Rubric

1. You need to implement a `MUX` module and a `decoder` module. Each module worth 15 points. (30 in total) If you do so, at least 5 points will be reserved for each module.

   If you do not define the two modules but have same function inside you main module, no points will be deducted for each module. But the deduction upper bound is 15 points rather than 10 points.

2. **Correct** call of module `MUX` in your main module worth 2 points. So does `decoder`.

   Correct call of module `buffer` worth 5 points. So does `register`.

   For those who didn't create `MUX` and `decoder` module, the corresponding 4 points will be decided by your function of the main module.

3. Minor mistake leads to 2 points deduction. Major mistake leads to 5 points deduction.

4. The maximum deduction in main module is 20 points.

Important Points

1. If you use `case` or `if` to build `MUX` and `decoder`, you need to consider the default condition. `default` should be included in `case` and your `if` branches should be ended with `else` rather than `else if`.
2. `if` and `else` can only be used **inside** `always` blocks. Modules can only be called **outside** `always` blocks. `reg` variables can only be assigned a value **inside** `always` blocks. `wire` variables can only be connected **outside** `always` blocks.
3. You cannot connect a `reg` variable to the **output** of a module. The output of a module can only be connected to a `wire` variable.
4. To build a `MUX`, in the always block, you should include all conditions. See sample code.
5. You need to care about the bit-width. It's a 32-bit circuit. As a result, when you call buffers and registers, you need the parameter `#(32)`. Besides, when you define a variable, you also need to include bit-width, e.g. `reg [31:0] x`.

Sample Code

The sample code is on the next page.

```verilog
module Decoder_2_4 (I, D);
    input [1:0] I;
    output [3:0] D;
    reg [3:0] D;

    always @ (I) begin
        case (I)
            2'b00: D=4'b0001;
            2'b01: D=4'b0010;
            2'b10: D=4'b0100;
            2'b11: D=4'b1000;
            default D=0;
        endcase
    end
endmodule

module MUX_4_1 (F, I3, I2, I1, I0, sel);
    input [31:0] I3, I2, I1, I0;
    input [1:0] sel;
    output [31:0] F;
    reg [31:0] F;

    always @ (I3, I2, I1, I0, sel) begin // You must include all conditions.
        case (sel)
            2'b00: F=I0;
            2'b01: F=I1;
            2'b10: F=I2;
            2'b11: F=I3;
            default F=0;
        endcase
    end
endmodule

module Interesting_quiz (Data, A, clock, sel, F);
    input [31:0] Data;
    input [1:0] A, sel;
    input clock;
    output [31:0] F;

    wire [31:0] D;
    Decoder_2_4 Decoder (A, D);

    wire [31:0] C3, C2, C1, C0;
    Tri_Buff #(32) Buffer0 (C0, D[0], Data);
    Tri_Buff #(32) Buffer1 (C1, D[1], Data);
    Tri_Buff #(32) Buffer2 (C2, D[2], Data);
    Tri_Buff #(32) Buffer3 (C3, D[3], Data);

    wire [31:0] Q3, Q2, Q1, Q0;
    Reg_N_bits #(32) Register0 (Q0, C0, clock);
    Reg_N_bits #(32) Register1 (Q1, C1, clock);
    Reg_N_bits #(32) Register2 (Q2, C2, clock);
    Reg_N_bits #(32) Register3 (Q3, C3, clock);

    MUX_4_1 MUX (F, Q3, Q2, Q1, Q0, sel);
endmodule
```