《组合逻辑电路设计》实验报告

姓名：许霁烨

学号：211275024

3-8译码器

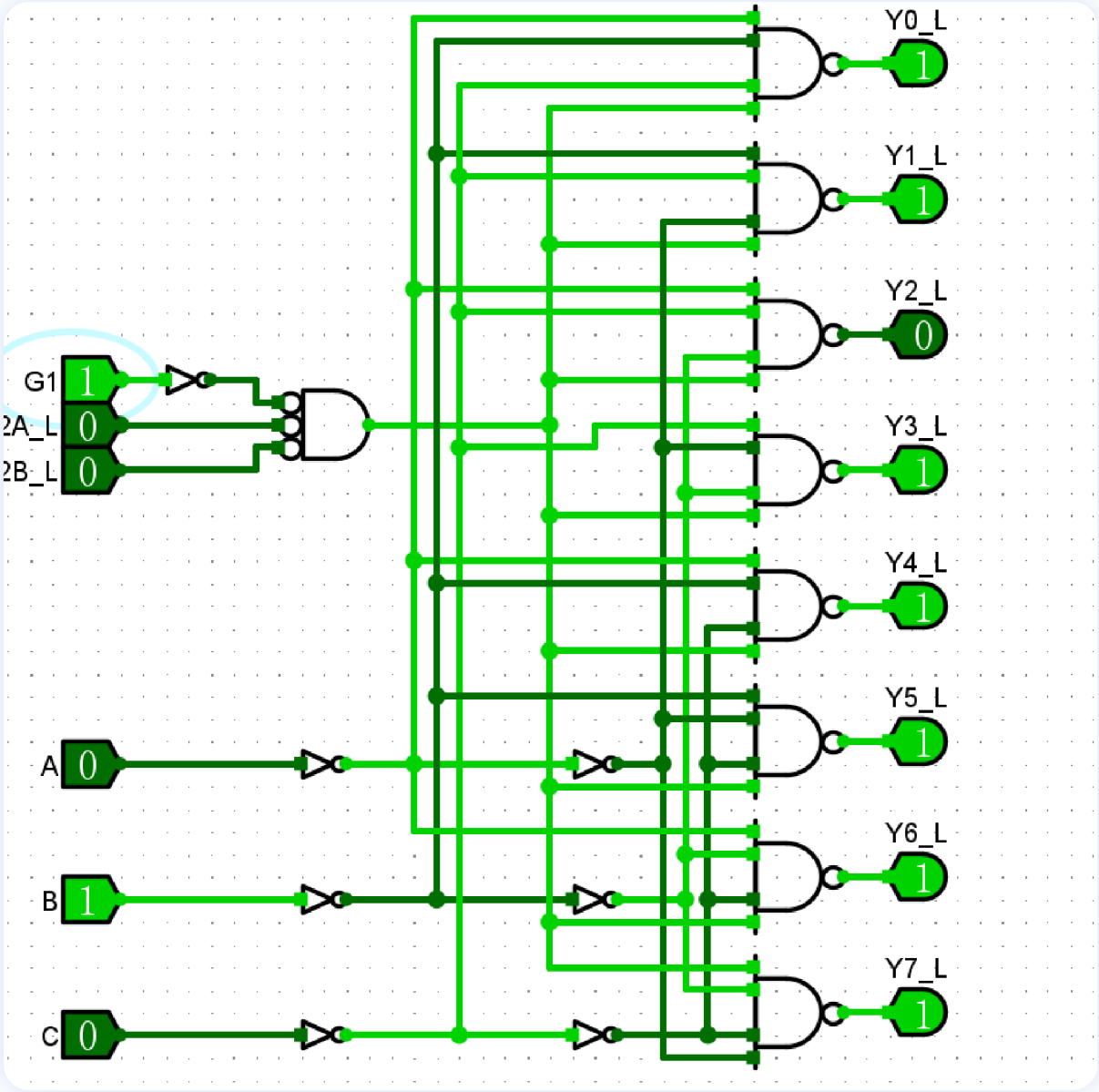
实验整体方案设计

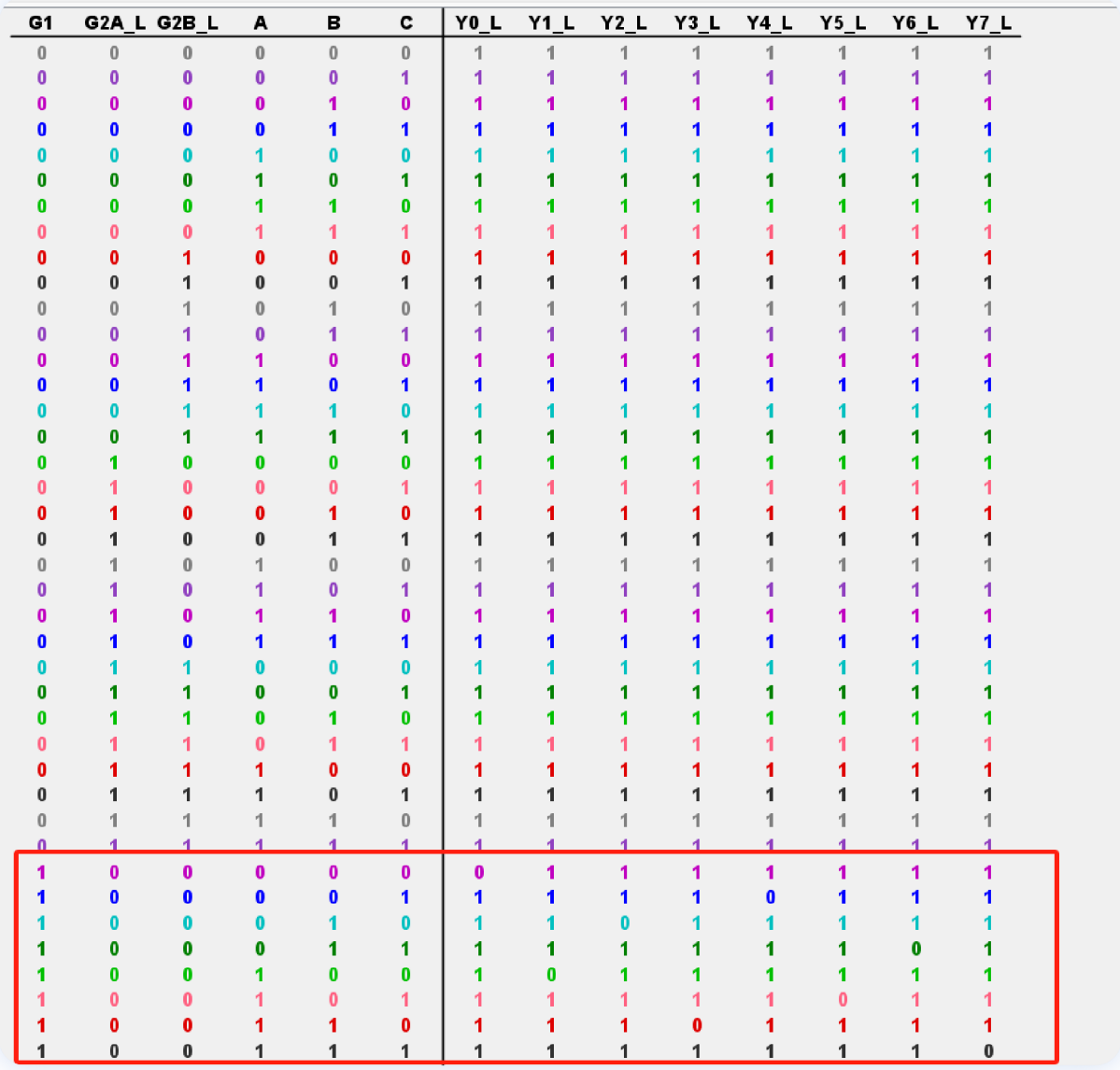
本实验就是复现74X138译码器，主要就是用 `G1,G2A_L,G2B_L` 的非与作为使能端，然后将每一个数值对应的ABC作为输入，同时满足即可

