同济大学
计算机科学与技术系

计算机组成原理实验报告

学　　号　1652228

姓　　名　王哲源

专　　业　计算机科学与技术

授课老师　陈永生

日　　期　2018.3.30

一、实验目标

通过 verilog 语言实现 32 位无符号除法器和 32 位带符号除法器

二、模块建模

1. 32 位无符号除法器

   被除数为 dividend，除数为 divisor，均为 32 位无符号数，将计算结果存在 q 中，余数存在 r 中

```
module DIVU(
        input [31:0]dividend,
        input [31:0]divisor,
        input start,
        input clock,
        input reset,
        output [31:0]q,
        output [31:0]r,
        output reg busy
        );

        reg [4:0]count;

        reg [31:0]tmp_q, tmp_r, tmp_d;
        reg r_sign;
        assign q = tmp_q;
        assign r = r_sign == 1 ? tmp_r + tmp_d : tmp_r;

        wire [32:0]tmp_sub;
        assign tmp_sub = r_sign ? {{tmp_r, q[31]} + {1'b0, tmp_d}} : {{tmp_r, q[31]} - {1'b0, tmp_d }};

        reg pre_start;
        always @(posedge clock or posedge reset or posedge start or negedge start)
        begin
                pre_start <= start;
                if(reset == 1)
                begin
                        busy <= 0;
                        count <= 0;
                end
                else if({pre_start, start} == 2'b10)
                begin
                        tmp_d <= divisor;
                        tmp_q <= dividend;
                        tmp_r <= 32'b0;
                        r_sign <= 0;
                        busy <= 1;
                        count <= 0;
                end
                else if({pre_start, start} == 2'b00 && busy == 1)
                begin
                        tmp_r <= tmp_sub[31:0];
                        r_sign <= tmp_sub[32];
                        tmp_q <= {tmp_q[30:0], ~tmp_sub[32]};
                        count <= count + 5'b1;
                        busy <= count == 5'd31 ? 0 : 1;
                end
        end
endmodule
```

2. 32 位带符号除法器

被除数为 dividend，除数为 divisor，均为 32 位带符号数，将计算结果存在 q 中，余数存在 r 中

```
module DIV(
    input [31:0]dividend,
    input [31:0]divisor,
    input start,
    input clock,
    input reset,
    output [31:0]q,
    output [31:0]r,
    output reg busy
    );

    reg [4:0]count;

    reg [31:0]tmp_q, tmp_r, tmp_d;
    reg r_sign, q_sign, d_sign;
    assign q = q_sign == 1 ? -tmp_q : tmp_q;
    assign r = r_sign == 1 ? (d_sign == 1 ? -(tmp_r + tmp_d) : tmp_r + tmp_d)
                                 : (d_sign == 1 ? -tmp_r : tmp_r);

    wire [32:0]tmp_sub;
    assign tmp_sub = r_sign ? {{tmp_r, tmp_q[31]} + {1'b0, tmp_d}} : {{tmp_r, tmp_q[31]} - {1'b0, tmp_d }};

    reg pre_start;
    always @(posedge clock or posedge reset or posedge start or negedge start)
    begin
        pre_start <= start;
        if(reset == 1)
        begin
            busy <= 0;
            count <= 0;
        end
        else if({pre_start, start} == 2'b10)
        begin
            tmp_d <= divisor[31] == 1 ? -divisor : divisor;
            tmp_q <= dividend[31] == 1 ? -dividend : dividend;
            tmp_r <= 32'b0;
            r_sign <= 0;
            q_sign <= divisor[31] ^ dividend[31];
            d_sign <= dividend[31];
            busy <= 1;
            count <= 0;
        end
        else if({pre_start, start} == 2'b00 && busy == 1)
        begin
            tmp_r <= tmp_sub[31:0];
            r_sign <= tmp_sub[32];
            tmp_q <= {tmp_q[30:0], ~tmp_sub[32]};
            count <= count + 5'b1;
            busy <= count == 5'd31 ? 0 : 1;
        end
    end
endmodule
```

