华中科技大学

计算机组成原理实验报告

实验名称：运算器实验

班　　级：人工智能 2204 班

学　　号：U202214123

姓　　名：陈博

报告日期：2024 年 5 月 19 日

目录

1. 实验内容及目的

1.1 快速加法器实验

掌握串行加法器逻辑实现，能设计 8 位可控加减法电路，掌握快速加法器逻辑实现，能设计 4 位先行进位电路，能设计 4 位快速加法器，掌握组内先行、组间先行的基本原理，利用 4 位快速加法器构建 16 位、32 位快速加法器，能分析相关电路延迟。

1.2 乘法器实验

理解阵列乘法器的实现原理能设计无符号阵列乘法器电路，能设计有符号补码阵列乘法器电路，掌握原码、补码一位乘法基本原理，能设计原码、补码一位乘法器，重点掌握寄存器、多路选择器的使用，能设计简单的状态机进行数据通路控制。

1.3 算术逻辑运算单元 ALU 实验

掌握定点数加减法溢出检测方法，熟悉 Logisim 中各种运算组件的使用方法，逻辑运算部件、乘法器、除法器、移位器，熟悉多路选择器的使用，设计 32 位 ALU，利用已完成的 32 位加法器、Logisim 其他运算组件构建。

2. 实验电路、设计思路及结果

2.1 快速加法器实验

快速加法器实验包括：8 位可控加减法电路，4 位先行进位电路，4 位快速加法器，16 位、32 位快速加法器。

1） 8 位可控加减法电路

8 位可控加减法电路的整体设计思路如图 1。当我们进行加法运算的时候，Sub 的值为 0，此时进行的操作即为 X+Y；而当我们进行减法运算的时候，Sub 的值为 1，此时对 Y 的每一位都取反，并且 Cin 为 1，达到了取反加一求 Y 的

补码的效果，因此最后实现了 X-Y。

最后，溢出信号由向最高位的进位和最高位进位相异或产生。实验电路图如图二所示，结果如图 3 所示。

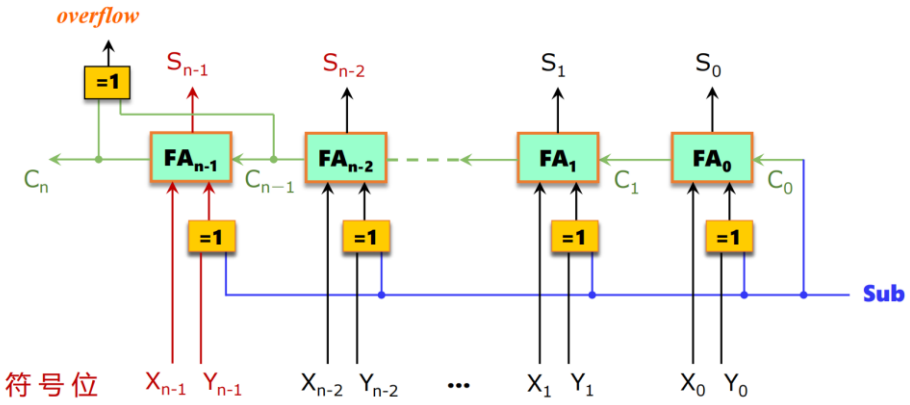

图 1 8 位可控加减法电路的设计思路

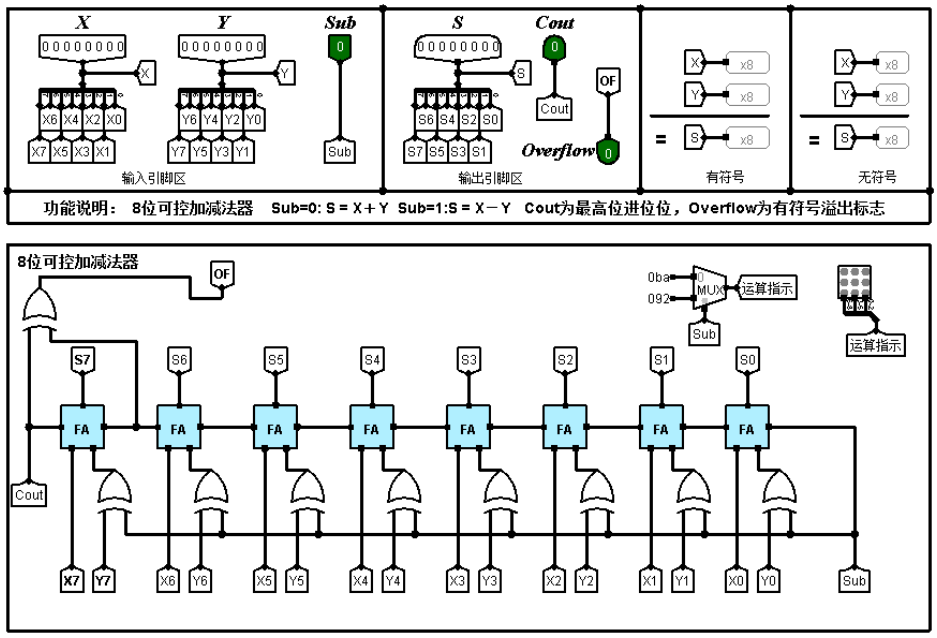

图 2 8 位可控加减法电路的电路图

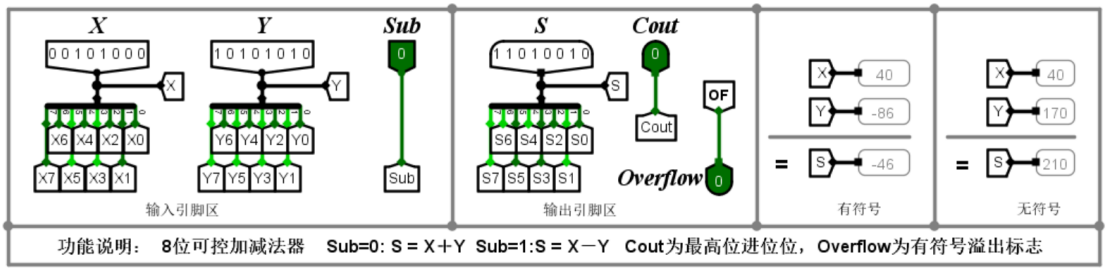

图 3 8 位可控加减法电路的实验结果

2） 4 位先行进位电路

通过提前计算每一位的进位，在计算时高位就不需要等待低位产生进位才能计算，这样可以大大减少延迟时间。

对于进位的计算，我们有进位生成函数和进位传递函数：

$$\begin{aligned} G_i &= X_i Y_i \\ P_i &= X_i \oplus Y_i \\ C_i &= G_i + P_i C_{i-1} \end{aligned} \quad , \tag{1}$$