> 比如 `Y0_L` 就需要ABC都是零，以此作为根据设计电路

电路设计与实验结果

电路设计如下，从电路可以看出，当使能端EN为1时，将CBA设置为 `010` （也就是2），`Y2_L` 对应的电路$\overline{\overline{A} \cdot B \cdot \overline{C} \cdot EN}$为0，其余为1，说明译码成功。具体各个的真值表如下：


Y0_L
1
Y1_L
1
Y2_L
0
Y3_L
1
Y4_L
1
Y5_L
1
Y6_L
1
Y7_L
1
G1
1
2A_L
0
2B_L
0
A
0
B
1
C
0

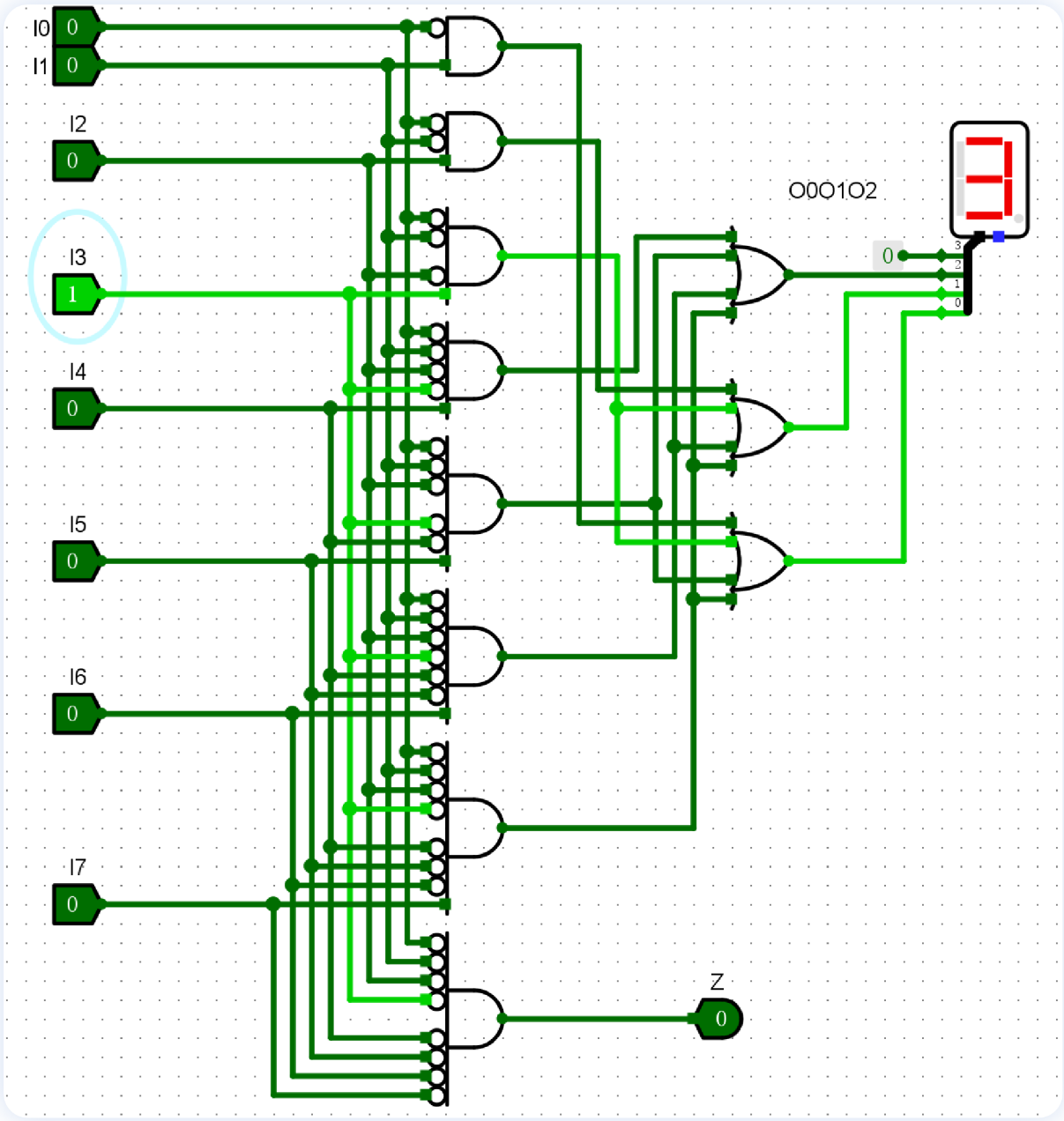


G1	G2A_L	G2B_L	A	B	C	Y0_L	Y1_L	Y2_L	Y3_L	Y4_L	Y5_L	Y6_L	Y7_L
0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	1	1	1	1	1
0	0	1	0	1	0	1	1	1	1	1	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1
0	0	1	1	0	0	1	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1
0	0	1	1	1	0	1	1	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	1	1	1	1	1	1	1	1	1
0	1	0	0	1	0	1	1	1	1	1	1	1	1
0	1	0	0	1	1	1	1	1	1	1	1	1	1
0	1	0	1	0	0	1	1	1	1	1	1	1	1
0	1	0	1	0	1	1	1	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1	1	1	1
0	1	0	1	1	1	1	1	1	1	1	1	1	1
0	1	1	0	0	0	1	1	1	1	1	1	1	1
0	1	1	0	0	1	1	1	1	1	1	1	1	1
0	1	1	0	1	0	1	1	1	1	1	1	1	1
0	1	1	0	1	1	1	1	1	1	1	1	1	1
0	1	1	1	0	0	1	1	1	1	1	1	1	1
0	1	1	1	0	1	1	1	1	1	1	1	1	1
0	1	1	1	1	0	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	0	1	1	1
1	0	0	0	1	0	1	1	0	1	1	1	1	1
1	0	0	0	1	1	1	1	1	1	1	1	0	1
1	0	0	1	0	0	1	0	1	1	1	1	1	1
1	0	0	1	0	1	1	1	1	1	1	0	1	1
1	0	0	1	1	0	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	1	1	1	1	1	1	0

8-3编码器

实验整体方案设计

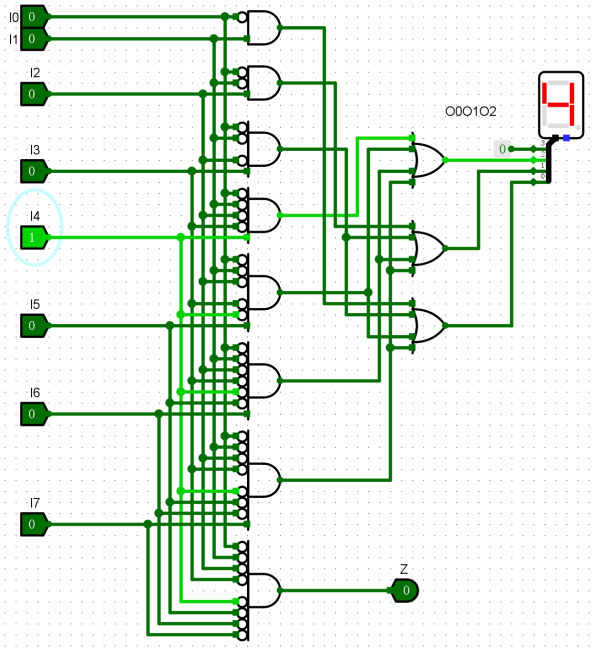
本实验较为简单，就按照书上的8-3优先级编码器实现即可。这里还多设计了一个表示$I_1I_2I_3I_4I_5I_6I_7$全0的标志位Z,如果全0则Z输出为1



