**实验3同步时序电路设计实验报告**

姓名: 学号:

**一、实验目的**

1. 掌握时序逻辑电路设计的基本方法。

2. 掌握计数器和移位寄存器的构建方法。

3. 熟悉计数器和移位寄存器的应用。

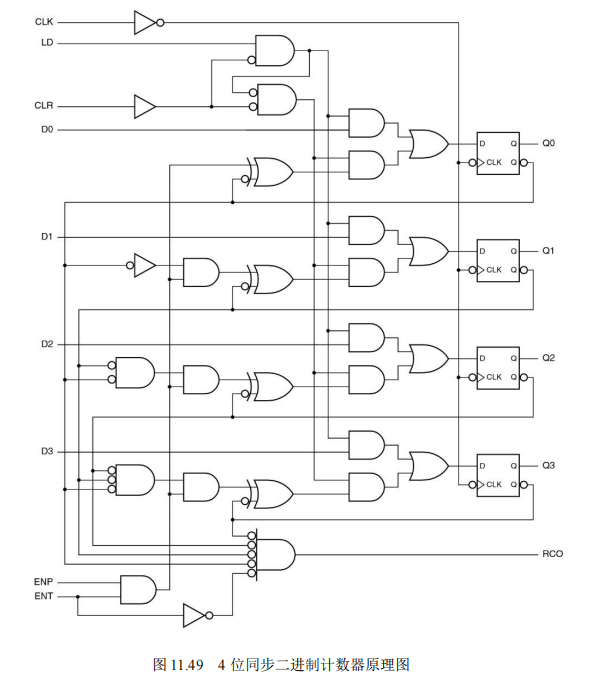
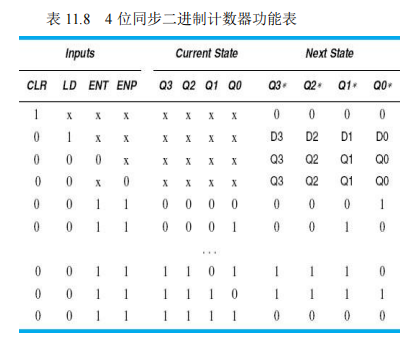
4. 掌握寄存器堆的设计方法。

