

# 同济大学计算机系

## 数字逻辑课程实验报告



学 号 2353900

姓 名 汪嘉晨

专 业 计算机科学与技术（精英班）

授课老师 郭玉臣

# 一、实验内容

在本次实验中，我们将深入了解桶形移位器的原理，使用 Verilog HDL 语言实现 32 位桶形移位器的设计和仿真。

桶式移位器是一种组合逻辑电路，通常作为微处理器 CPU 的一部分。它具有 n 个数据输入和 n 个数据输出，以及指定如何移动数据的控制输入，指定移位方向、移位类型（循环、算术还是逻辑移位）及移动的位数等等。

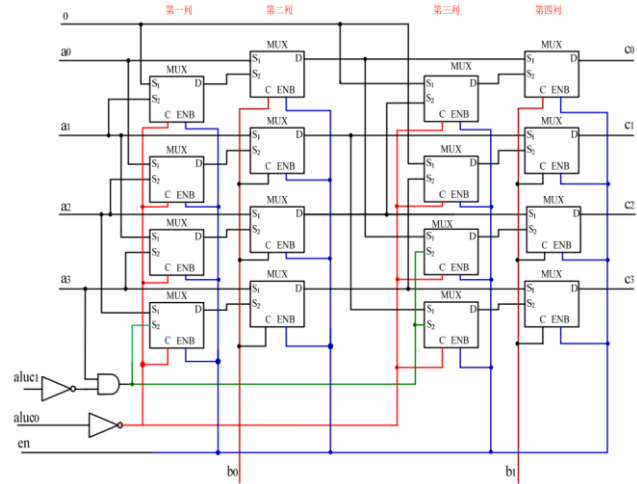
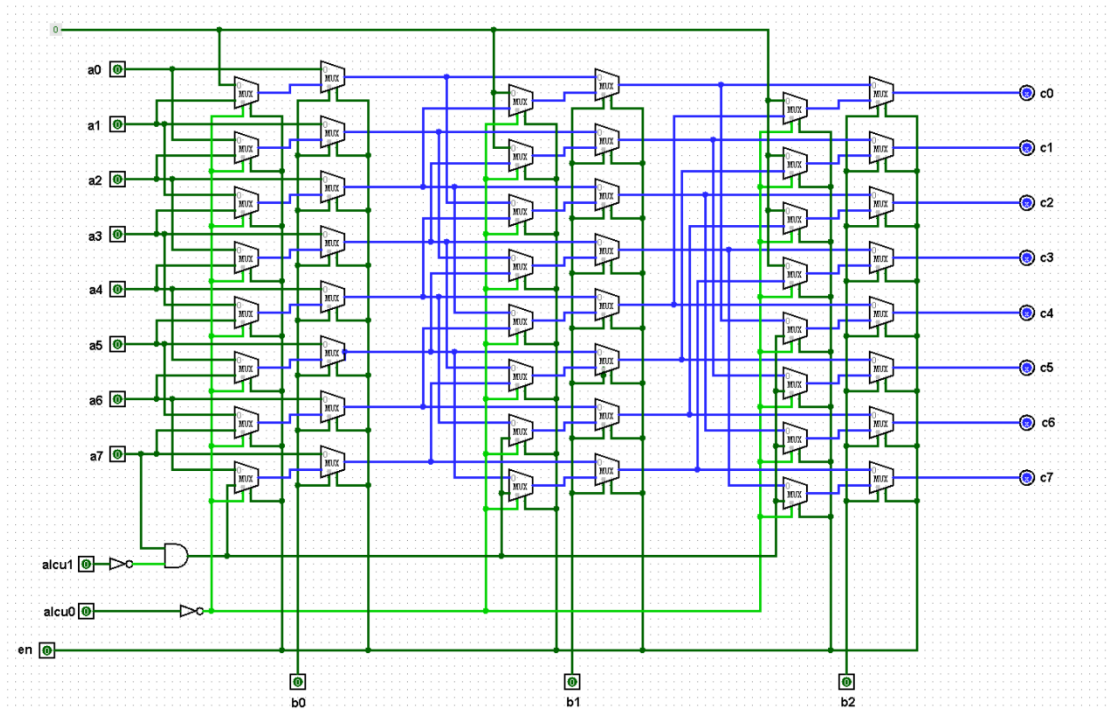


图 6.4.1.1 4 位桶形移位器原理图

# 二、硬件逻辑图

下图为 8 位桶形移位器的原理图，算术右移、算数左移、逻辑右移、逻辑左移均符合要求：



# 三、模块建模

```

module barrelshifter32(
    input [31:0] a,          // 32 位原始输入数据
    input [4:0] b,           // 5 位输入信号，控制移位的位数
    input [1:0] aluc,        // 2 位输入信号，控制移位的方式
    output reg [31:0] c      // 32 位移位后的输出数据
);

always @(*) begin
    case (aluc)
        2'b00: begin // 算术右移 (sra)
            c = a;
            if (b[0]) c = $signed(c) >>> 1;
            if (b[1]) c = $signed(c) >>> 2;
            if (b[2]) c = $signed(c) >>> 4;
            if (b[3]) c = $signed(c) >>> 8;
            if (b[4]) c = $signed(c) >>> 16;
        end
        2'b10: begin // 逻辑右移 (srl)
            c = a;
            if (b[0]) c = c >> 1;
            if (b[1]) c = c >> 2;
            if (b[2]) c = c >> 4;
            if (b[3]) c = c >> 8;
            if (b[4]) c = c >> 16;
        end
        2'b01, // 算术左移 (sll)
        2'b11: begin // 逻辑左移 (sll)
            c = a;
            if (b[0]) c = c << 1;
            if (b[1]) c = c << 2;
            if (b[2]) c = c << 4;
            if (b[3]) c = c << 8;
            if (b[4]) c = c << 16;
        end
        default: c = 32'b0; // 默认情况下输出 0
    endcase
end

endmodule