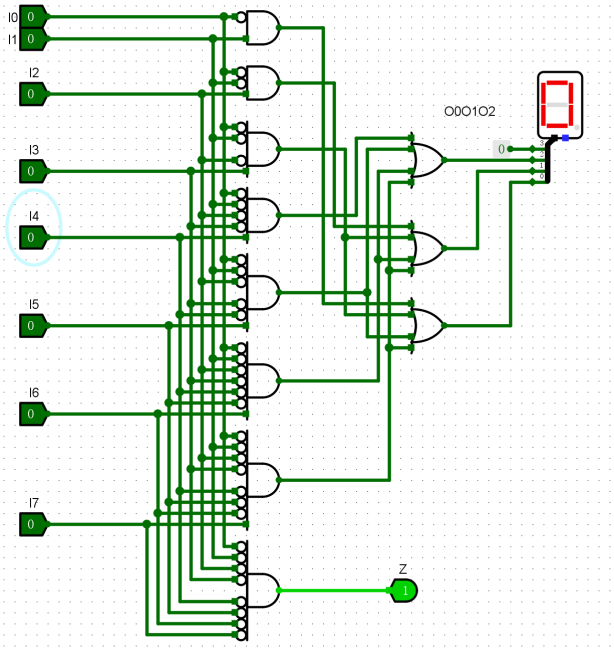
电路设计与实验结果

下面举几个例子：

输入为4（$I_4=1$，其余为0）：



输入全0（Z输出为1）：



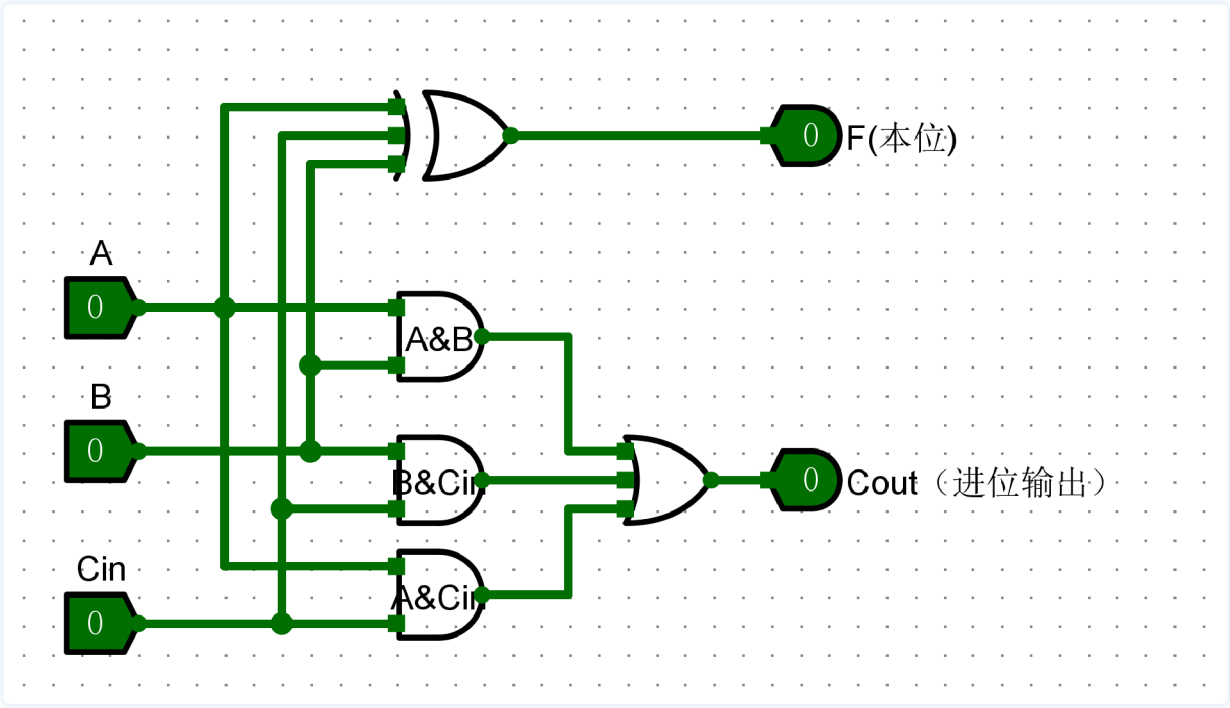
4位加减法器

实验整体方案设计

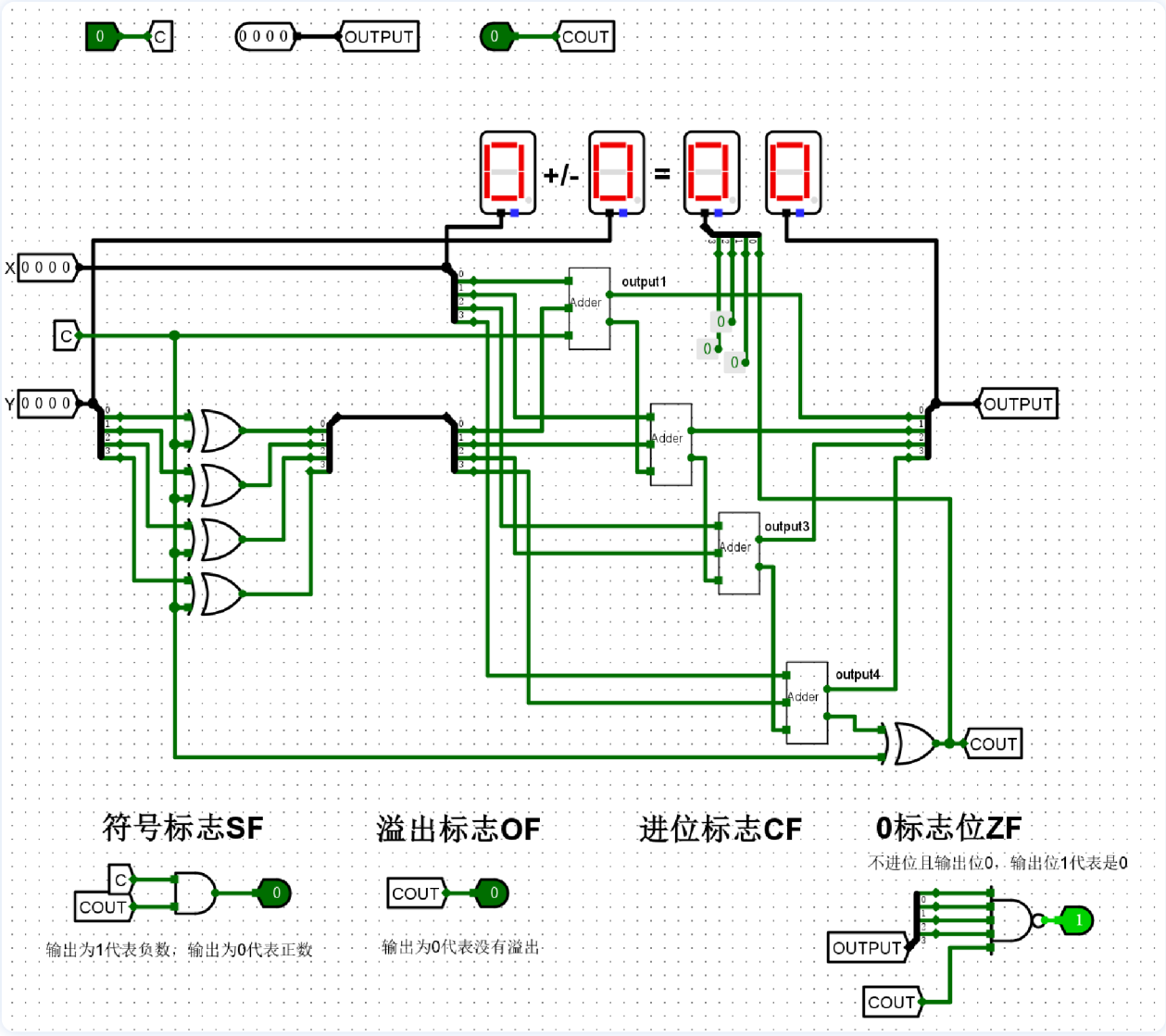
- 加法实现：由于四位加法器，所以我们需要四组**全加器**，其中最低位的cin是0，最高位的cout代表是否溢出。然后其余的cout作为更高位的cin即可
- 减法实现：如果要实现减法，显然不能设计一个减法器，这里我们使用**补码**实现，即$A-B$等价于$A+B_{补码}$，因为补码是**各位取反，最低位加1**，这里我们就先各位取反，然后最低位加的1就以最低位的cin输入即可。