在 4 位先行进位电路中，我们有：

$$\begin{aligned} C_1 &= G_1 + P_1 C_0 \\ C_2 &= G_2 + P_2 G_1 + P_2 P_1 C_0 \\ C_3 &= G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_0 \\ C_4 &= G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 + P_4 P_3 P_2 P_1 C_0 \end{aligned} \quad , \tag{2}$$

于是，我们可以根据此来设计电路，电路图如图 4 所示。

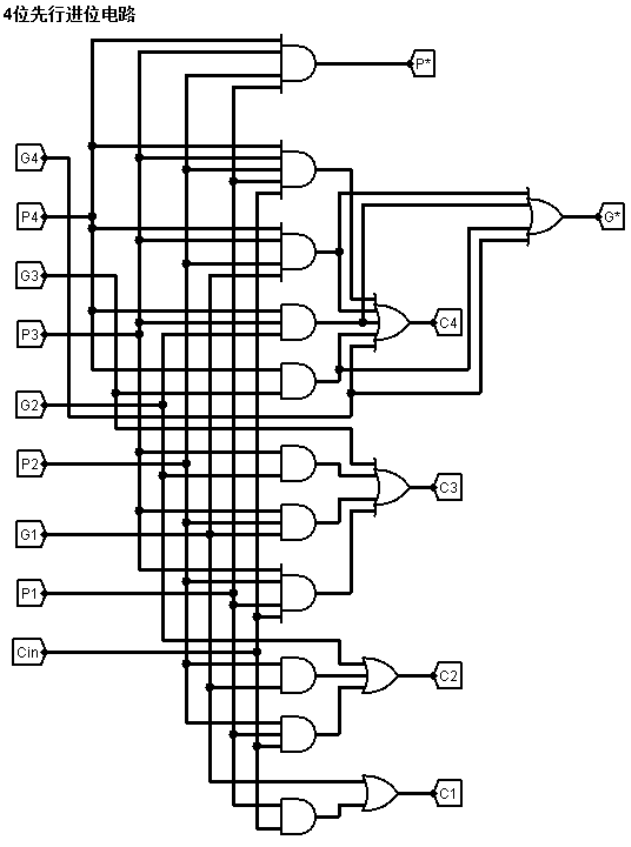

图 4 4 位先行进位电路

3） 4 位快速加法器

有了先行进位电路，我们再使用简单的逻辑电路，即可实现 4 位快速加法器，原理如下：

$$P_i = X_i \oplus Y_i$$
$$S_i = X_i \oplus Y_i \oplus C_i = P_i \oplus C_i \ , \tag{3}$$

电路图设计如图 5，电路实验结果如图 6。

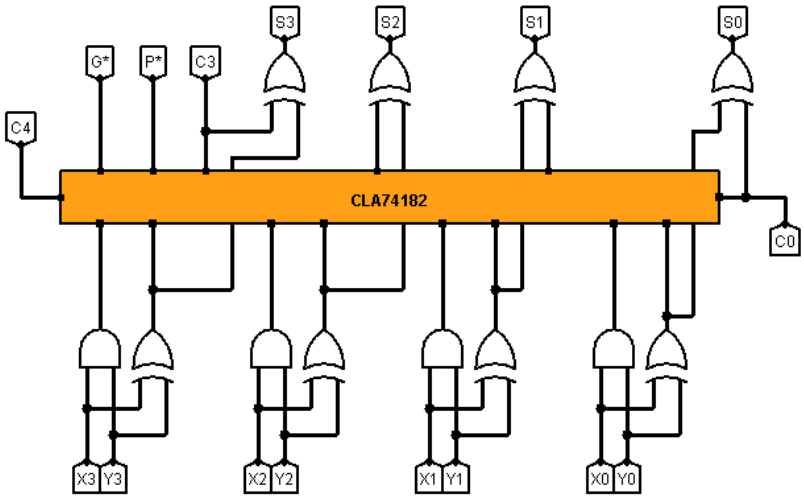

图 5 4 位快速加法器电路

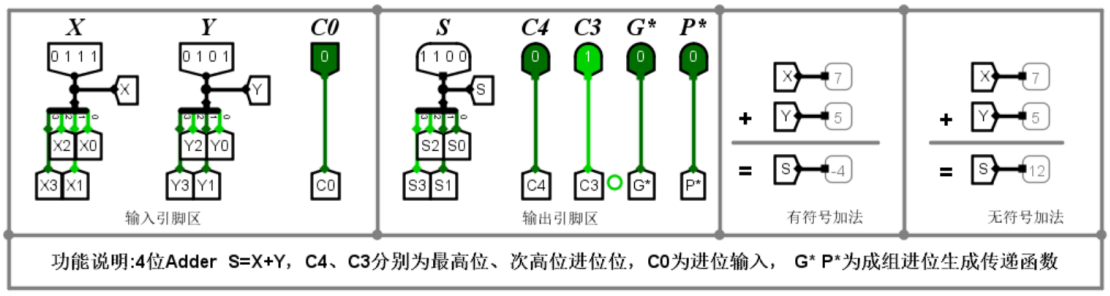

图 6 4 位快速加法器实验结果

4） 16 位、32 位快速加法器

有了四位快速加法器，把它当成一个模块，通过使用组内先行进位，组间串行进位，将 4 位先行进位拓展为成组进位生成函数和成组进位传递函数，我们可以实现 16 位快速加法器，实验电路图如图 7 所示，实验结果如图 8 所示。同理，使用两片 16 位快速加法器，我们也可以实现 32 位快速加法器，实验电路图如图 9 所示，实验结果如图 10 所示。