**二、实验环境**

Logisim：https://github.com/Logisim-Ita/Logisim

**三、实验内容**

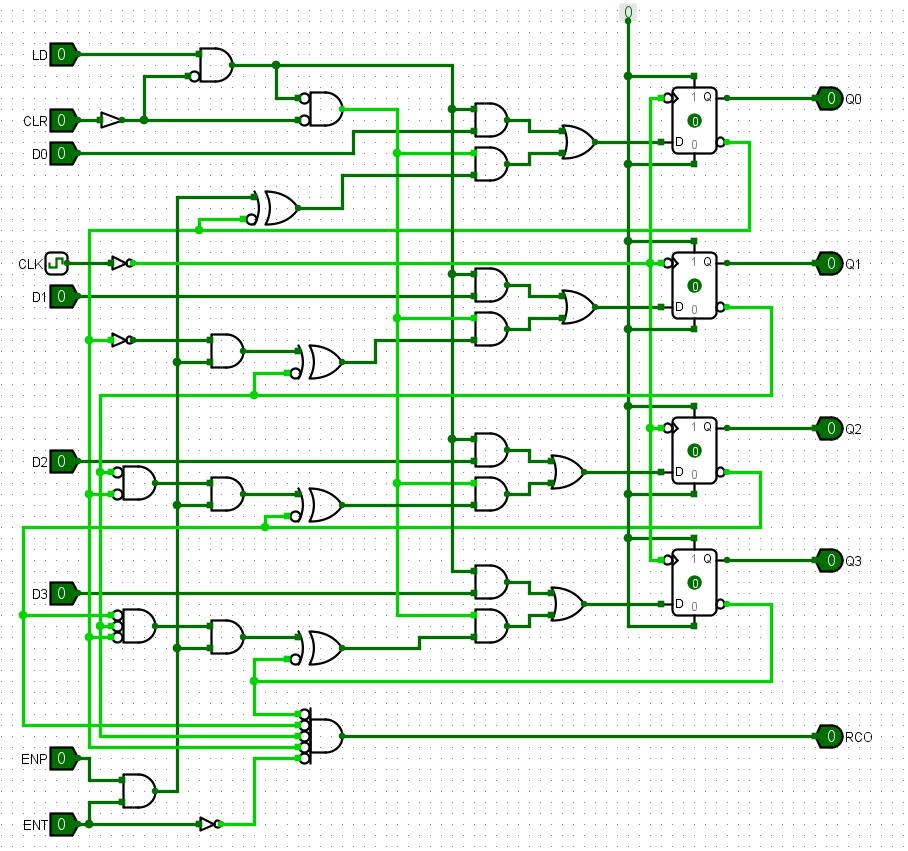
**1. 计数器实验**

根据下表给出的功能表和图示电路原理图构建 4 位同步二进制计数器CNTR4U子电路，利用该子电路和少量