电路设计与实验结果

全加器设计如下：

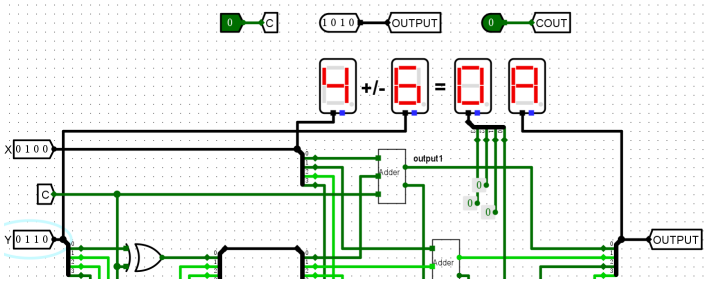


电路设计如下：

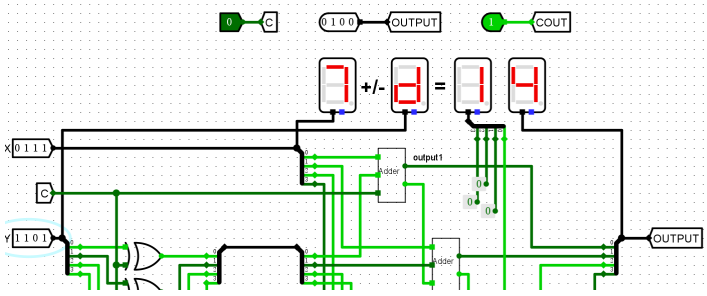


下面展示相关结果：

- 加法（不进位）

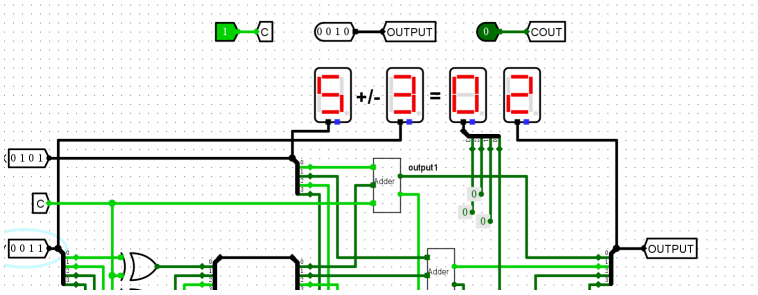


- 加法（进位）

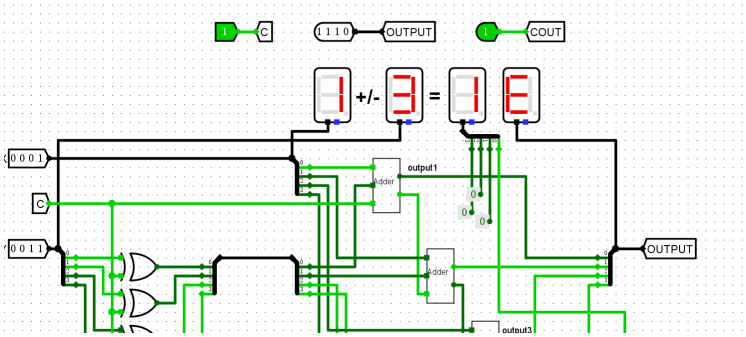


> 7+d(13)=20，在16进制中展示的就是1,4，其中高位的“1”是从cout中得到的

- 减法（结果为正）



- 减法（结果为负数）



> 1-3得到的时-2，这里是让前面那个“1”作为符号位，其值为1代表是负数，后面的E是2的补码

遇到的问题

最开始没想明白取补码的那个“末尾加1”怎么实现，后来想到当是减法的时候C正好为1，正好就可以作为最低位加法时的cin，就正好可以实现这个功能。

汉明码校验电路

实验整体方案设计

汉明码的主要逻辑就是设置多个校验位P_i代表在序列中排序的二进制从低到高的第i位为1的所有数据进行校验（这里为偶校验，也就是校验码和对应的原码的**1的个数为偶数**）

序号 / 分组 含义	1	2	3	4	5	6	7	故障字	正确	出错位
	P_1	P_2	M_1	P_3	M_2	M_3	M_4			1 2 3 4 5 6 7
第3组				√	√	√	√	S_3	0	0 0 0 1 1 1 1
第2组		√	√			√	√	S_2	0	0 1 1 0 0 1 1
第1组	√		√		√		√	S_1	0	1 0 1 0 1 0 1

