哈爾濱工業大學（深圳）
HARBIN INSTITUTE OF TECHNOLOGY

实验作业

开课学期：2021 春季

课程名称：计算机组成原理（实验）

实验名称：原码除法器设计

实验性质：综合设计型

实验学时：4 地点：T2615

学生班级：1901105

学生学号：190110509

学生姓名：王铭

作业成绩：

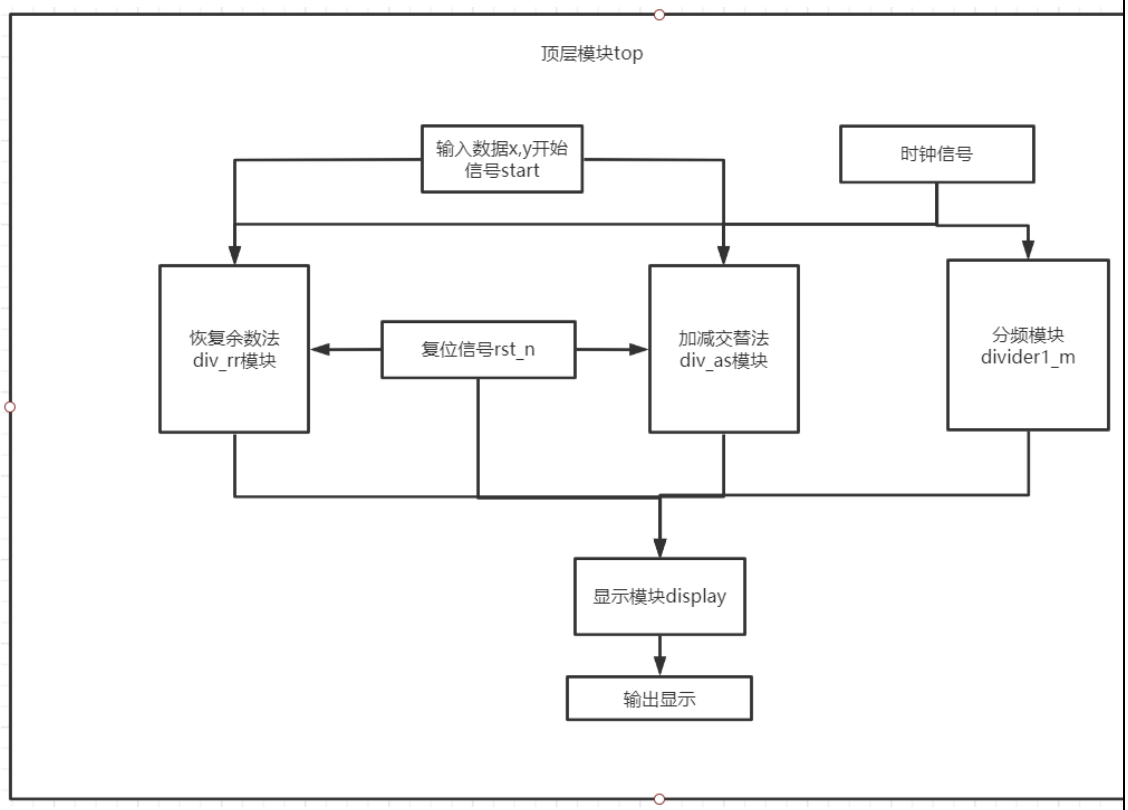
实验与创新实践教育中心制

2021 年 4 月

一、 系统功能详细设计

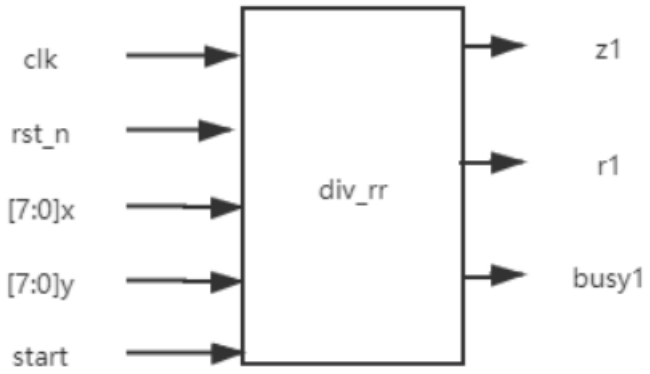
设计模块：top，divider_1m，div_rr，div_as，display