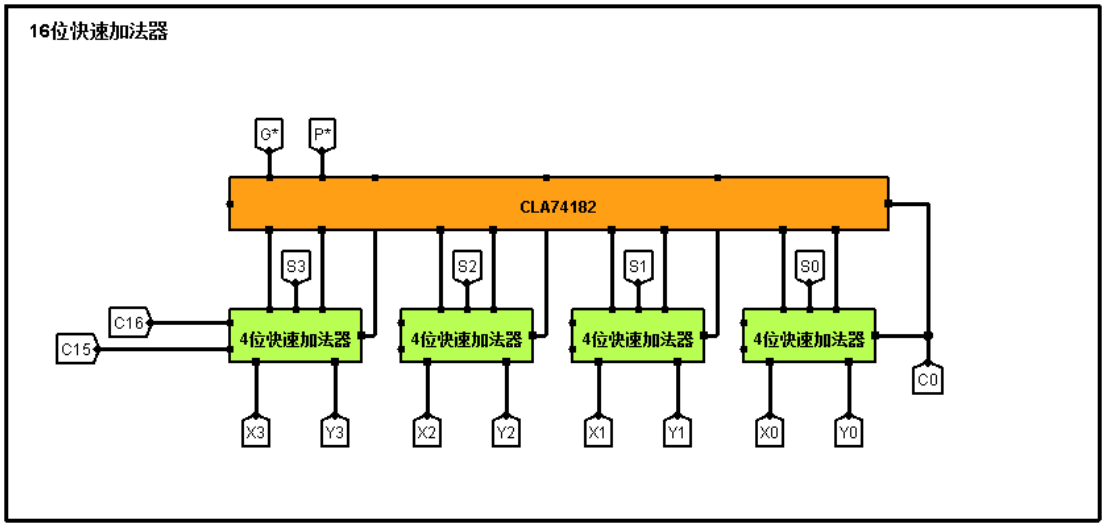

图 7 16 位快速加法器电路

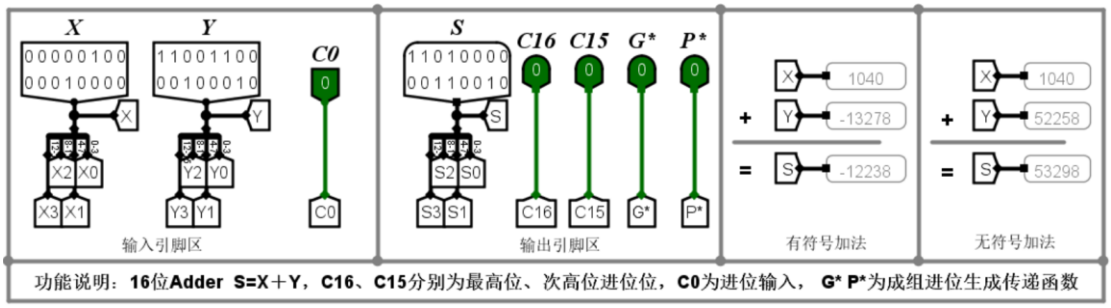

图 8 16 位快速加法器实验结果

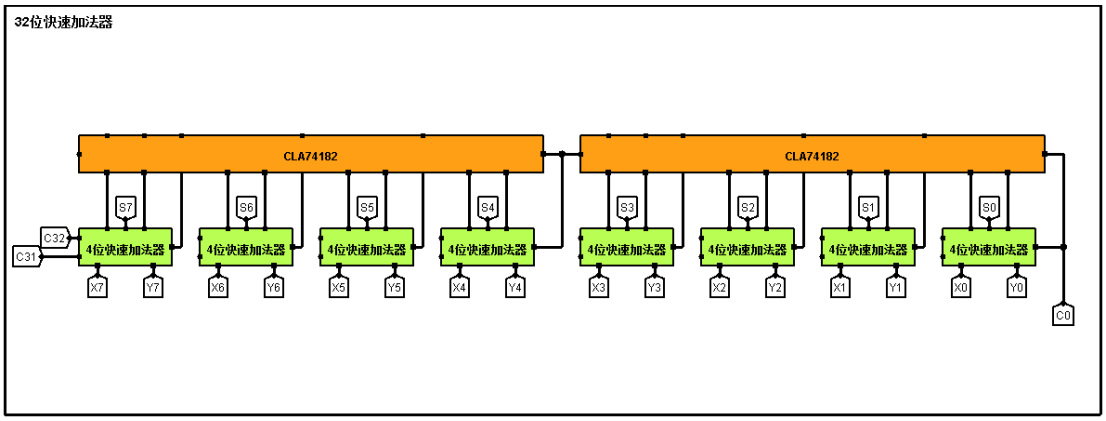

图 9 32 位快速加法器电路

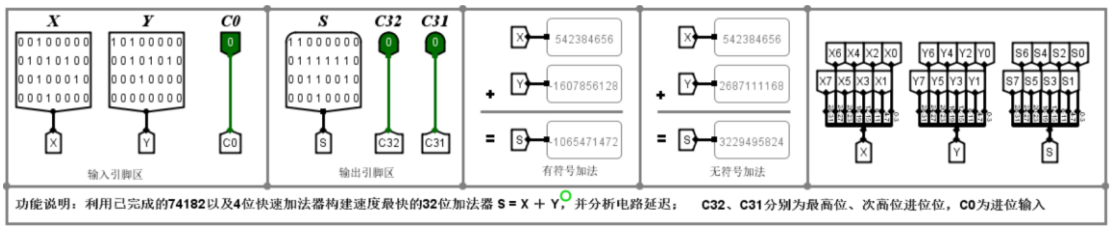

图 10 32 位快速加法器实验结果

2.2 乘法器实验

1) 5 位阵列乘法器

跟我们手工计算乘法的思路一样，5*5 的横向阵列乘法器包括 4 行全加器，每行 5 个，需要 20 个全加器，进位信号横向传递，每一行都是一个 5 位串行进位加法器。具体的电路图如图 11 所示，实验结果如图 12 所示。

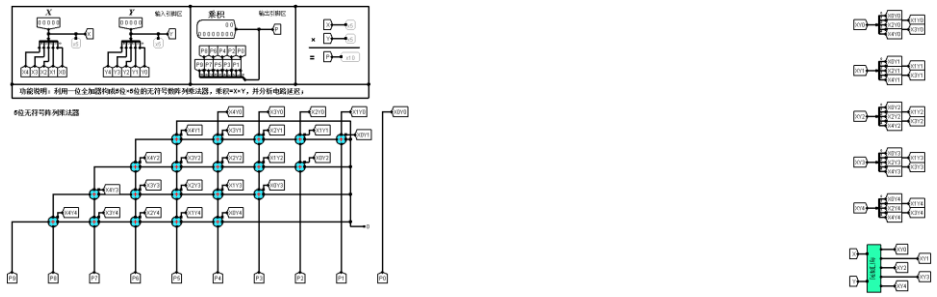

图 11 5 位阵列乘法器电路

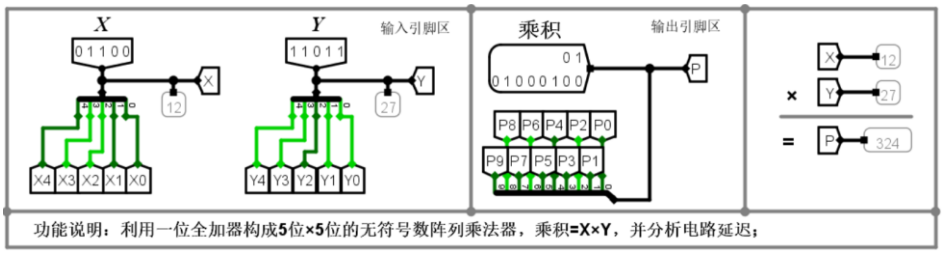

图 12 5 位阵列乘法器实验结果

2) 6 位补码阵列乘法器

首先对于补码的运算来说，关键就是要将符号位与其数值位区分开来，如果这个数是一个正数，那么就直接用数值位进行相应的运算，然后前面在加上符号位即可，但如果这是一个负数的话，就需要特殊处理这个数值位，然后根据相应的逻辑再转化即可。然后利用 5 位阵列乘法器，最后添加上结果的符号位，即可实现补码阵列乘法器，具体电路图如图 13 所示，结果如图 14 所示。