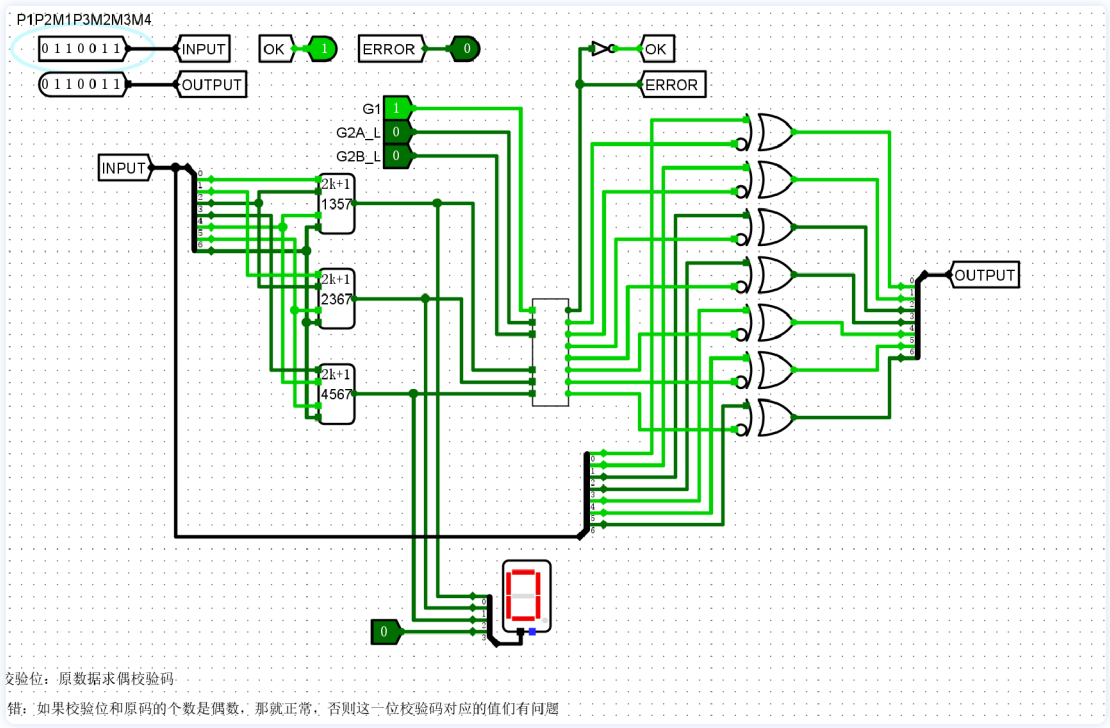
例子如下：

$$P_1P_2M_1P_3M_2M_3M_4$$

那么P1的值就需要让P1M1M2M4中1的个数为偶数，其余校验位依次类推。

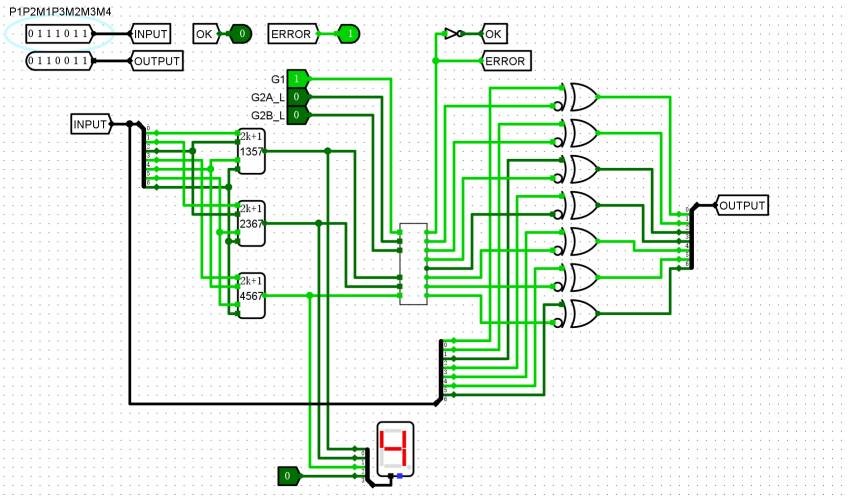
根据此原理，就可以对要检验的汉明码进行检验，根据P_1, P_2, P_3与对应码位的偶校验结果就能定位到，例如P1和P2的校验失败，P3的校验成功，那么就能定位到是"011"位出错。由此可以设计电路。

电路设计与实验结果



测试，正确的输入应该是 `0110011`：

将第四位改错，即输入为 `0111011`：



> 可以看出，这里能显示出是第四位错误了，同时output也能正确输出改正后的结果

遇到的问题

这里需要让错误的地方展示1，才能在后续的译码器找到错误的地方，因此在校验的时候需要用奇校验而不是偶校验。

8位桶形移位器

实验整体方案设计

这里设计主要是基于0-7的任何一个数都可以表示为：

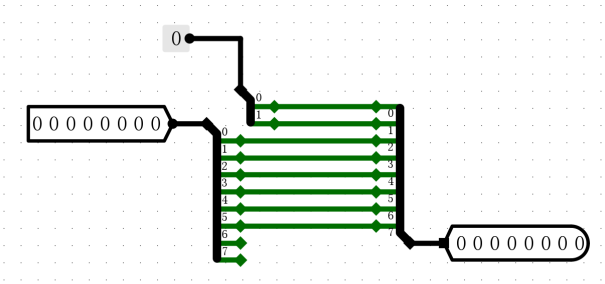
$$P = i \times 2^0 + j \times 2^1 + k \times 2^2$$