门电路，分别通过清零端和置位端各设计一个10进制计数器。 

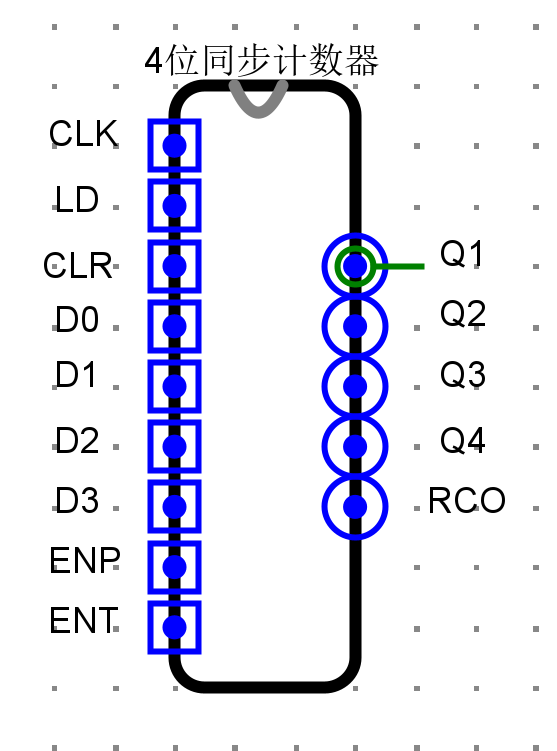
实验步骤：

1. 构建 4 位同步二进制计数器子电路。并封装为子电路，命名CNTR4U

电路如图：

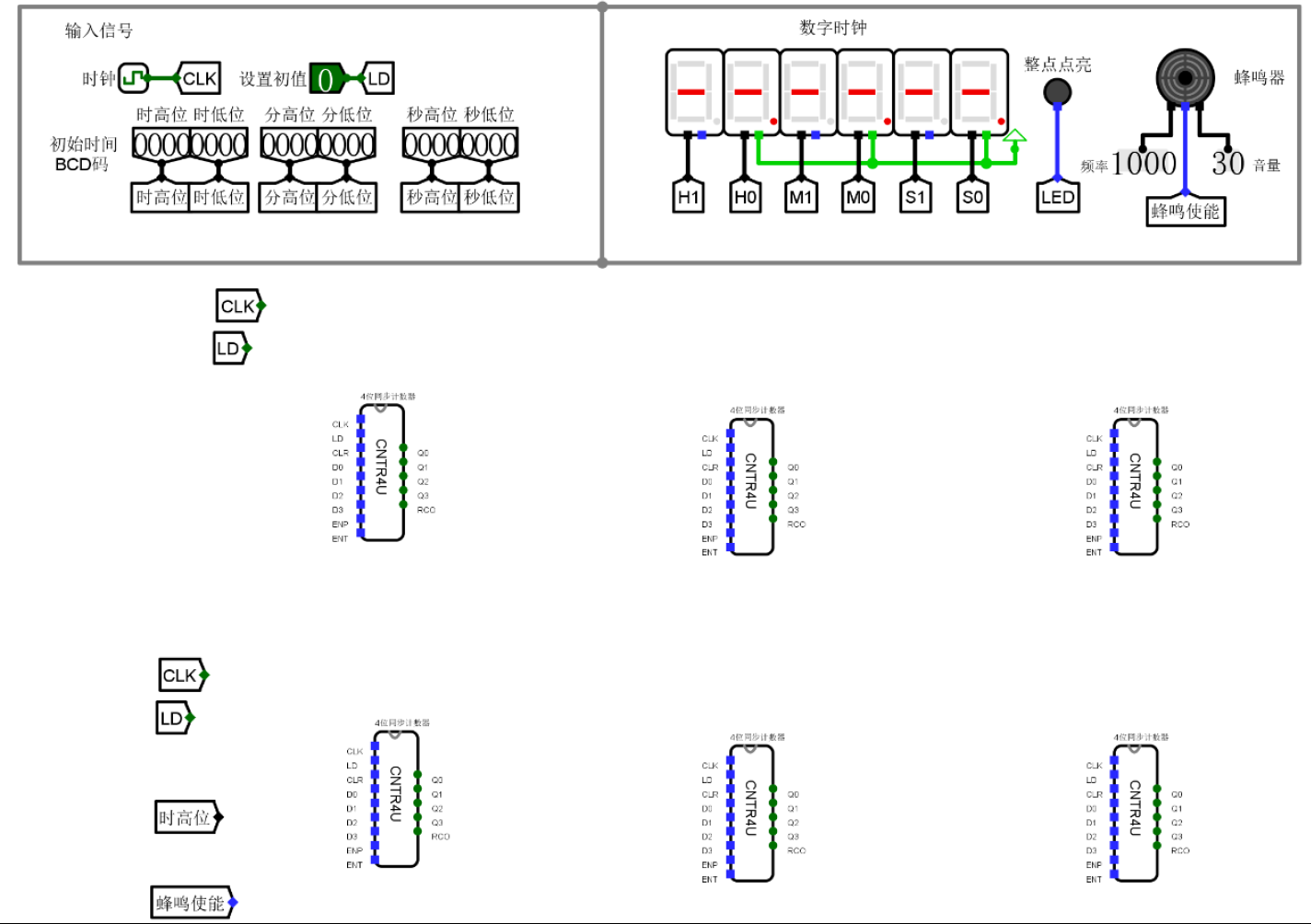