整体框架图：



主要功能：top 为顶层模块，模块内部分别调用 div_rr 和 div_as 模块计算 x 除以 y 的余数和商得到 z1,z2,r1,r2，调用 divider_1m 模块产生分频时钟，最终将产生的时钟和计算得到的结果作为参数调用 display 模块将计算的结果 z1,z2,r1,r2 显示到数码管上。

div_rr 模块：



输入：时钟信号 clk，复位信号 rst_n，被除数 x 和除数 y，开始信号 start

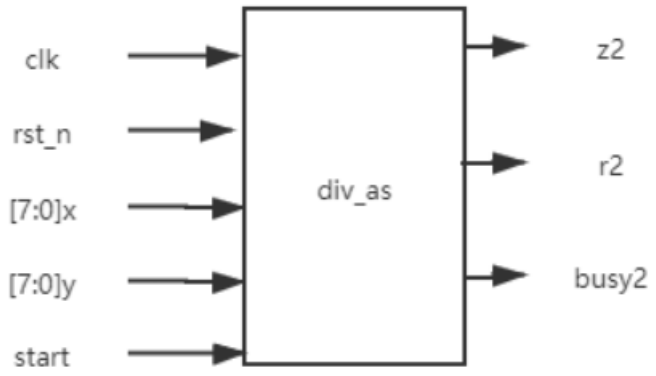
输出：加减交替法计算 x/y 的商 z1 和余数 r1.

重要变量：flag,Syn_res,div_reg

div_reg 寄存器用于保留计算过程中的余数和商，Syn_res 表示商的符号,flag 为 1 表示模块开始运算。

在时钟上升沿或复位信号有效时，先判断复位信号是否有效，有效则将 flag,div_reg,Syn_res 均置 0。否则判断 start 信号是否有效，若 start 信号有效，置 flag 为 1,div_reg 置为 x 的真值，Syn_res 置为 x,y 符号位的异或。当 flag 为 1 且 start 为 0 时，对 div_reg 左移一位后，取 div_reg 的高 8 位加 $(-|y|)_{补}$，若结果符号位为 1，则 div_reg 的前 8 位加 $|y|_{补}$,div_reg[0]置 0；若符号位为 0，div_reg 置 1。此过程由 cnt 控制完成 7 次。cnt 为 0 时，判断余数的符号位 div_reg 的最高位是否为 1 决定是否恢复余数，将 Syn_res 与 div_reg 的[13:7]位组合形成 r1，将 x 的最高位与 div_reg 的低 7 位[6:0]组合形成 z1。整个过程中 busy1 的值与 flag 相同。

div_as 模块：



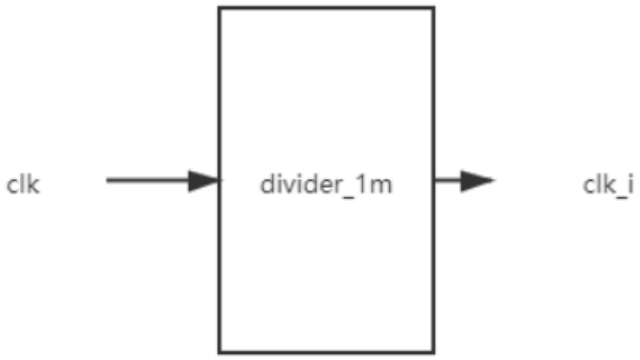
输入：时钟信号 clk，复位信号 rst_n，被除数 x 和除数 y，开始信号 start

输出：加减交替法计算 x/y 的商 z2 和余数 r2.

重要变量：plus_reg，Syn_reg,bflag

在时钟上升沿或复位信号有效时，先判断复位信号是否有效，有效则将 flag,plus_reg,Syn_reg 均置 0。否则判断 start 信号是否有效，若 start 信号有效，置 bflag 为 1,plus_reg 置为 x 的真值，Syn_reg 置为 x,y 符号位的异或。当 bflag 为 1 且 start 为 0 时，对 plus_reg 的最高位判断，若符号位为 1，则对 plus_reg 左移一位后,plus_reg 的前 8 位加$|y|_{补}$,若符号位为 0，则对 plus_reg 左移一位后，plus_reg 的前 8 位加$-|y|_{补}$,最终判断 plus_reg 的最高位，为 1 则置 plus_reg[0]为 0 否则置 1。此过程由 cnt 控制完成 7 次。cnt 为 0 时，判断余数的符号位 plus_reg 的最高位是否为 1 决定是否恢复余数，将 Syn_reg 与 plus_reg 的[13:7]位组合形成 r2，将 x 的最高位与 plus_reg 的低 7 位[6:0]组合形成 z2。整个过程中 busy2 的值与 bflag 相同。

divider_1m 模块:

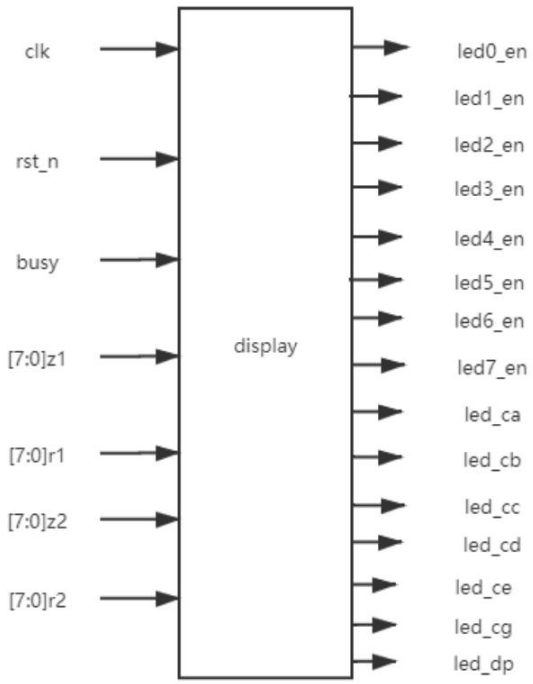


输入:时钟信号 clk

输出:1ms 的时钟信号 clk_i

设置计数器 counter 的值，控制生成时钟的周期为 1ms

display 模块:

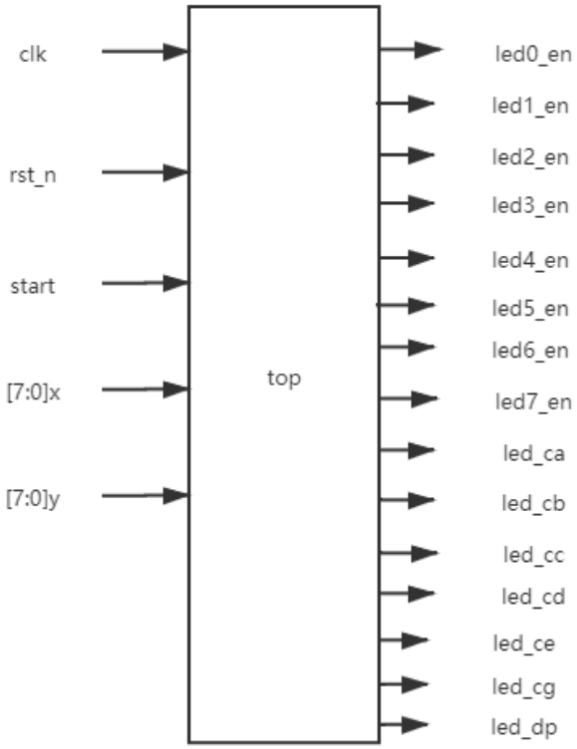


输入：时钟信号 clk，恢复余数法和加减交替法计算的商和余数 z1,r1,z2,r2，复位信号 rst_n，busy

输出：led0_en,led1_en,led2_en,led3_en,led4_en,led5_en,led6_en,led7_en,led_ca,led_cb,led_cc,led_cd,led_ce,led_cf,led_cg,led_dp

display 相当于一个存储器，根据输入的 z1,r1,z2,r2，找到其相应的 16 进制数在 7 段数码管上的显示。

top 模块:



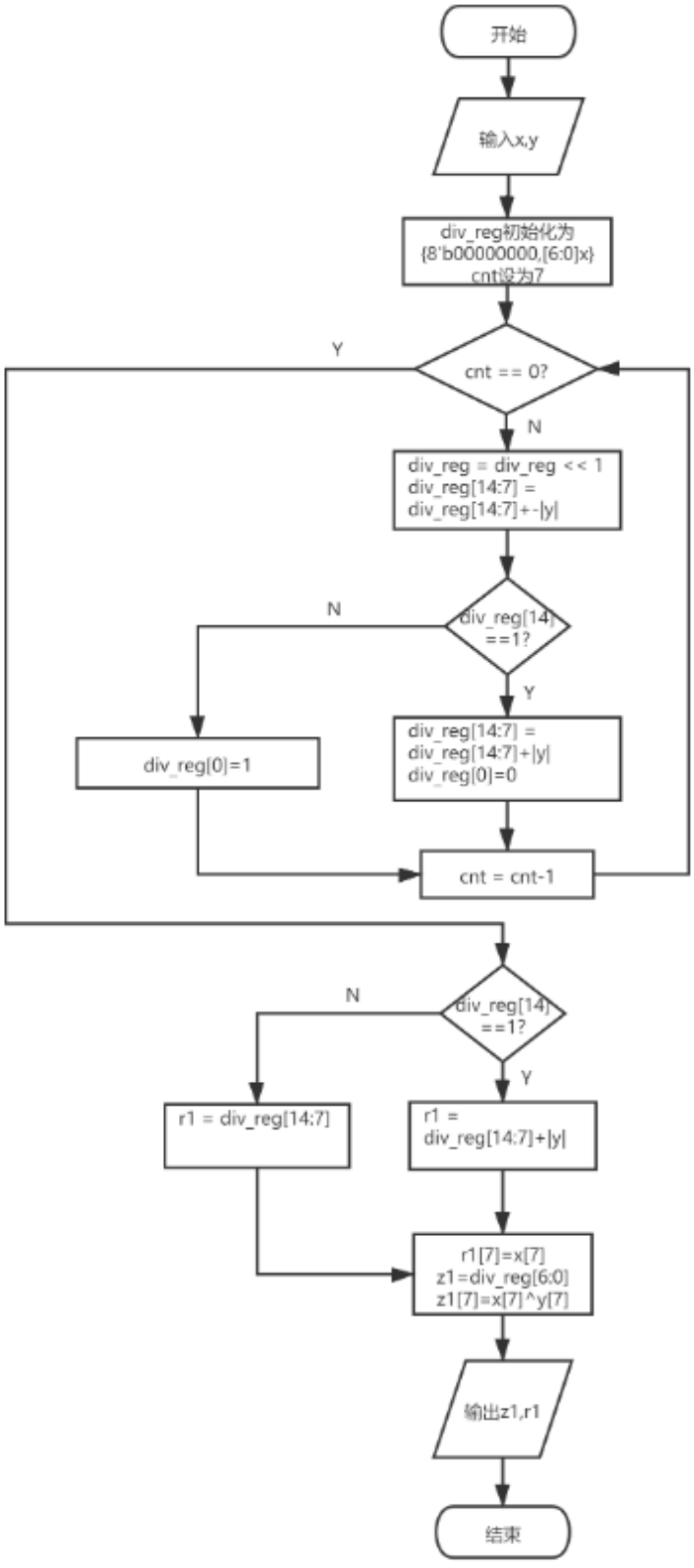
输入：时钟信号 clk，复位信号 rst_n，被除数 x 和除数 y，开始信号 start

输出：7 段数码管的控制信号 led0_en,led1_en,led2_en,led3_en,led4_en,led5_en,led6_en,led7_en,led_ca,led_cb,led_cc,led_cd,led_ce,led_cf,led_cg,led_dp

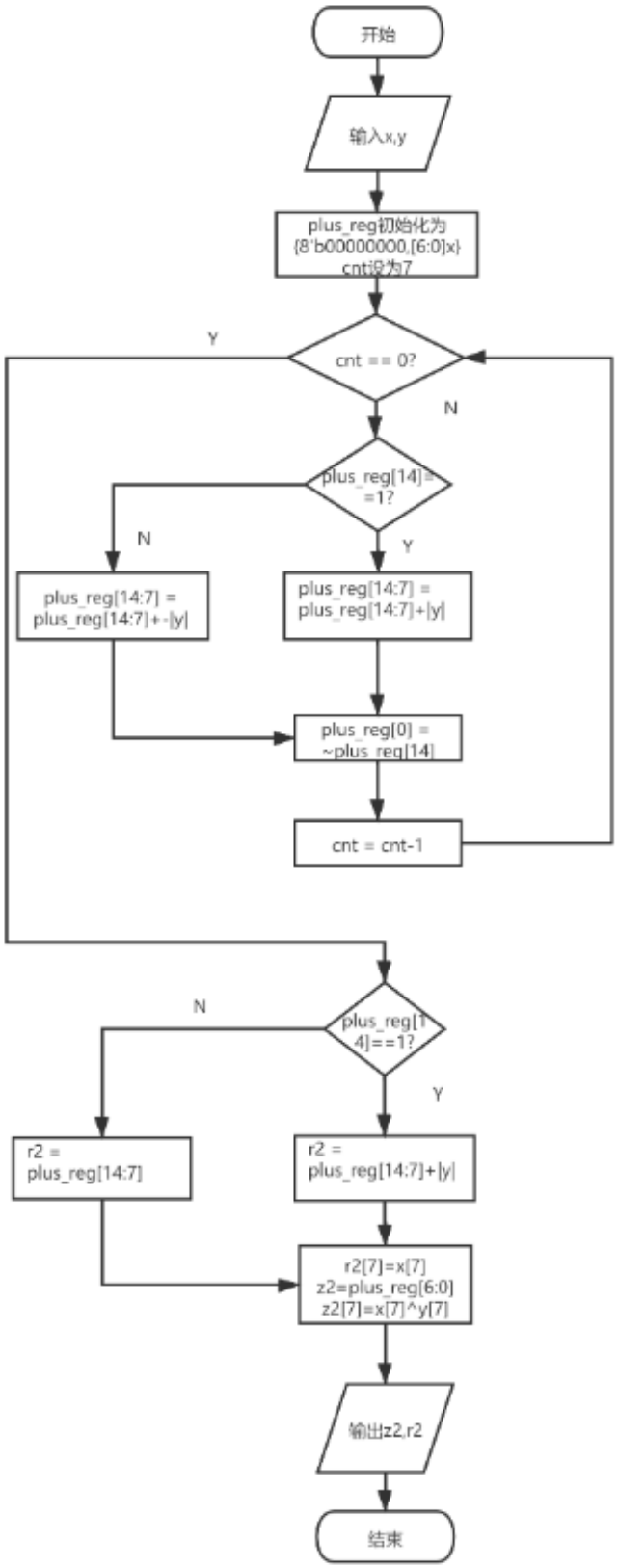
顶层模块，调用 div_as,div_rr,divider_1m,display 模块。