因此我们可以设计多个函数，实现左/右移n位（n=1,2,4）

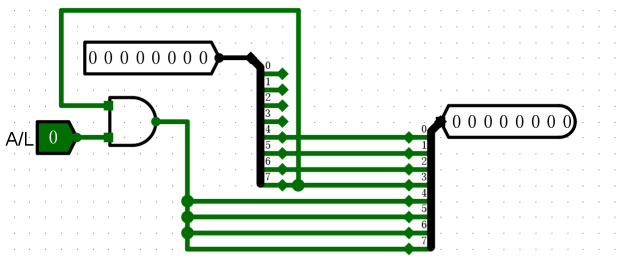
然后对一个序列，假如要移动5位，我们就可以让他先移动1位，再将结果移动4位即可

电路设计与实验结果

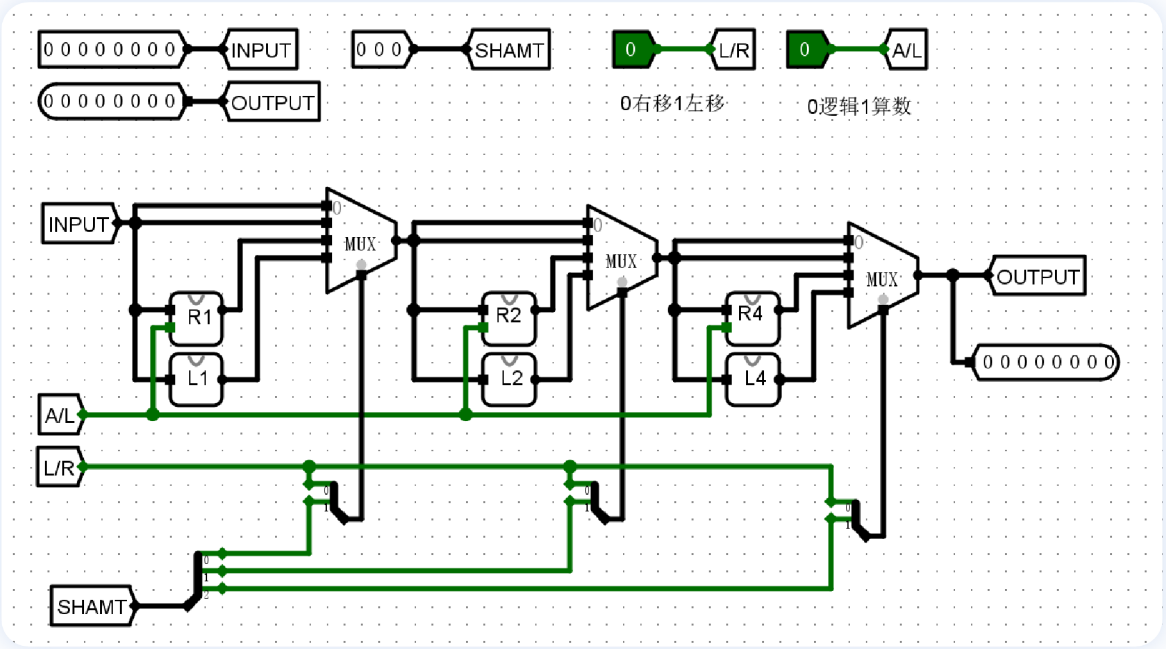
- 移位电路：
  - L2（左移2位）



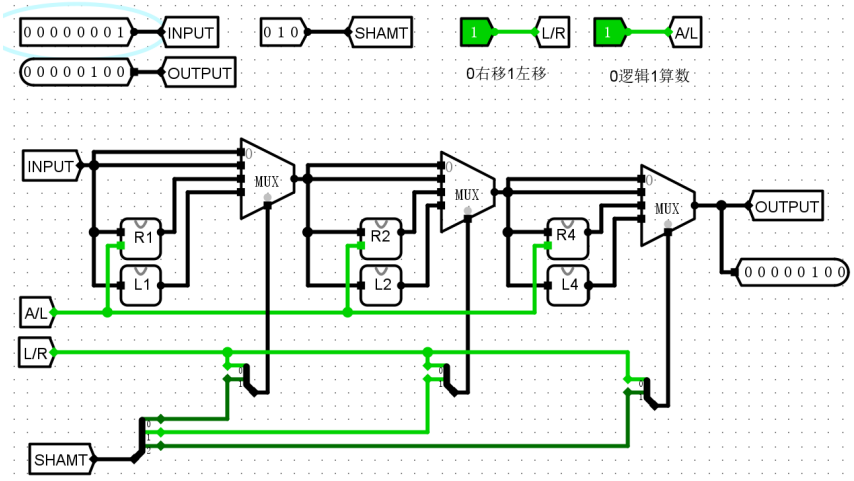
  - R4（右移4位，这里需要输入一个A/L用于区分算数移动还是逻辑移动）：



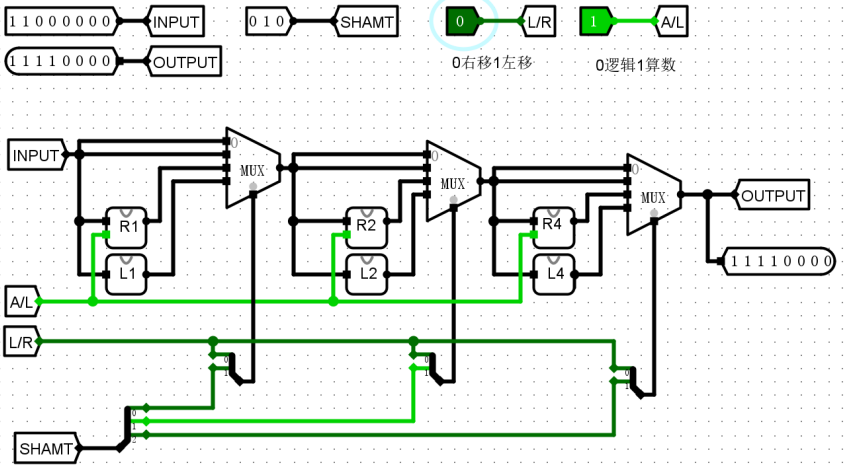
- 总体电路：



- 实验结果（几个例子）：
  - 左移2位：



  - 算数右移3位：



思考题

思考题1：

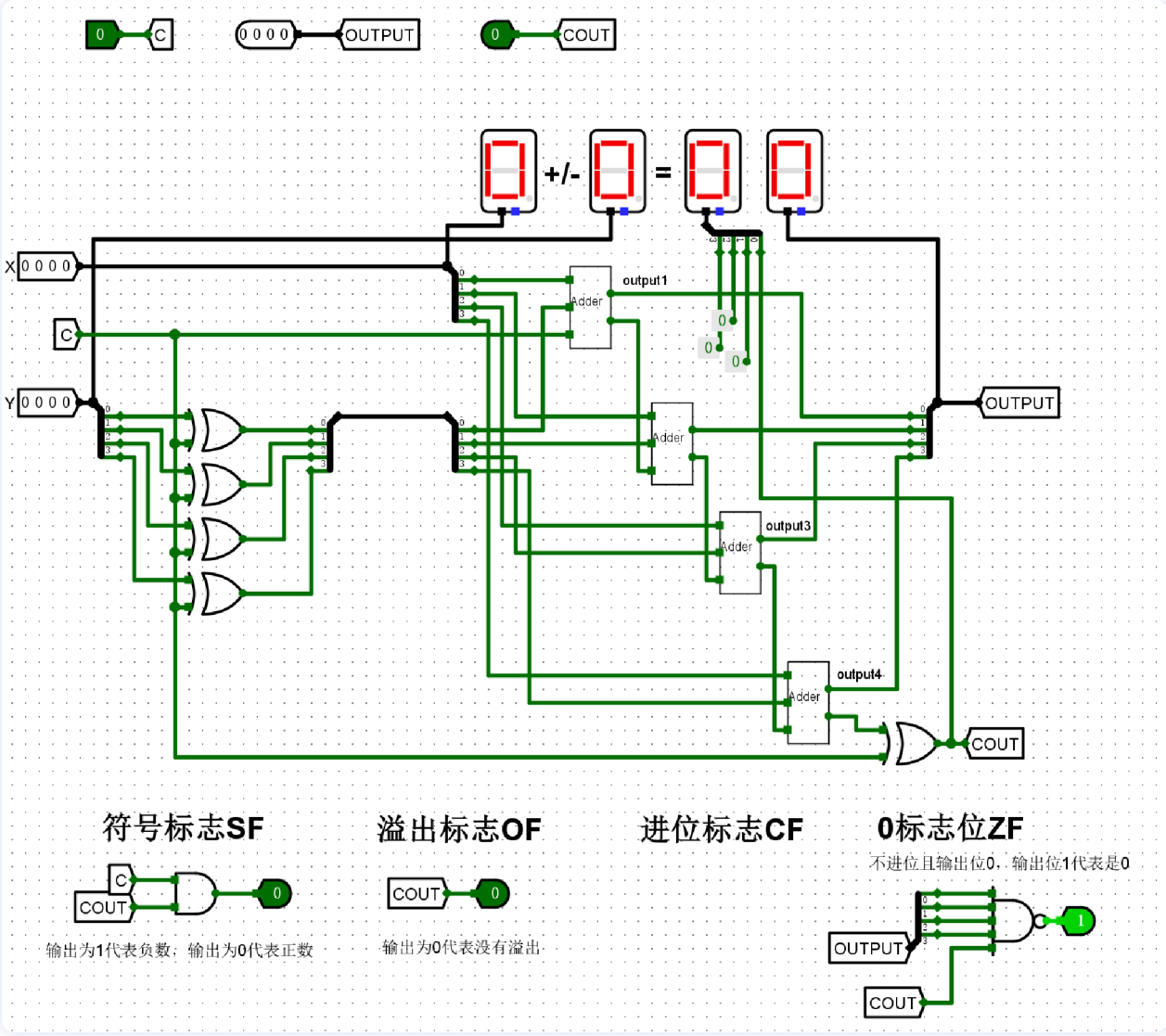
修改实验中的加法器电路，生成进位标志 CF、溢出标志 OF、符号标志 SF 和结果为零标志位 ZF。

设计其实比较简单：

- 进位标志和溢出标志：可以看cout，如果是1则说明发生了溢出（加法结果超过了16或者减法结果为负数）
- 标志符号：当且仅当输入C=1&&COUT=1时是负数，其余时候是正数
- 0标志位：结果为0且标志位为0时才是0

