**Chapter Seven: Verilog**

**Chapter Eight: Memory**

1. **True or False.** Given a ROM with address input A0~A7 and data output D0~D3, how many word lines it has? ( **16** )

2. A Verilog module for the SHRG4U 4-bit **shift register** and its function table are shown as below. There are several errors in the Verilog module. Please find the errors and correct them.

| Function table for the SHRG4U 4-bit shift register | | | | | | | |
|---|---|---|---|---|---|---|---|
| Inputs | | | Next state | | | | Function |
| /CLR | S1 | S0 | Q3* | Q2* | Q1* | Q0* | |
| 0 | X | X | 0 | 0 | 0 | 0 | Asynchronous clear |
| 1 | 0 | 0 | Q3 | Q2 | Q1 | Q0 | Hold |
| 1 | 0 | 1 | Q2 | Q1 | Q0 | RIN | Shift right |
| 1 | 1 | 0 | LIN | Q3 | Q2 | Q1 | Shift left |
| 1 | 1 | 1 | D3 | D2 | D1 | D0 | Synchronous load |

```
module SHRG4U( CLK, CLR_L, RIN, LIN, S, D, Q );          //1
    input CLK, CLR_L, RIN, LIN;                          //2
    input S;                                             //3
    input [3:0]D;                                        //4
    output reg [3:0]Q;                                   //5
                                                         //6
    always @ ( posedge CLK ) begin                       //7
        if( CLR_L ) Q <= 0;                              //8
        else case(S)                                     //9
            0: Q <= Q;                                   //10
            1: Q <= {RIN,Q[3:1]};                        //11
            2: Q <= {Q[2:0],LIN};                        //12
            3: Q <= D;                                   //13
        endcase                                          //14
    end                                                  //15
endmodule                                                //16
```

3. **True or False.** To implement a 1024*8 bit ROM, we need（ 16 ）256*1 bit ROM chips.

4. Programmable Logic Design and Verilog Design.

The following Truth table can be used to implement a circuit that convert a 5421 BCD to Excess-3 Code.

(1) Use Karnaugh map to simplify the output functions, write the minimal sum-of-products expression.

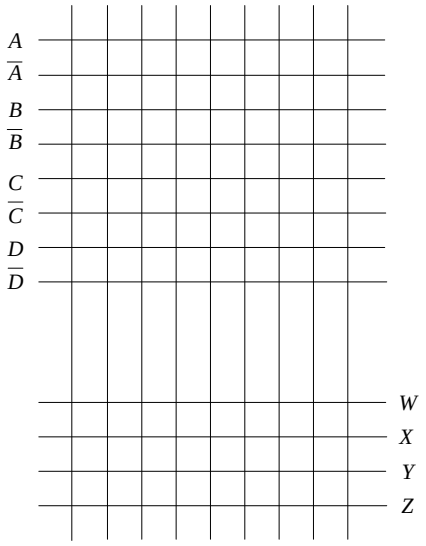
(2) Implement the Truth table by Programmable Logic Device. Mark the '●' to program the Programmable Logic Device.

(3) Design a Verilog module by Dataflow style or Gate Structural style that can implement the same function.

Inputs Outputs

| A | B | C | D | W | X | Y | Z |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 0 | 0 |

Truth Table




Programmable Logic Device