

（深圳）

实验作业

开课学期： 2022春季

课程名称：计算机组成原理（实验）

实验名称： Booth乘法器设计

实验性质： 综合设计型

实验学时： 4 地点：

学生班级： 计算机类4班

学生学号： 200110428

学生姓名： 杨杰睿

作业成绩：

实验与创新实践教育中心制

2022年4月

|  |
| --- |
| 1. **Booth乘法器算法流程图** |
| Booth算法流程图：  booth.v  改进的Booth算法流程图：  booth2.v |
| 1. **调试报告** |
| booth.v仿真通过，波形截图如下所示：  booth.v乘法运算分析：  0ns-55ns：在计算开始前，各寄存器值均为0，on\_button标记为0   1. 第1次乘法分析：   55ns-65ns：start拉高一个周期，在这个周期各寄存器初始化，周期结束后完成赋值  65ns-75ns：on\_button, busy信号拉高，初始时截取y的前15位放入z的低15位的z的初始值2，，加的补码。当前高17位部分和（双符号位）为，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到  75ns-85ns：on\_button, busy信号高位，，加的补码。当前高17位部分和为，周期结束后部分和加上原z值并右移1位，符号扩展得到z=[0 0 0 0 c 0 0 0]。  85ns-95ns：，加的补码。当前高17位部分和（双符号位）为，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到。  95ns-105ns：[yn+1 yn]=[1 0]，加x的补码[0 0 0 6]。当前高17位部分和（双符号位）为z\_pp=[0 0 0 0 3]，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到z=[0 0 0 0 f 0 0 0]。  105ns-225ns：[yn+1 yn]=[0 0]。当前z逐次右移，直到15次右移完成后，busy信号拉低，最终得到z=[0 0 0 0 0 0 1 e]。  225ns-285ns：z值保持不变，计数器cnt保持不变，busy信号保持低位。   1. 第2次乘法分析：   285ns-295ns：start拉高一个周期，在这个周期各寄存器初始化，周期结束后完成赋值，得到初始化z=[0 0 0 0 0 0 0 2]。  295ns-305ns：[yn+1 yn]=[0 1]，加-x的补码[0 0 0 6]。当前高17位部分和（双符号位）为z\_pp=[0 0 0 0 6]，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到z=[0 0 0 1 8 0 0 1]。  305ns-315ns：[yn+1 yn]=[1 0]，加x的补码[f f f a]。当前高17位部分和（双符号位）为z\_pp=[1 f f f d]，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到z=[f f f f 4 0 0 0]。  315ns-325ns：[yn+1 yn]=[0 1]，加-x的补码[0 0 0 6]。当前高17位部分和（双符号位）为z\_pp=[0 0 0 0 4]，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到z=[0 0 0 1 2 0 0 0]  325ns-335ns：[yn+1 yn]=[1 0]，加x的补码[f f f a]。当前高17位部分和（双符号位）为z\_pp=[1 f f f c]，周期结束后，部分和加上原z值并完成右移1次，进行符号扩展，得到z=[f f f f 1 0 0 0]。  335ns-455ns：[yn+1 yn]=[0 0]。当前z逐次右移，直到15次右移完成后，busy信号拉低，最终得到z=[f f f f f f e 2]。  455ns-515ns：z值保持不变，计数器cnt保持不变，busy信号保持低位。   1. 第5次乘法分析（因此次执行了完整的16个周期计算，且含有debug过程记录附后）   具体的分析过程同上两次乘法运算，下面简要总结过程中出现的计算情况：   * 从1215ns-1375ns共计16个时钟周期，busy处于高位，总共完成完整的16个周期的计算过程。 * 输入的x，y分别为[8 0 0 1]和[6 5 7 8]，经计算得的-x为[7 f f f]（均为补码表示） * 向量comparator即[yn+1 yn]在各个周期（16个计算周期+剩余周期）的取值为：   [0 0] => [0 0] => [0 0] => [0 1] => [1 1] => [1 1] => [1 1] => [1 0] =>  [0 1] => [1 0] => [0 1] => [1 0] => [0 0] => [0 1] => [1 1] => [1 0] =>  [0 0] ……（此后全[0 0]）   * 对于取值为[0 0]和[1 1]的周期，z右移1位，并进行符号扩展 * 对于取值为[0 1]的周期，部分和z\_pp加上-x，与z的高17位相加，并完成z的右移和符号扩展 * 对于取值为[1 0]的周期，部分和z\_pp加上x，与z的高17位相加， 并完成z的右移和符号扩展   波形图如上所示，需要特别指出的是，在第16个周期，即cnt=15时（因为0时占据了一个周期），z在完成计算后，不再进行右移，因为z的低位从起初是截取的y的高15位存放的，在计算过程中，当y的高15位被完全移出z后，计算完成，总共需要右移15次，故第16个周期时，在不应当继续右移。