> 如果只是结果为0，那么当cout=1且output=0时，其代表的是16而不是0，因此还需要特别判断

设计电路如下：

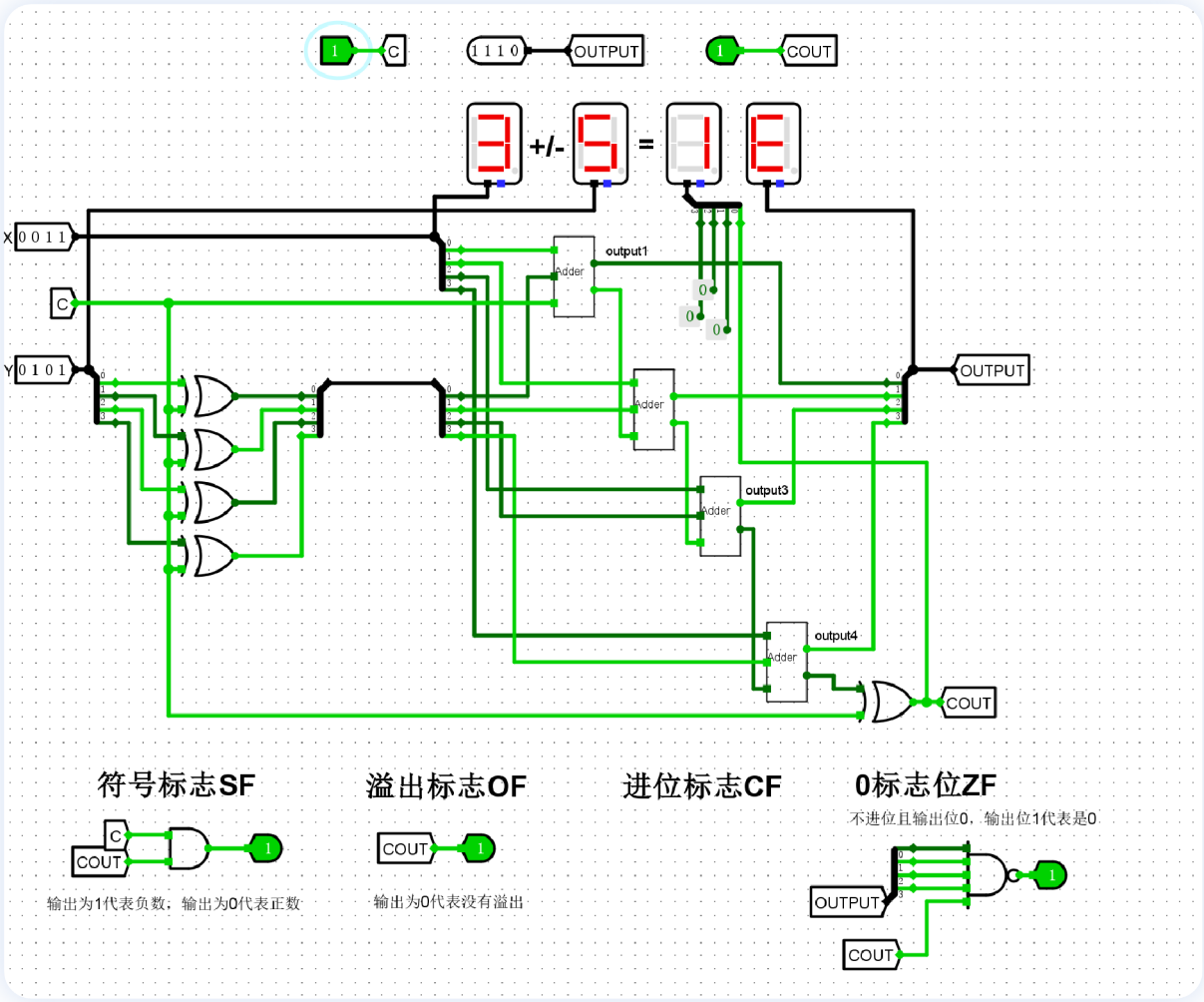


思考题2：

在执行比较指令时，通常使用减法运算后，判断标志位的方式来实现，试通过上述加法器实验举例说明判别的方法。

这题比较简单，例如要比较X和Y的大小关系，那么可以用加减法器求X-Y，如果小于0的话就能说明X<Y，反之则是X>Y

举例如下：例如比较 `X=3,Y=5` 的大小



可以看见，结果的表示位是负数，也就是说X<Y，证明了我们前面说的理论

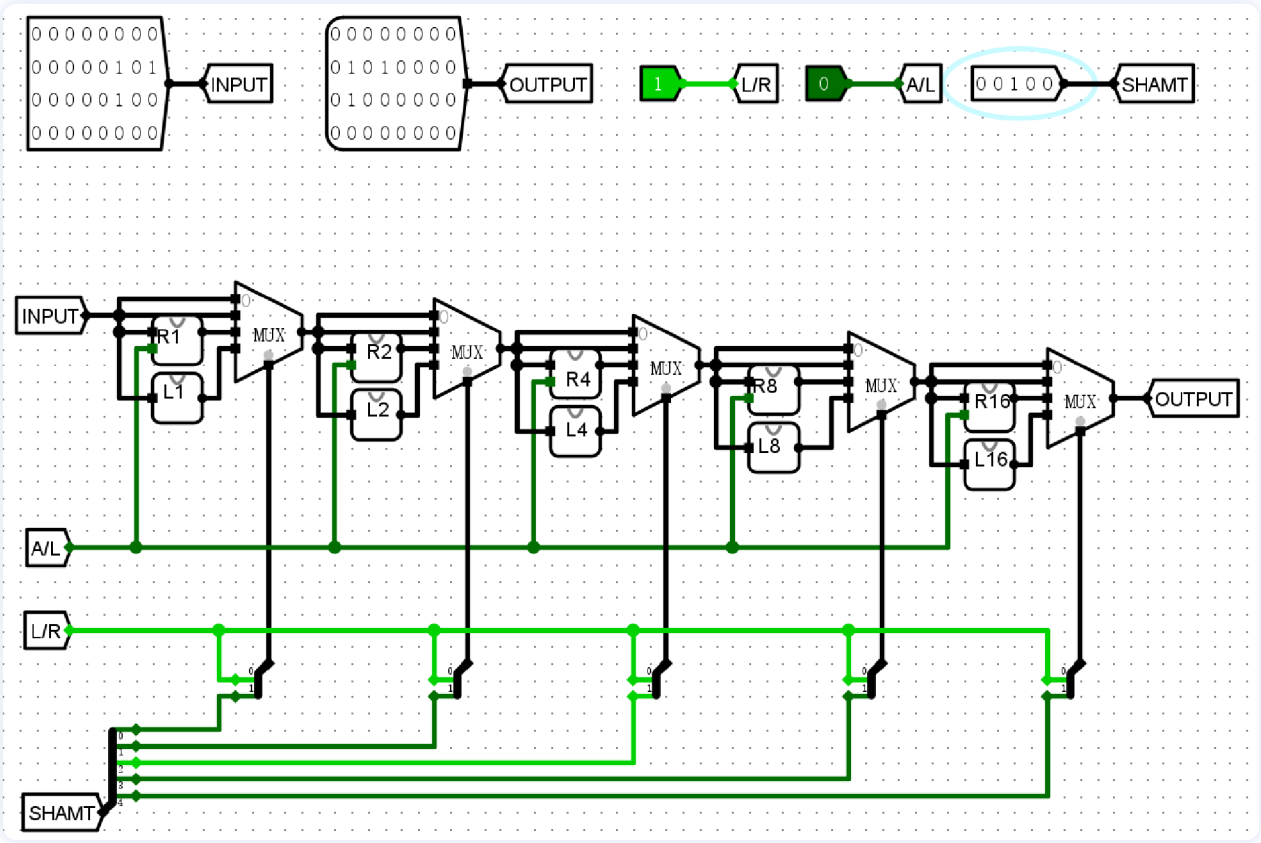
思考题3：

如何使用 8 位桶形移位器扩展到 32 位桶形移位器。

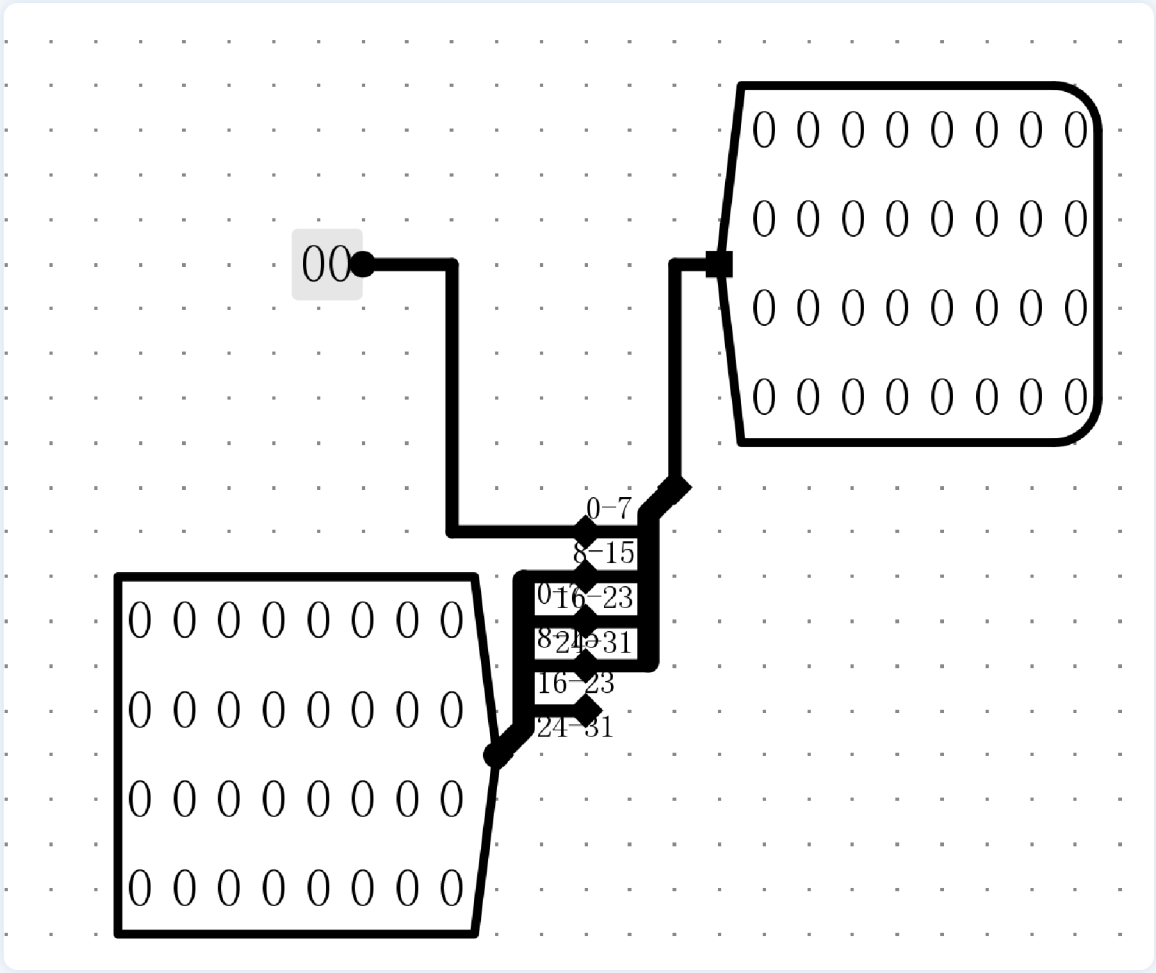
我觉得用8位桶形移位器比较麻烦，就自己重新设计了，具体思路和8位桶形移动器相同，只是再多个移动8位和移动16位的即可。其中 `L<X>/R<X>` 分别代表向左或者向右移动X位。这里的移动32位同样可以表达为：