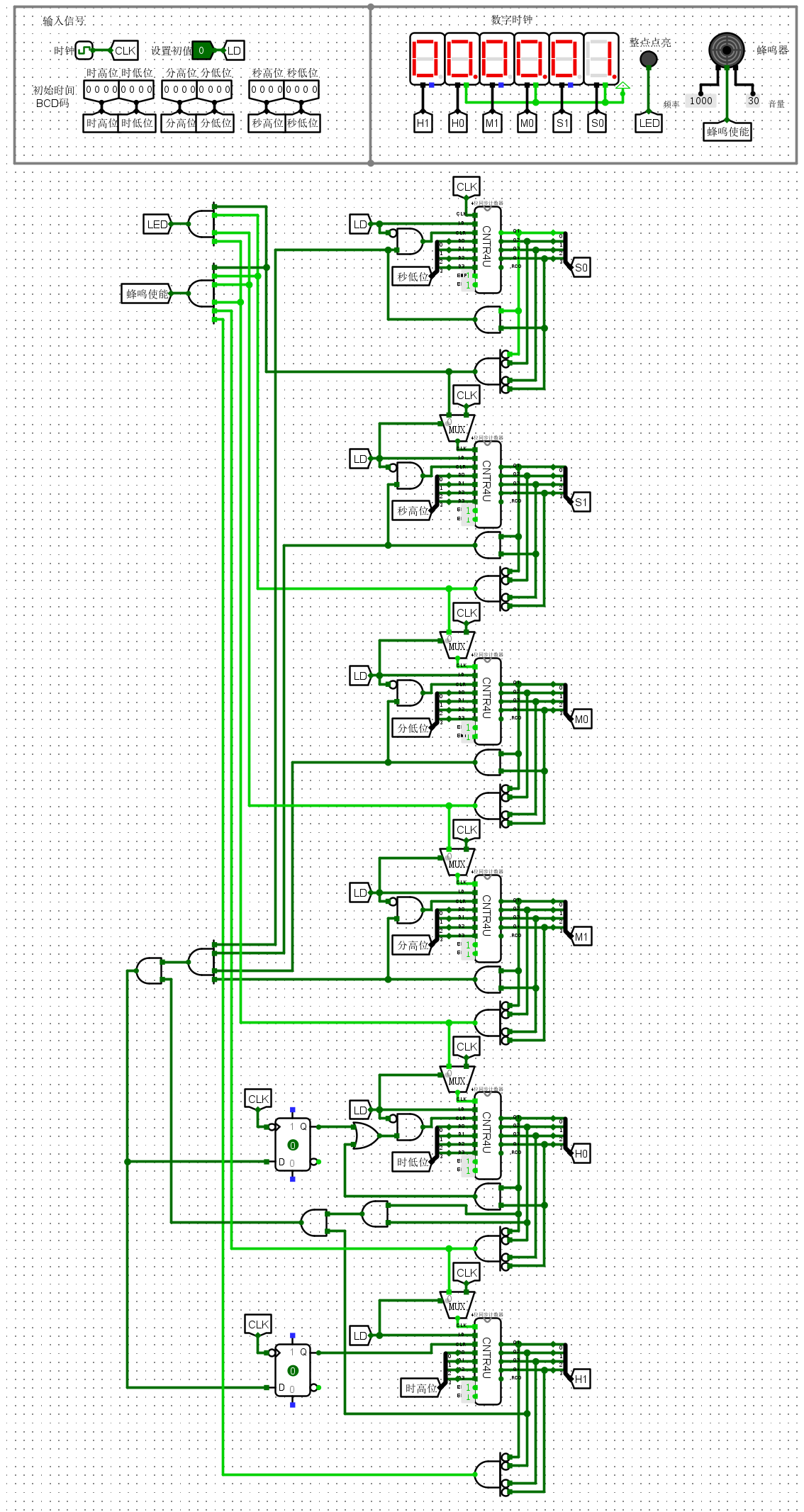
2）按照实验要求修改封装外观如下：



**2、数字时钟实验**

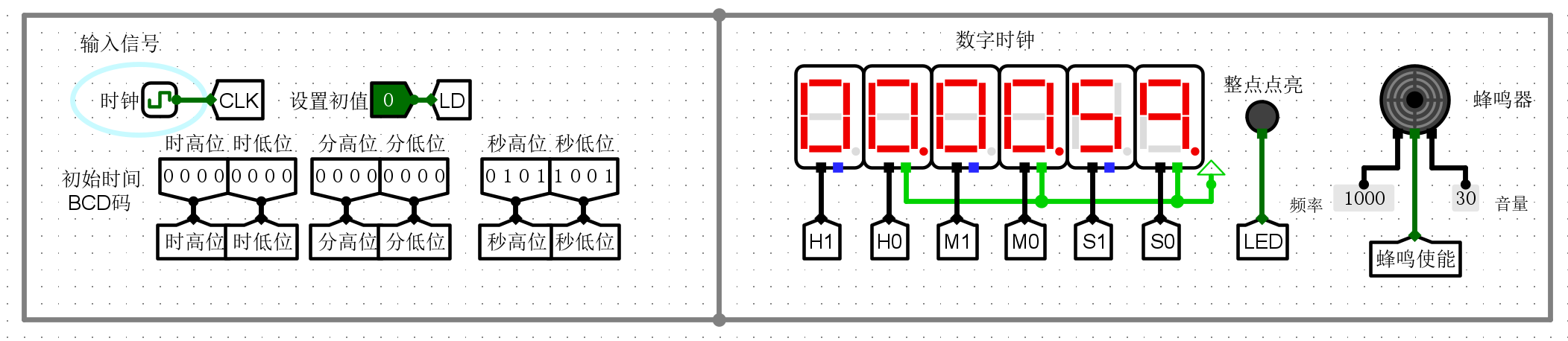
1) 要求：显示时分秒6个BCD码数字，到23:59:59后清零，同时蜂鸣1秒，到整点时点亮LED灯1秒，可设置BCD码格式的初始时间。

