

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 计算机组成 |
| 姓 名： | 姜雨童 |
| 学 院： | 计算机科学与技术学院 |
| 专 业： | 计算机科学与技术 |
| 邮 箱： | 3220103450@zju.edu.cn |
| QQ 号： | 1369218489 |
| 电 话： | 19550103468 |
| 指导教师： | 马德 |
| 报告日期： | 2024年4月 7日 |

**浙江大学实验报告**

课程名称： 计算机组成 实验类型： 综合

实验项目名称： Lab 3：复杂操作实现

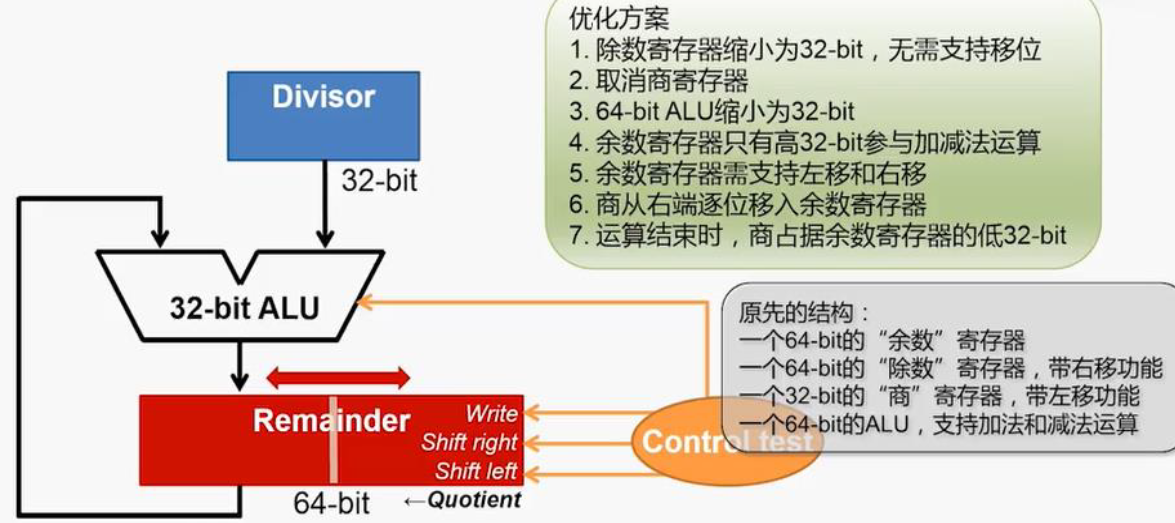
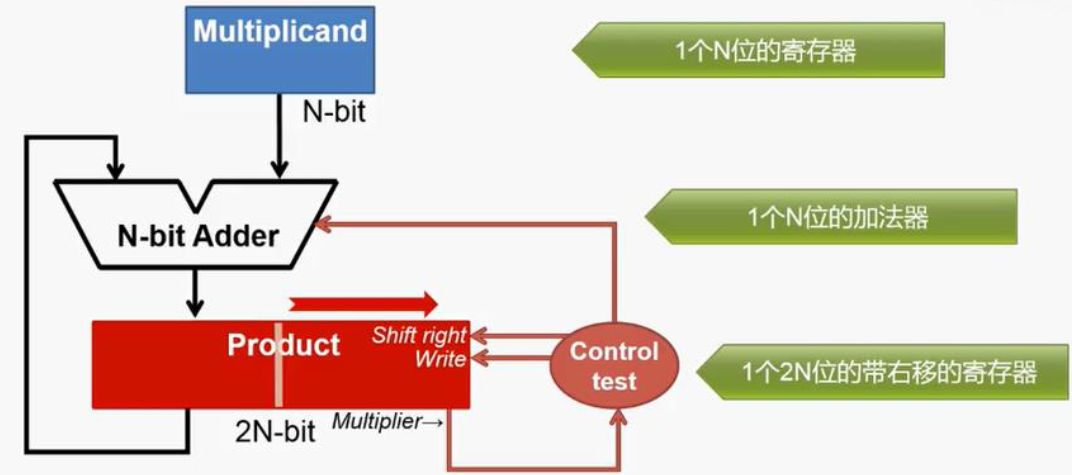
学生姓名： 姜雨童 学号： 33220103450 同组学生姓名： /