二、 除法器算法流程图

恢复余数法：



加减交替法：


开始
输入x,y
plus_reg初始化为{8'b00000000,[6:0]x}
cnt设为7
cnt == 0?
Y
N
plus_reg[14]==1?
N
Y
plus_reg[14:7] = plus_reg[14:7]+-|y|
plus_reg[14:7] = plus_reg[14:7]+|y|
plus_reg[0] = ~plus_reg[14]
cnt = cnt-1
plus_reg[14]==1?
N
Y
r2 = plus_reg[14:7]
r2 = plus_reg[14:7]+|y|
r2[7]=x[7]
z2=plus_reg[6:0]
z2[7]=x[7]^y[7]
输出z2,r2
结束

三、 调试报告

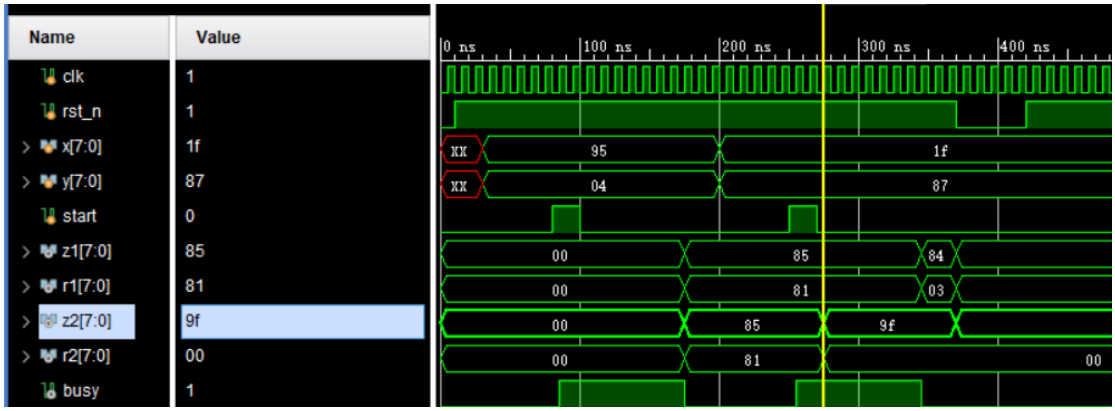
仿真核心代码：

```
#10 rst_n = 1'b1;

#20 begin x = 8'b1001_0101;y = 8'b0000_0100; end //x 为-21 y 为 4  输出 85 81

#50 start = 1'b1;

#20 start = 1'b0;



#100 begin x = 8'b0001_1111;y = 8'b1000_0111; end //x 为 31,y 为-7  输出 84 03

#50 start = 1'b1;

#20 start = 1'b0;



#100 rst_n = 1'b0;

#50 rst_n = 1'b1;

#100 begin x = 8'b0000_1101;y=8'b0000_0011;end //x 是 13,y 是 3  输出 04 01

#50 start = 1'b1;

#20 start = 1'b0;



#100 begin x = 8'b1111_1111;y=8'b1001_0100;end//x 是-127,y 是-20  输出 06 87

#50 start = 1'b1;

#20 start = 1'b0;
```