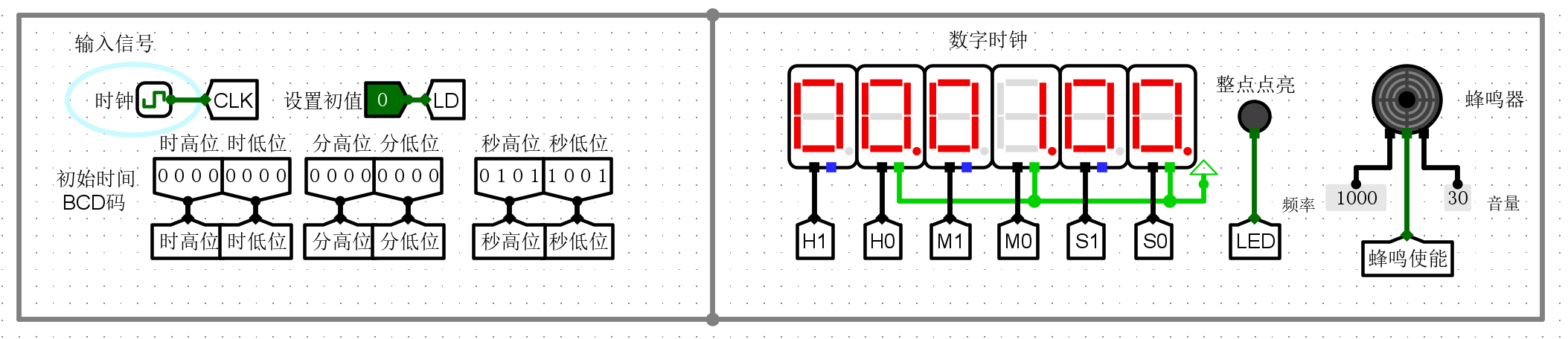
2) 按需取用实验元件并排放如下：

3) 连接元件，电路图如下：

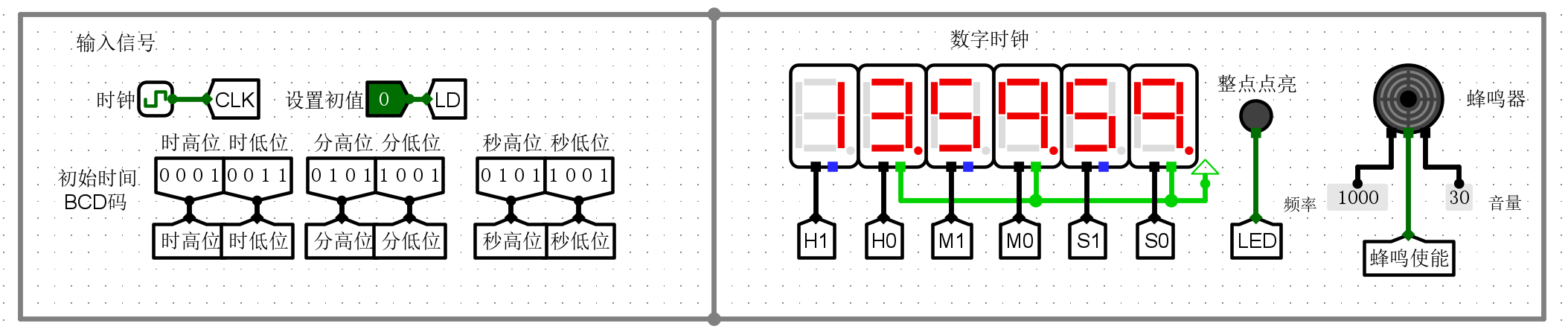