```

## 四、测试模块建模

```

module tb_barrelshifter32;

    reg [31:0] a;          // 32 位原始输入数据

```

```

reg [4:0] b;          // 5 位输入信号，控制移位的位数
reg [1:0] aluc;       // 2 位输入信号，控制移位的方式
wire [31:0] c;        // 32 位移位后的输出数据

// 实例化桶形移位器
barrelshifter32 uut (
    .a(a),
    .b(b),
    .aluc(aluc),
    .c(c)
);

initial begin
    // 初始化信号
    a = 32'hF0F0F0F0; // 输入数据
    b = 5'b00001;      // 移位 1 位
    aluc = 2'b00;       // 算术右移
    #10; // 等待 10 个时间单位
    $display("ALUC=00, A=%h, B=%b, C=%h", a, b, c);

    b = 5'b00010;      // 移位 2 位
    aluc = 2'b10;       // 逻辑右移
    #10;
    $display("ALUC=10, A=%h, B=%b, C=%h", a, b, c);

    b = 5'b00100;      // 移位 4 位
    aluc = 2'b01;       // 算术左移
    #10;
    $display("ALUC=01, A=%h, B=%b, C=%h", a, b, c);

    b = 5'b01000;      // 移位 8 位
    aluc = 2'b11;       // 逻辑左移
    #10;
    $display("ALUC=11, A=%h, B=%b, C=%h", a, b, c);

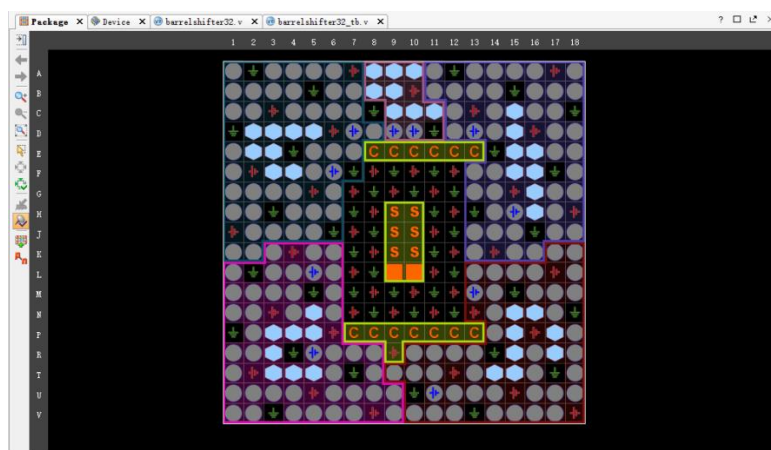
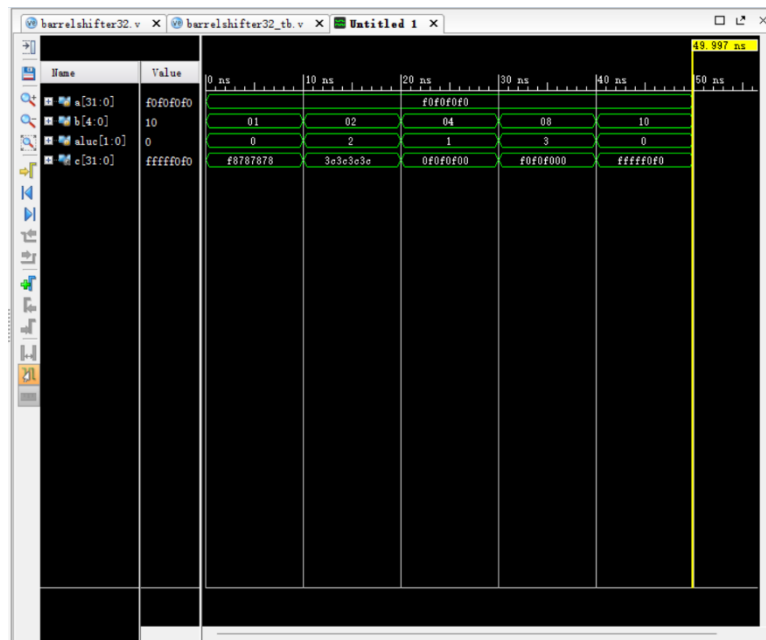
    b = 5'b10000;      // 移位 16 位
    aluc = 2'b00;       // 算术右移
    #10;
    $display("ALUC=00, A=%h, B=%b, C=%h", a, b, c);

    // 结束仿真
    $finish;
end

```

endmodule

## 五、实验结果



Logisim 验证图（以逻辑左移为例）：