出现的问题：

1.不能重复计算多组测试数据。

仿真截图：

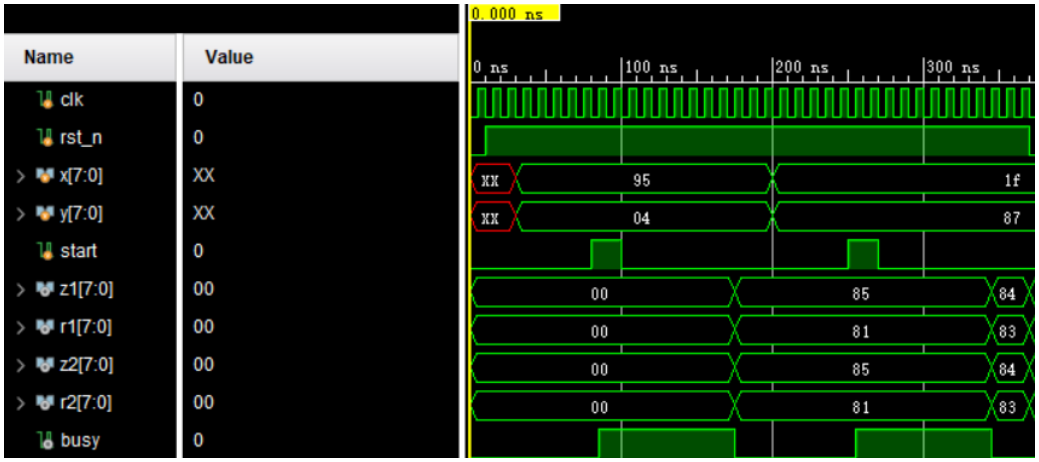


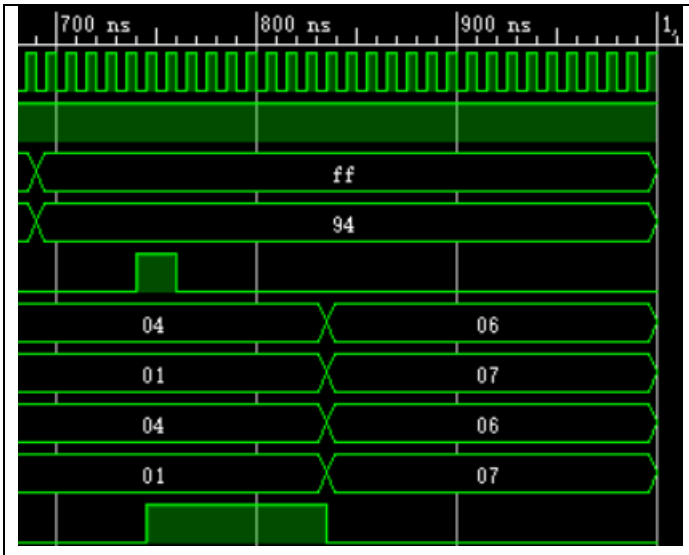
原因：在未按复位键的条件下，程序按下 start 键后，未复位计数器的值为 7，导致算法无法正确运行，z2,r2 输出错误。

