10: begin
                            if(x) state <= got101;
                            else state <= start;
                        end
                        got101: begin
                            if(x) state <= got1011;
                            else state <= got10;
                        end
                        got1011: begin
                            if(x) state <= got1;
                            else state <= got10;
                        end
                    endcase
                end
            assign z = (state == got1011);
        end
endmodule
```
3. Write Verilog code to implement an SRAM memory containing 512 entries. Each entry is a 32-bit value. The reads and writes are synchronized with the positive edge of the clock. Your design should include the following inputs and outputs:

- Data_in
- Data_out
- Address
- Rw_enable
- Clk

**Answer:**

```verilog
module sram(Clk, Rw_enable, Address, Data_in, Data_out);

  parameter BIT_WIDTH = 32;
  parameter ADDR_SIZE = 9;

  input Clk;
  input Rw_enable;
  input [ADDR_SIZE-1:0] Address;
  input [BIT_WIDTH-1:0] Data_in;
  output reg [BIT_WIDTH-1:0] Data_out;

  reg [BIT_WIDTH-1:0] mem [0:(1<<ADDR_SIZE)-1];

  always @(posedge Clk)
  begin
    if(Rw_enable) // write
      begin
        mem[Address] <= Data_in;
      end
    else // read
      begin
        Data_out <= mem[Address];
      end
  end
endmodule
```