实验地点： 紫金港东四509室 实验日期： 2024 年 4 月 3 日

# 一、操作方法与实验步骤

**实验目的**：

设计实现32位乘法器和除法器，实现结构如下图所示



**实验步骤：**

Testbench文件已经给出，因此只需要完成.v文件的Verilog代码并进行仿真测试即可。

我的代码如下所示（debug部分在模块三——讨论与心得给出）：

（备注：32位乘法器模块并没有使用优化后的算法。）

module mul32(

input clk,

input rst,

input [31:0] multiplicand,

input [31:0] multiplier,

input start,

output reg [63:0] product,

output reg finish

);

reg [5:0] counter;

reg [63:0] op1;

reg [63:0] op2;

always @(posedge clk or posedge rst) begin

if (start) begin

finish <= 0;

product <= 0;

counter <= 0;

op1 <= {32'b0, multiplicand};

op2 <= {32'b0, multiplier};

end

else if (counter < 32) begin

if(op2[0] == 1)

product <= product + op1;

op1 <= {op1[62:0],1'b0};

op2 <= {1'b0, op2[63:1]};

counter <= counter + 1;

end

else begin

finish <= 1;

end

end

endmodule

module div32(

input clk,

input rst,

input start,

input [31:0] dividend,

input [31:0] divisor,

output finish,

output [31:0] quotient,

output [31:0] remainder

);

reg [63:0] op;

reg [5:0] counter;

reg done;

always @(posedge clk or posedge rst) begin

if (rst) begin

counter <= 0;

done <= 0;

op <= 0;

end

if(start) begin

done <= 0;

counter <= 0;

op <= {32'h0,dividend};

end

else if(counter < 6'd32) begin

counter <= counter + 1'b1;

op = op << 1;

if(op[63:32] >= divisor) begin

op[63:32] = op[63:32] - divisor[31:0];

op[0] <= 1;

end else

op[0] <= 0;

end

else done <= 1'b1;

end

assign finish = done;

assign quotient = op[31:0];

assign remainder = op[63:32];

endmodule

随后进行仿真测试，这部分结果在模块二——实验结果与分析中给出。

# 二、实验结果与分析

**Mul32仿真结果：**

|  |
| --- |
| 整体仿真结果如下图所示，后续给出更详细的说明。（图中结果均以16进制表示） |
|  |

|  |
| --- |
| Case 1：2 \* 3 = 6 （0x6） |
|  |
| Case 2：10 \* 8 = 80 （0x50） |
|  |