解决方案：在 start==1 的情况下，设置计数器 cnt 的值为 7

2.部分测试用例余数符号出现错误。

仿真截图：

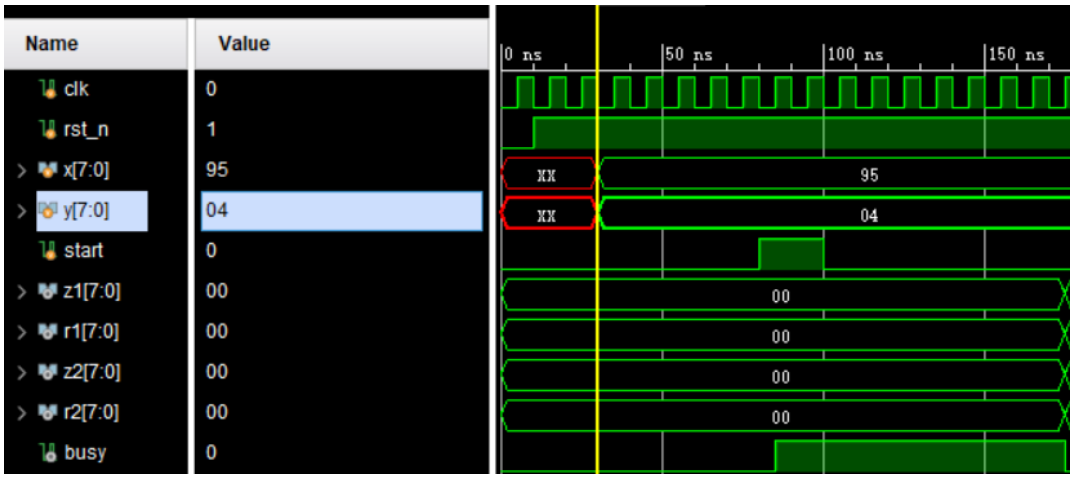


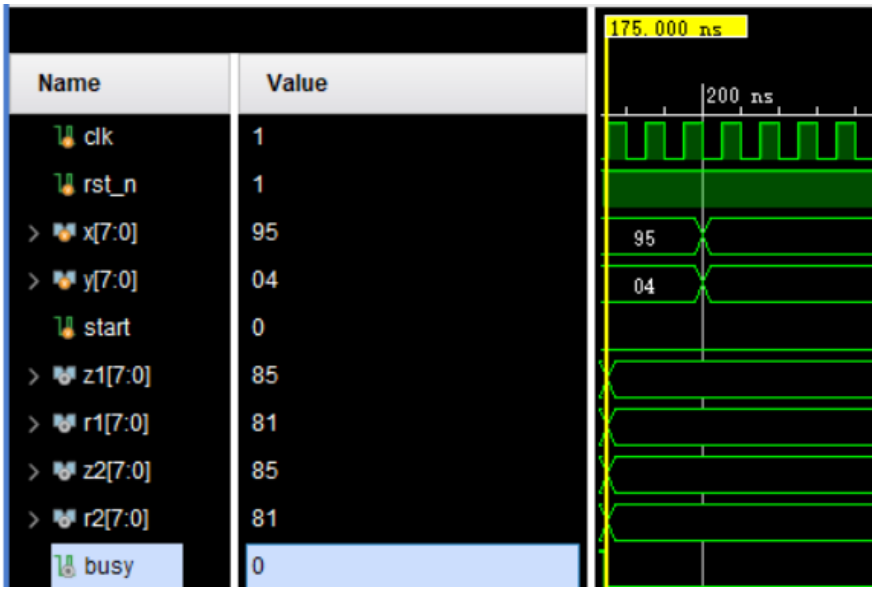


原因：余数的符号，与被除数相同而非 x 和 y 符号位的异或。

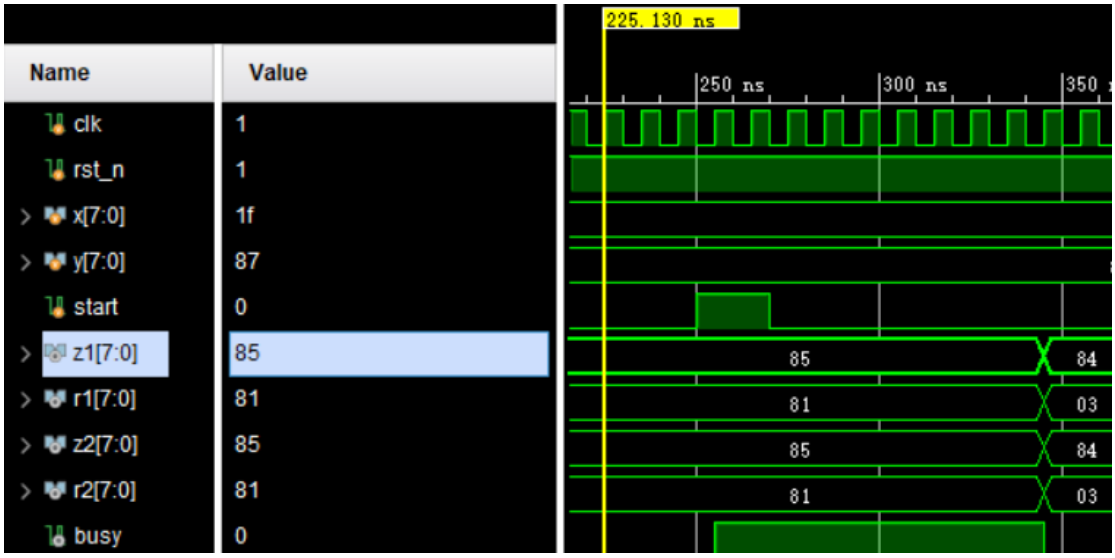
解决方案：将余数的符号 r[7]直接表示为 x[7]