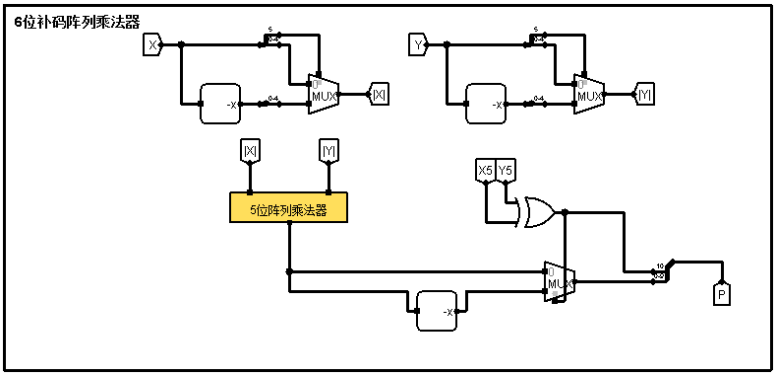

图 13 6 位补码阵列乘法器电路

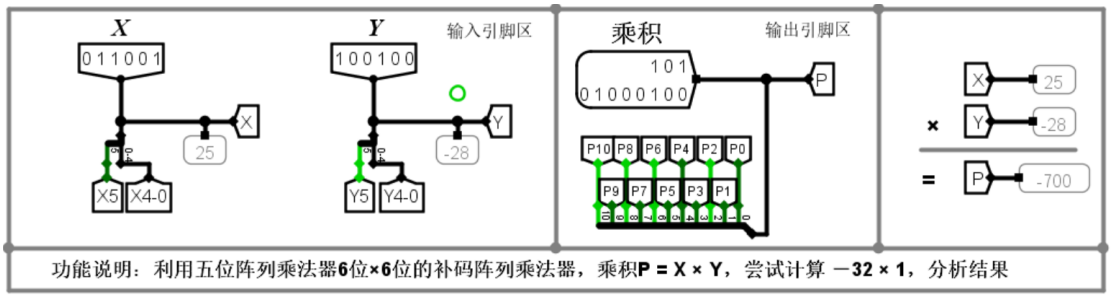

图 14 6 位补码阵列乘法器结果

3) 原码一位乘法器

实现原码一位乘法器的大致思路如图 15 所示。通过将符号位与数据位分开计算，Yn 决定当前步骤是加上|X|还是 0，Y 和结果一起右移，可以减少不必要的存储空间，这就是原码一位乘法器的大致思路。

电路图的大致框架如图 16 所示，R0 存放∑高 n 位，初值为 0，R1 存放 Y、∑其它位（如何载入 Y 初值），运算结果右移后送寄存器输入。时钟到来，R0，R1 锁存新值，状态机控制使能信号停机，停机后最终乘积存放在 R0，R1 中。如何对 R1 进行初始化，以及如何确保乘法能够停止，需要单独对时钟信号进行计数，当次数为 0 时，R1 载入初始值 Y，而当移位的次数达到上限时，就停止，这里是 8 位，一共移位 8 次。最终电路图如图 17 所示，实验结果如图 18 所示。