$$P = i \times 2^0 + j \times 2^1 + k \times 2^2 + x \times 2^3 + y \times 2^4$$

因此可以通过五级的移位电路实现32位数据左移/右移0-31位



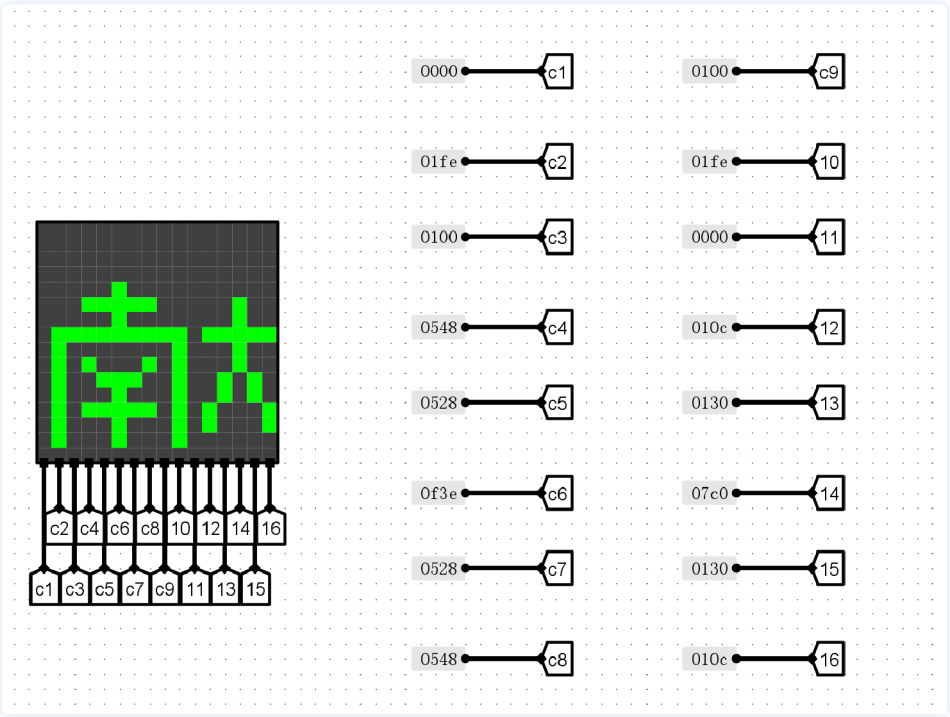
Tips：在移动较大位数时，可以减少分线器端口数，这样可以提高电路简洁程度，示例如下（L8左移八位电路）：



思考题4：

本题较为简单，用01序列表示LED灯阵的亮灭与否

> 特别注意，这里要用常量表示，如果用正常的引脚的话关闭打开就会有问题


0000 c1
01fe c2
0100 c3
0548 c4
0528 c5
0f3e c6
0528 c7
0548 c8
0100 c9
01fe 10
0000 11
010c 12
0130 13
07c0 14
0130 15
010c 16
c2 c4 c6 c8 10 12 14 16
c1 c3 c5 c7 c9 11 13 15