最终可以计算出z的正确答案为z=[c d 4 4 6 5 7 8]。  下附debug过程的手算记录，代码此前因最后1个周期没有计算部分和z\_pp导致了错误，检查计算过程后更正了代码：  booth2.v仿真通过，波形截图如下：  booth2.v乘法运算分析：   1. 第1次乘法分析   0-55ns：在rst\_n高位后，第一次start启动前，寄存器的值均置0  55ns-65ns：start高位，这个周期中start的高有效使得x\_reg和y\_reg在该周期结束后被赋值，z作为线网类型其低位在这一周期结束后通过z\_reg赋值与y的高14位相同。  65ns-75ns；busy高位，乘法器开始正式计算，z低位的值在这一周期被赋值为y的高14位即（二进制）[0000 0000 0000 0101]的前14位，为（十六进制）[00000001]。[yn+2 yn+1 yn]=[0 1 0]，z\_pp保存z[30:14]与x的双符号位补码之和，在这一周期结束后，z的值高位由部分和z\_pp右移2位后填充，低位为原z[13:2]，在这周期结束后，z=[00006000]。  75ns-85ns：busy高位，[yn+2 yn+1 yn]=[0 1 0]，z\_pp保存z[30:14]与x的双符号位补码之和，在这一周期结束后，z的值高位由部分和z\_pp右移2位后填充，低位为原z[13:2]，在这周期结束后，z=[0 0 0 0 7 8 0 0]  85ns-145ns：busy高位，[yn+2 yn+1 yn]=[0 0 0]，z\_pp保存z[30:14]与x的双符号位补码之和，在每一周期结束后，z的值高位均由部分和z\_pp右移2位后填充，低位为原z[13:2]，在这6个周期结束后，z=[0 0 0 0 0 0 1 e]  145ns-195ns：busy低位，计算已经在上一周期结束，z保持不变，[yn+2 yn+1 yn]收到算法的影响会进行一次计算，之后保持不变。计数器保持不变。   1. 第3次乘法分析   335-345ns：本次start启动前，寄存器的值保持上一周期结果不变，在start拉高位之后，各寄存器的值将在本周期结束后更新。  345ns-425ns：start高位，这8个周期中[yn+2 yn+1 yn]变化如下所示：  [0 1 0] => [0 0 0] => [0 0 1] => [1 0 0] => [0 1 1] => [1 1 1] => [1 1 1]  但受到x取值为0的影响，部分积z\_pp始终为0，在进行计算（如[0 1 0]为x的1倍加法，[0 0 1]为-x的2倍加法）的过程中，实际上相当于始终每个周期将z的值友谊2位，最终得到z=[0 0 0 0 0 0 0 0]  425ns-475ns：busy低位，计算已经在上一周期结束，z保持不变，[yn+2 yn+1 yn]收到算法的影响会进行一次计算，之后保持不变。计数器保持7不变。   1. 第5次乘法分析   755ns-765ns：本次start启动前，寄存器的值保持上一周期结果不变，在start高位之后，各寄存器的值将在本周期结束后更新。  765ns-775ns：[yn+2 yn+1 yn]=[0 0 0]，z\_pp= z[30:14],z= {z\_pp[16], z\_pp[16], z\_pp, z[13:2]},最终z=[0000195e]  775ns-785ns：[yn+2 yn+1 yn]=[0 0 1]，z\_pp= {nx\_reg[15], nx\_reg} + {nx\_reg[15], nx\_reg} + z[30:14], z= {z\_pp[16], z\_pp[16], z\_pp, z[13:2]}，最终z=[00000657]  785ns-795ns：[yn+2 yn+1 yn]=[1 1 1], z\_pp = z[30:14], z= {z\_pp[16], z\_pp[16], z\_pp, z[13:2]}, 最终z=[0fffe195]  795ns-805ns: [yn+2 yn+1 yn]=[1 1 0], z\_pp = {x\_reg[15], x\_reg} + {x\_reg[15], x\_reg} + z[30:14], 最终z=[03fff865]  805ns-815ns及815ns-825ns: [yn+2 yn+1 yn]=[0 1 0], z\_pp= {x\_reg[15], x\_reg} + z[30:14]，z= {z\_pp[16], z\_pp[16], z\_pp, z[13:2]}, 最终z=[74401786]  825ns-835ns: [yn+2 yn+1 yn]=[0 0 1], z\_pp={nx\_reg[15], nx\_reg} + {nx\_reg[15], nx\_reg} + z[30:14], 最终z=[751015e1]  835ns-845ns: [yn+2 yn+1 yn]=[1 1 0], z\_pp={x\_reg[15], x\_reg} + {x\_reg[15], x\_reg} + z[30:14], 最终z=[0d43e578]  845ns-855ns:在上一周期结束后计算出z=[cd446578]，在本周期开始前完成赋值  剩余周期各寄存器保持不变，直到下一次start高位 |
|  |