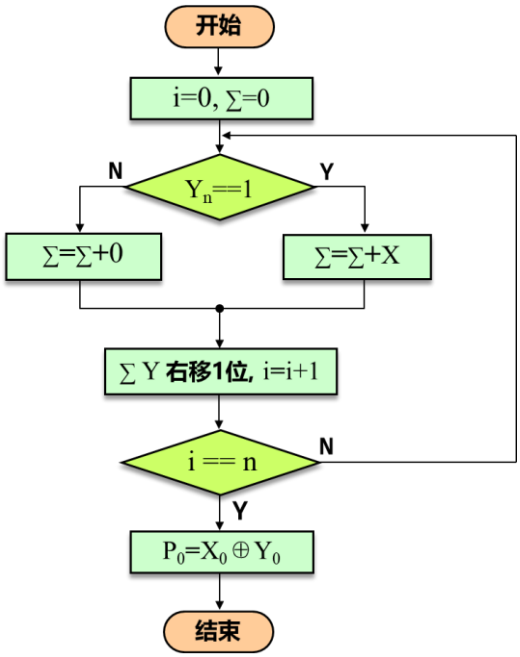

图 15 原码一位乘法器的实现思路

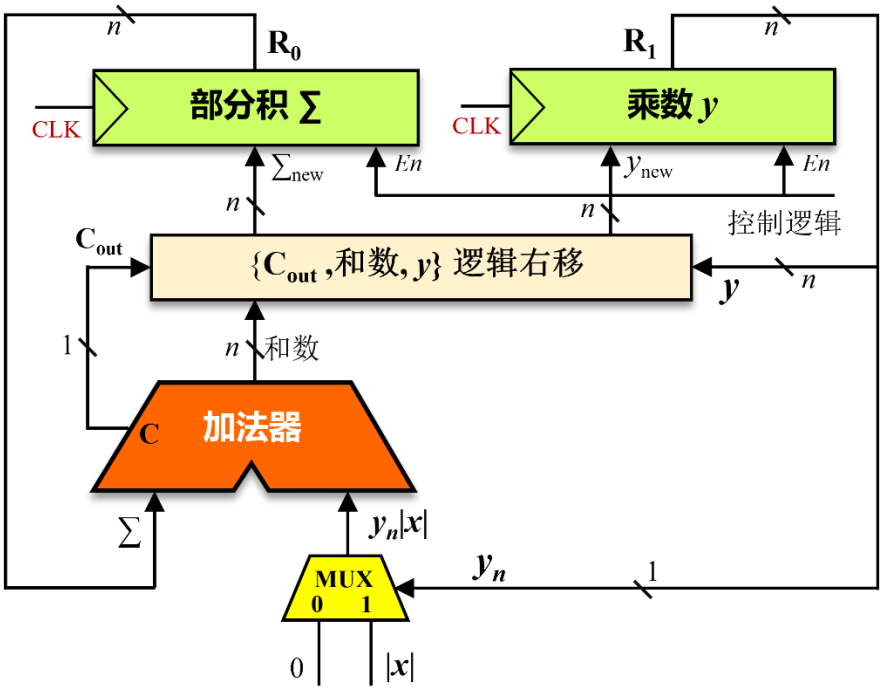

图 16 原码一位乘法器的电路图框架

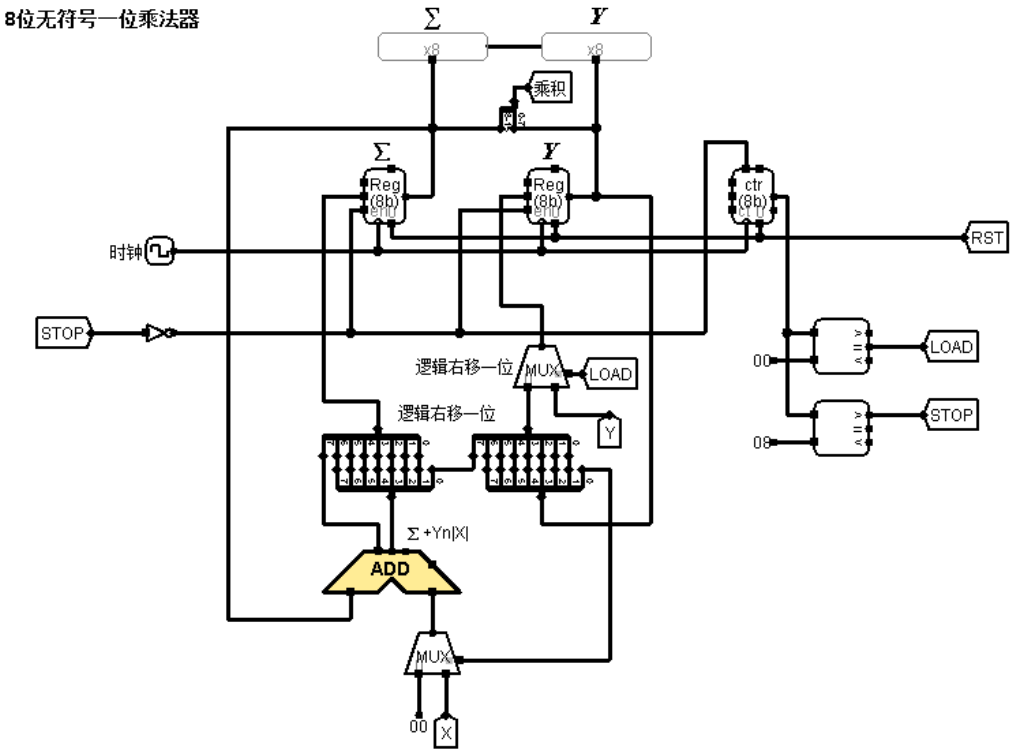

图 17 原码一位乘法器的电路图

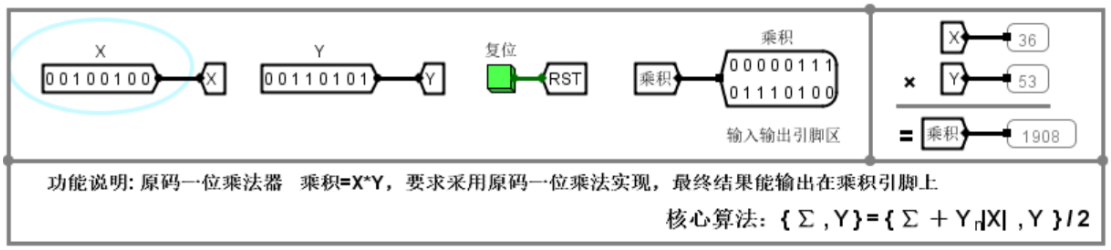

图 18 原码一位乘法器实验结果

4) 补码一位乘法器

实现原码一位乘法器的大致思路如图 19 所示。其中有 n+1 次加法，n 次移位，所以控制逻辑电路和原码一位乘法器一样。每次判断需要 Yn 和 Yn+1，在原码一位乘法器的基础上改变这些地方即可实现补码一位乘法器，实验电路图如图 20 所示，实验结果如图 21 所示。