解决上述问题后最终仿真截图：

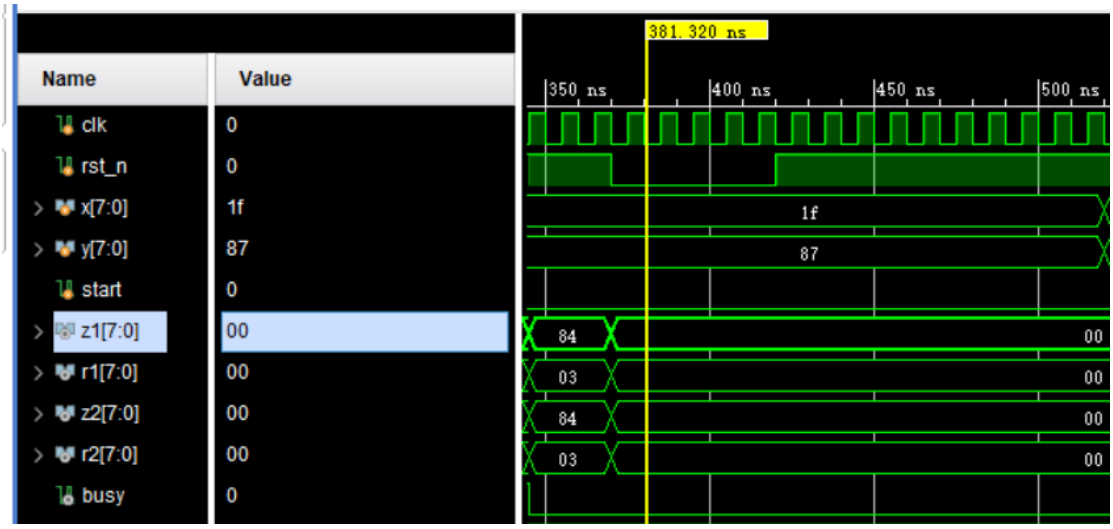




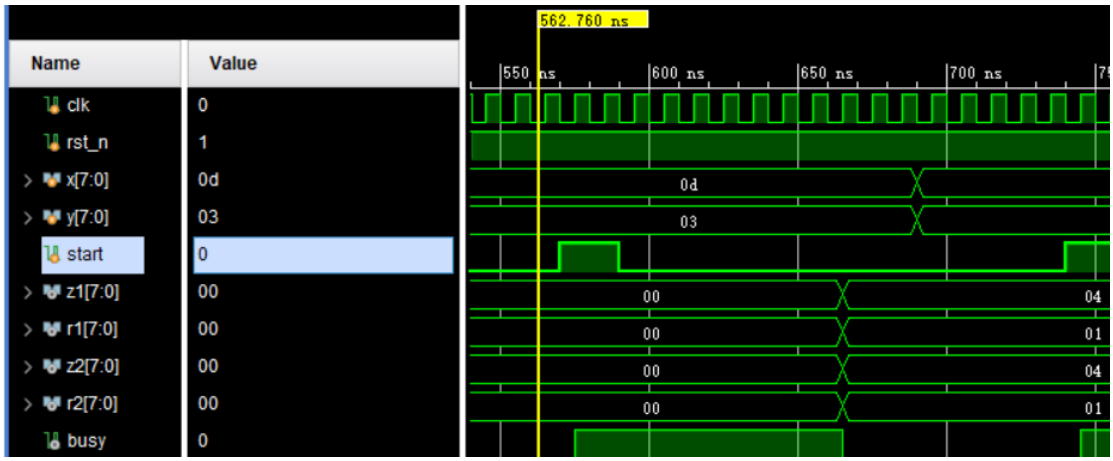
时序分析：start 为 1 且时钟处于上升沿时，立即输出 busy 信号为 1，start 为 0 时开始计算，cnt 初始化为 7，每一个时钟上升沿减一，7 个周期后输出答案 85,81(即商为-5,余数为-1)，期间 busy 信号保持为 1。最终结果的输出时间可一直维持到下一次计算结束。



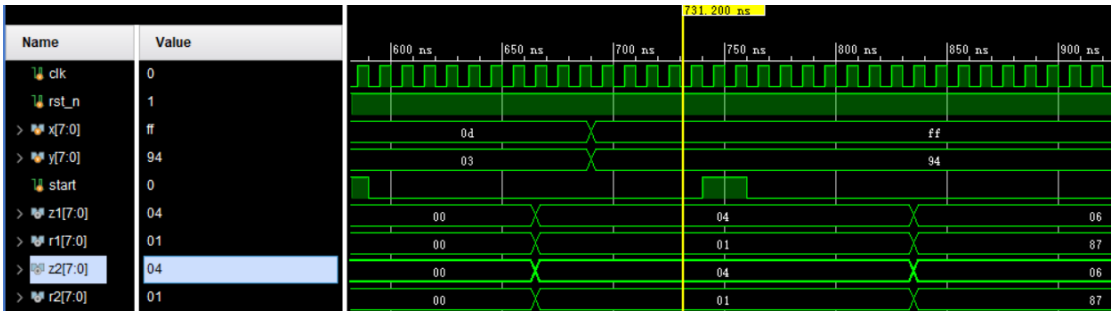
第二组测试数据，同样 start 为 1 且时钟处于上升沿时，输出 busy 为 1，start 为 0 后的 7 个周期后，输出结果 84,03（即商为-4，余数为 3），期间 busy 信号保持为 1



遇复位信号低电平有效，全部复位，清零。



第三组测试数据，同样 start 为 1 且时钟处于上升沿时，输出 busy 为 1，start 为 0 后的 7 个周期后，输出结果 04,01（即商为 4，余数为 1），期间 busy 信号保持为 1



第四组测试数据，同样 start 为 1 且时钟处于上升沿时，输出 busy 为 1，start 为 0 后的 7 个周期后，输出结果 06,87（即商为 6，余数为-7），期间 busy 信号保持为 1