|  |
| --- |
| Case 3：9 \* 9 = 81（0x51） |
|  |

|  |
| --- |
| Case 4：50 \* 6 = 300（0x12c） |
|  |

|  |
| --- |
| Case 5：6 \* 60 = 360（0x168） |
|  |

**Div32仿真结果：**

|  |
| --- |
| 整体结果：（图上结果同样以16进制表示） |
|  |

|  |
| --- |
| Case 1：8 / 4 = 2 …… 0 |
|  |

|  |
| --- |
| Case 2：100 / 10 = 10 …… 0 |
|  |

|  |
| --- |
| Case 3： 9 / 4 = 2 …… 1 |
|  |

|  |
| --- |
| Case 4：100 / 99 = 1 …… 1 |
|  |

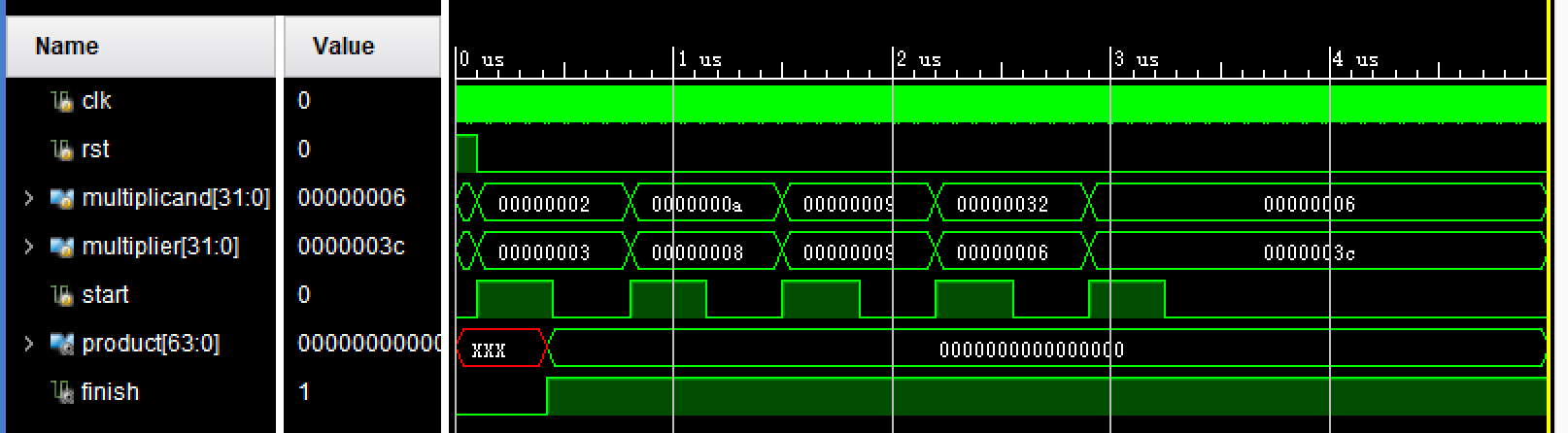
# 三、讨论、心得

**心得：**

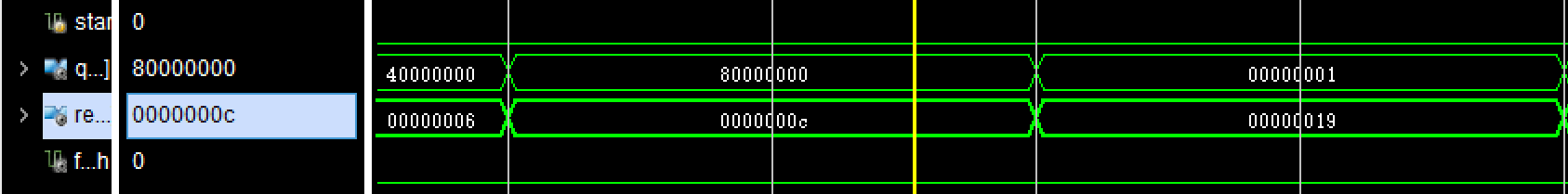
因为只涉及到了.v文件的编写，因此感觉lab3更考察对乘法器、除法器的理解，以及verilog语言的使用熟练程度。整体来说没有那么复杂，就是一些细节的地方不注意的话，会需要花一些时间debug。

**Debug：**

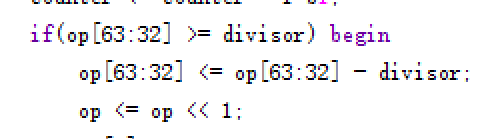
1. 最开始写乘法器的时候，忘记把临时变量定义成64位的了，还是按照32位来写，导致后续结果错误，数据对不上，修改后没有其他问题。



1. 写除法器的时候发现对优化后32位除法器的原理还不是特别清楚，因此先看了上课的PPT和智云，在明白原理后，写代码就顺畅了很多。
2. 初步实现除法器，验证结果的时候发现数据对不上，逐步检查发现是“被除数减除数”这一步操作没有执行到位（在该例中具体表现为：remainder应操作为0xC-0xA，然后左移一位，末位填充为1，结果是0x5而非0x19）.



检查相应代码后发现是赋值语句有问题，使用了“<=”并不会立刻对op进行赋值，修改为“=”后没有问题，测试样例的结果全部正确。



**四、个人生活照片**