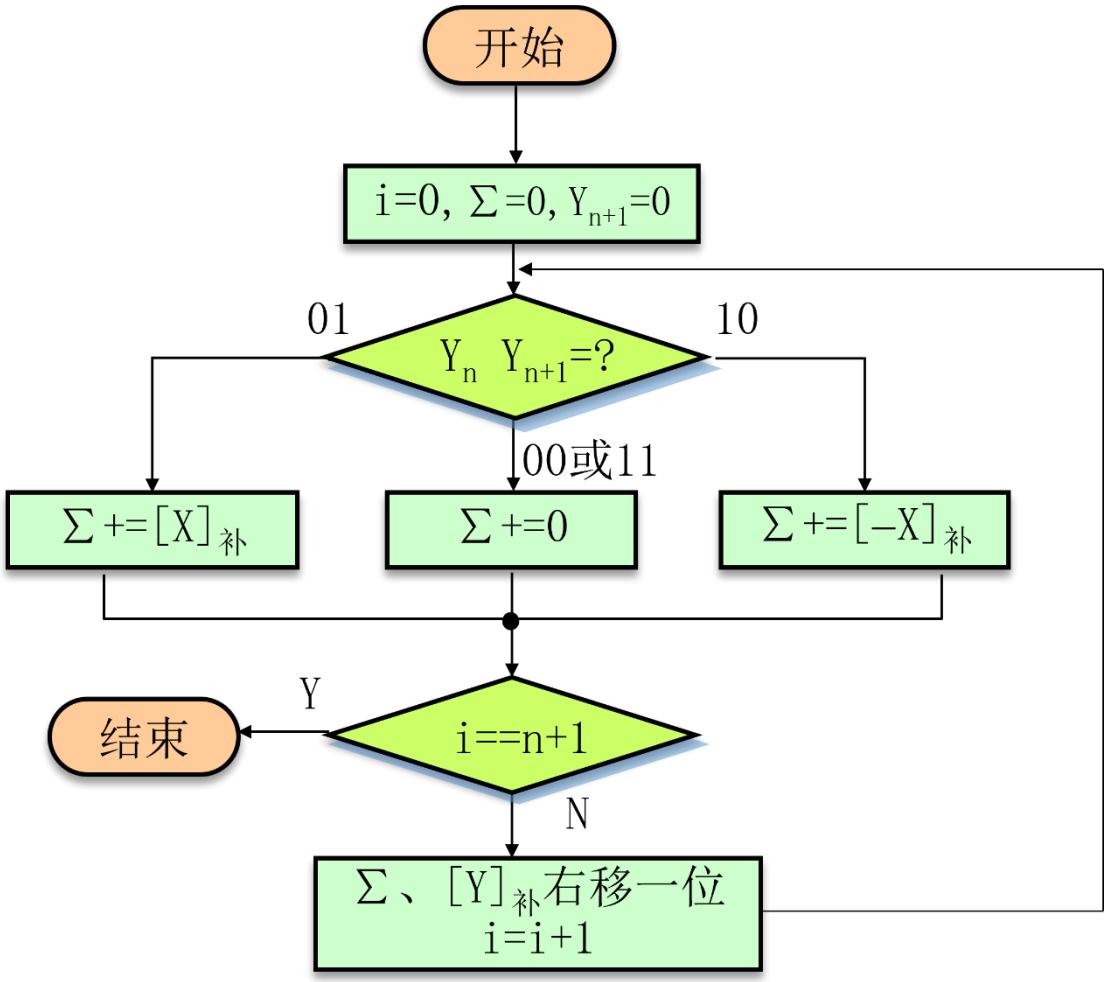

图 19 补码一位乘法器的实现思路

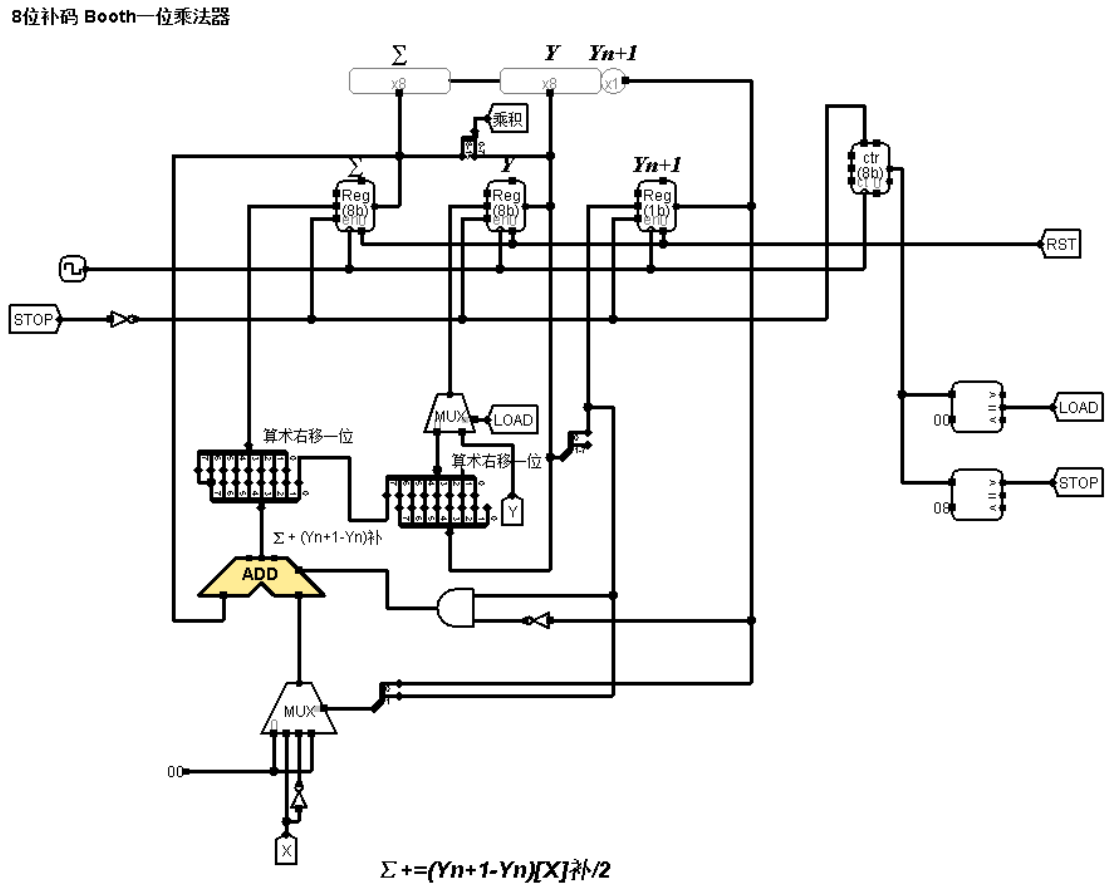

图 20 补码一位乘法器的电路图

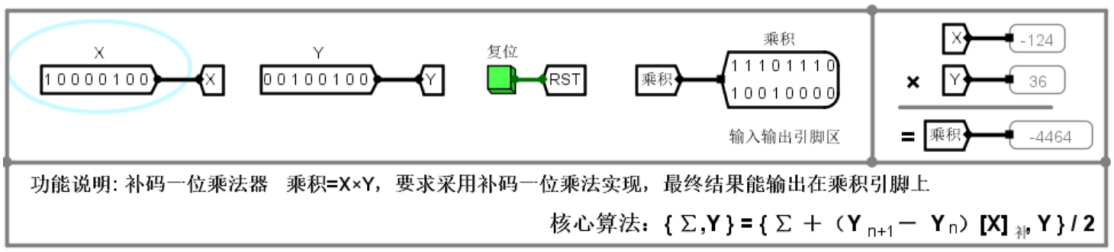

图 21 补码一位乘法器实验结果

2.3 算术逻辑运算单元 ALU 实验

实验的要求如图 22 所示，前三个使用移位器即可实现，然后 3 和 4 分别使用乘法器和除法器即可实现，5 和 6 使用前面的 32 位加法器，减法的时候对 y 求补码即可，7~12 使用软件自带的单元即可，最终，用 S 对这些信号进行片选，输出 Result，Result2，OF，UOF。实验电路如图 23 所示，实验结果如图 24 所示。

ALU_OP	十进制	运算功能
0000	0	Result = X << Y 逻辑左移（Y取低五位） Result2=0
0001	1	Result = X >> Y 算术右移（Y取低五位） Result2=0
0010	2	Result = X >> Y 逻辑右移（Y取低五位） Result2=0
0011	3	Result = $(X * Y)_{[31:0]}$ Result2 = $(X * Y)_{[63:32]}$ 无符号乘法
0100	4	Result = X/Y Result2 = X%Y 无符号除法
0101	5	Result = X + Y (Set OF/UOF)
0110	6	Result = X - Y (Set OF/UOF)
0111	7	Result = X & Y 按位与
1000	8	Result = X | Y 按位或
1001	9	Result = X⊕Y 按位异或
1010	10	Result = ~(X |Y) 按位或非
1011	11	Result = (X < Y) ? 1 : 0 有符号比较
1100	12	Result = (X < Y) ? 1 : 0 无符号比较