三、测试模块设计

1. 32 位无符号除法器 testbench

```
`timescale 1ns / 1ps
module DIVU_tb;
    reg clk = 1, rst, st;
    reg [31:0] dividend, divisor;
    wire bsy;
    wire [31:0] q, r;
    DIVU uut(.clock(clk), .reset(rst), .start(st),
                .dividend(dividend), .divisor(divisor),
                .busy(bsy),
                .q(q), .r(r));

    always #5 clk = ~clk;
    initial
    begin
        rst <= 0; st <= 1;
        #3
            rst <= 0;
            dividend <= 32'd7;
            divisor <= 32'd2;
        #6 st <= 0;
        #400 st <= 1;
        #5
            st <= 0;
            dividend <= 32'hffffffff;
            divisor <= 32'h55555555;
    end
endmodule
```

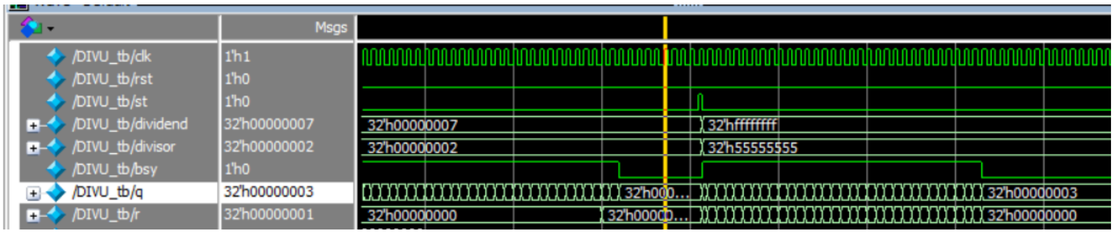
2. 32 位带符号除法器 testbench

```
`timescale 1ns / 1ps
module DIV_tb;
    reg clk = 1, rst, st;
    reg [31:0] dividend, divisor;
    wire bsy;
    wire [31:0] q, r;

    DIV uut(.clock(clk), .reset(rst), .start(st),
                .dividend(dividend), .divisor(divisor),
                .busy(bsy),
                .q(q), .r(r));

    always #5 clk = ~clk;
    initial
    begin
        rst <= 0; st <= 1;
        #3
            rst <= 0;
            dividend <= 7;
            divisor <= -2;
        #6 st <= 0;
        #400 st <= 1;
        #5
            st <= 0;
            dividend <= -7;
            divisor <= -2;
    end
endmodule
```

四、实验结果

1. 32 位无符号除法器

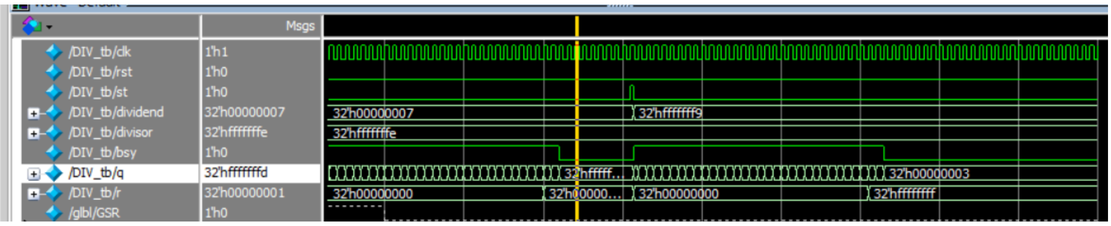
   1.1 modelsim 仿真波形图



```
                     Msgs
/DIVU_tb/clk         1'h1
/DIVU_tb/rst         1'h0
/DIVU_tb/st          1'h0
/DIVU_tb/dividend    32'h00000007
/DIVU_tb/divisor     32'h00000002
/DIVU_tb/bsy         1'h0
/DIVU_tb/q           32'h00000003
/DIVU_tb/r           32'h00000001
```

2. 32 位带符号除法器

   2.1 modelsim 仿真波形图



```
                     Msgs
/DIV_tb/clk          1'h1
/DIV_tb/rst          1'h0
/DIV_tb/st           1'h0
/DIV_tb/dividend     32'h00000007
/DIV_tb/divisor      32'hfffffffe
/DIV_tb/bsy          1'h0
/DIV_tb/q            32'hfffffffd
/DIV_tb/r            32'h00000001
/glbl/GSR            1'h0
```

五、总结

1. 32 位无符号除法实现采用不恢复余数的方法进行，而 32 位带符号除法采用不恢复余数方法的时候中间判断较为繁琐，因此采用先将被除数与除数的符号记下用于判断余数与商的符号，再取绝对值后进行计算的方式

2. 在计算带符号除法时，移位对余数进行补位的数值应当来源于取绝对值后的被除数