图 22 ALU 实验要求

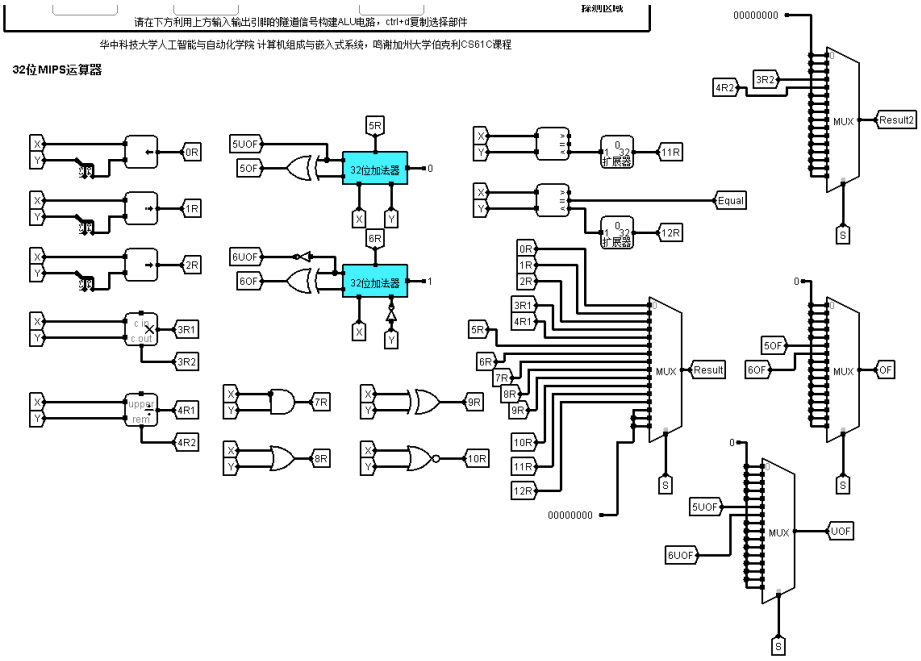

图 23 ALU 实验电路图

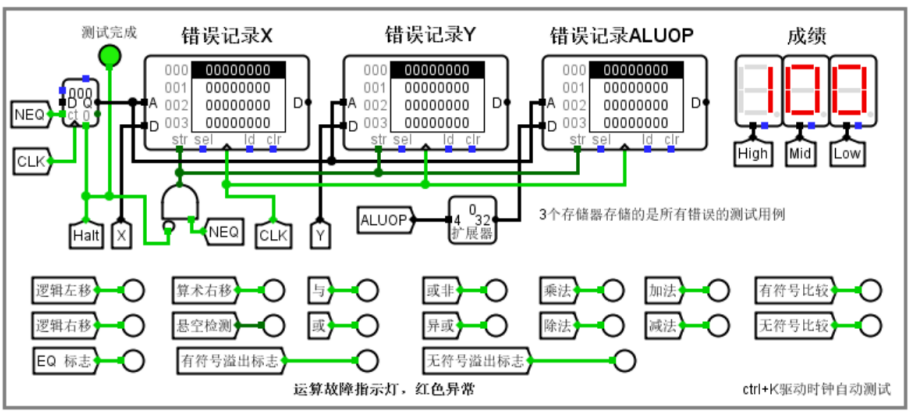

图 23 ALU 实验结果

3. 实验中遇到的问题

1) 快速加法器实验

在做 8 位可控加减法电路的时候，遇到了红线的情况，如图 24 所示。然后我回头检查，发现 Y5 和 Sub 接到了一起，然后将其修正，发现还是有问题，红线还是没有消失，经过各种尝试，都没有成功，最后将软件重启就好了……

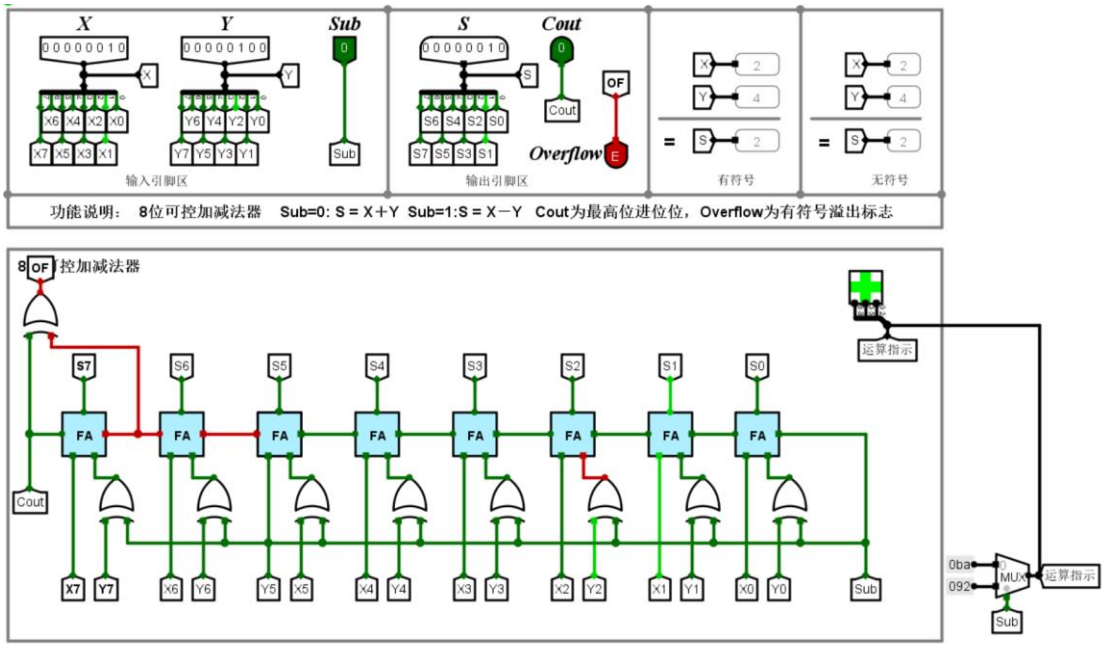

图 24 8 位可控加减法电路出现的问题

在 4 位快速加法器的实验中，出现了计算错误的问题，如图 25 所示，然后我去检查电路，发现下方接错了，接成了或门，应该是异或门。这种问题在第二次实验还在出现。

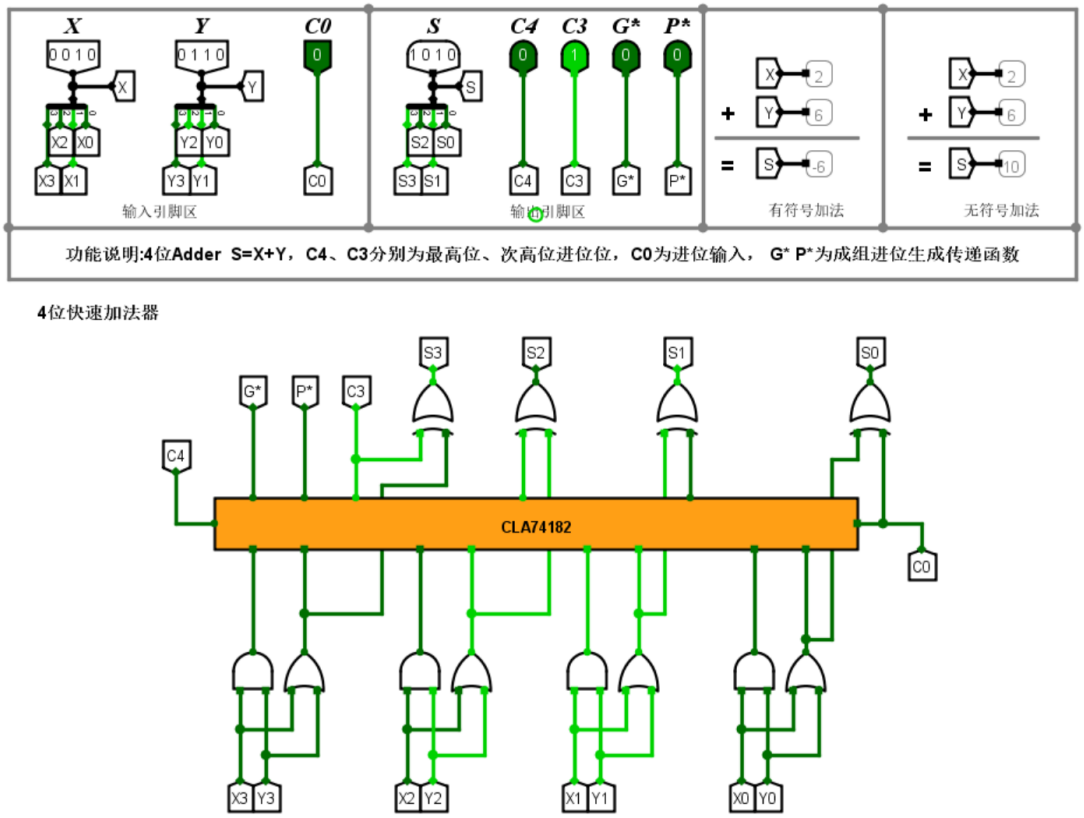

图 25 4 位快速加法器出现的问题

2) 乘法器实验

在做一位原码乘法器的时候，遇到了无限循环，停不下来的情况，此时已经有计数器的限制，但一直停不下来，如图 26 所示。

然后初步检查电路没啥问题，就去仔细看各个引脚是否正确，尤其是仔细看了计数器的引脚，发现使能端接错了，然后重新将 stop 非接到使能端就没问题了。

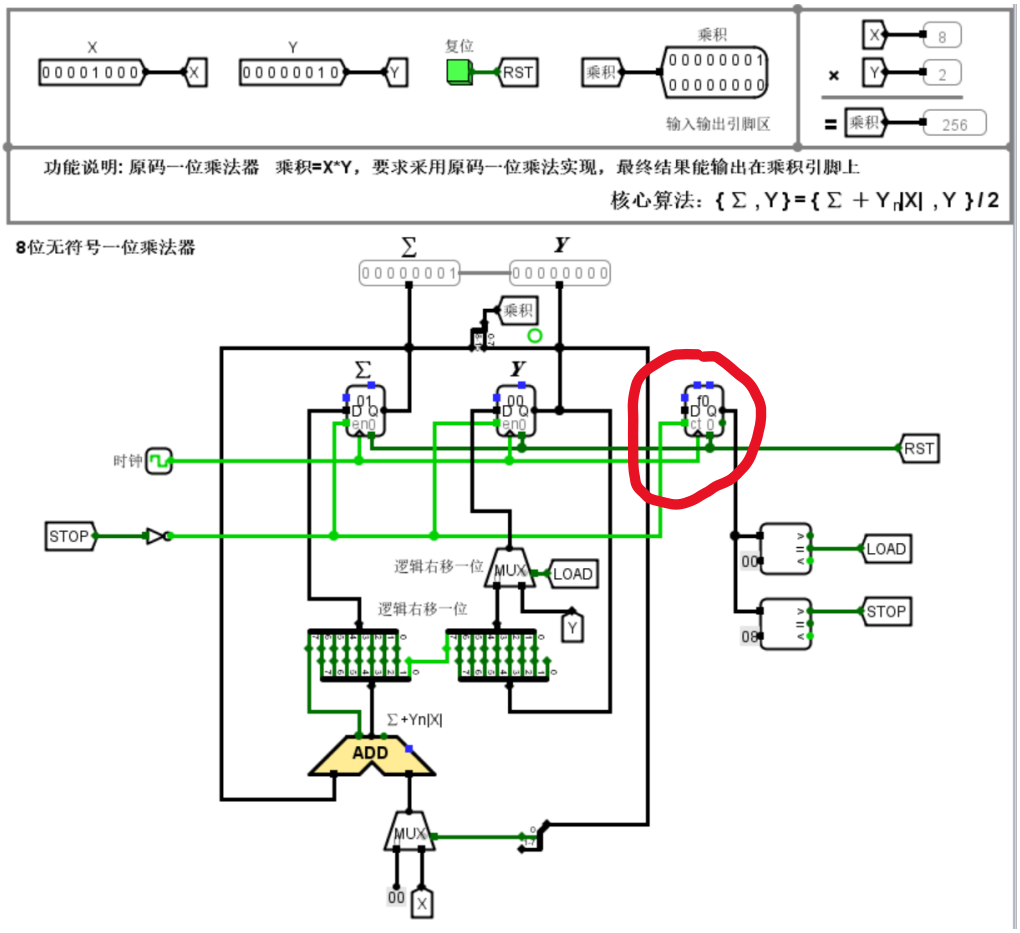

图 26 一位原码计数器实验遇到的问题

3) 算术逻辑运算单元 ALU 实验

实验中评测的分数为 95 分，然后无符号比较那里出问题了，如图 27 所示。于是回到无符号比较那里看，发现比较器的型号没改，如图 28 所示，改过来就没问题了。

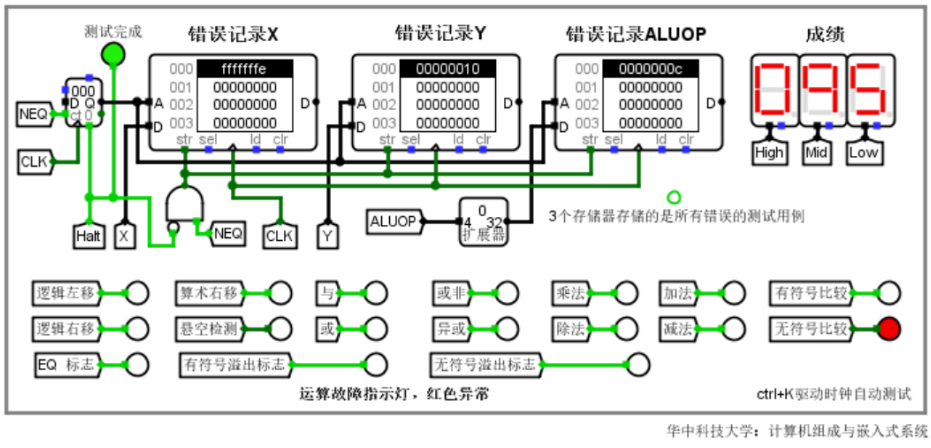

图 27 ALU 实验遇到的问题

图 28 比较器型号的问题

4. 实验总结

运算器实验总体来说没有那么困难，但是还是要注重细节，尤其是前面的加法器，一环扣一环，所以不能有一点错，然后乘法器那里的设计比较巧妙，需要对乘法过程非常熟悉，经过本次实验，我对加减乘除有了一个复习，软件的使用更加熟练了。