4）仿真测试电路功能：

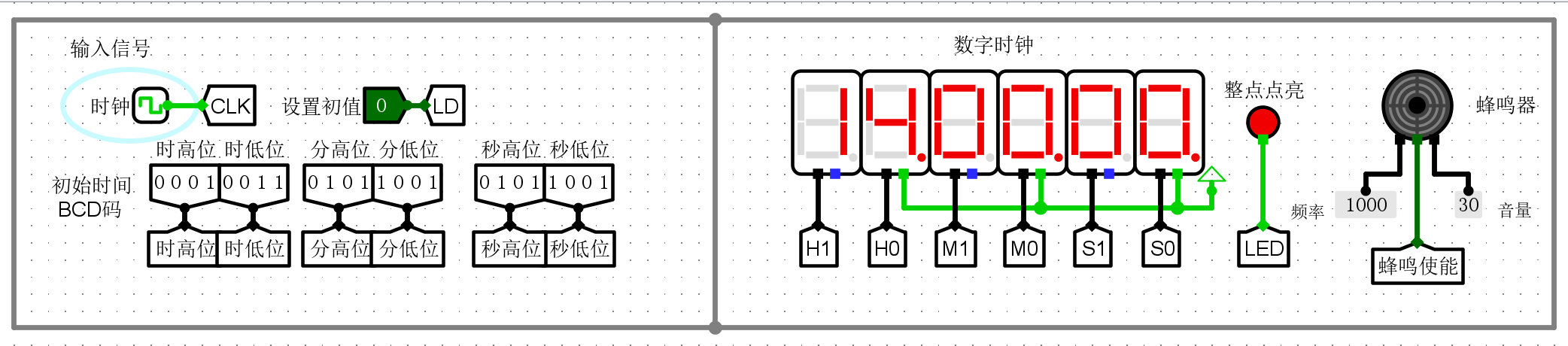
进位：



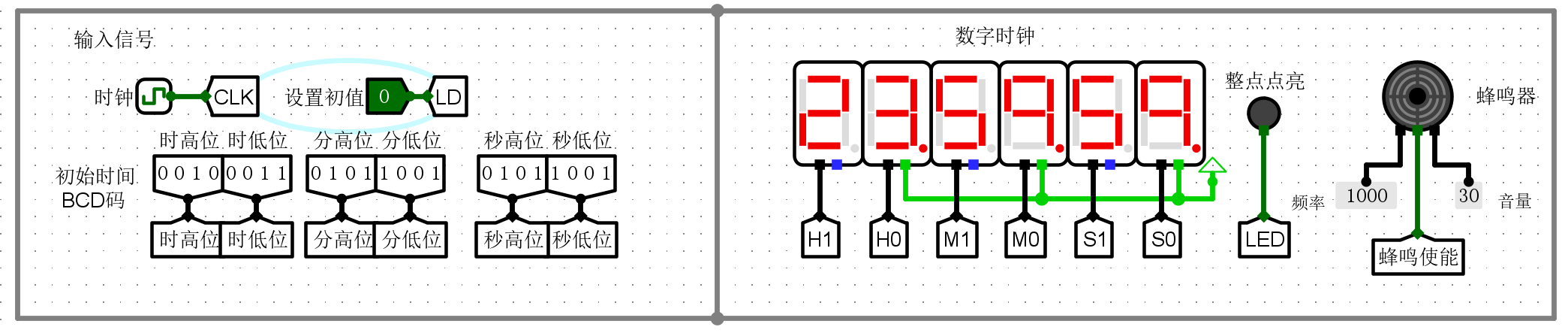


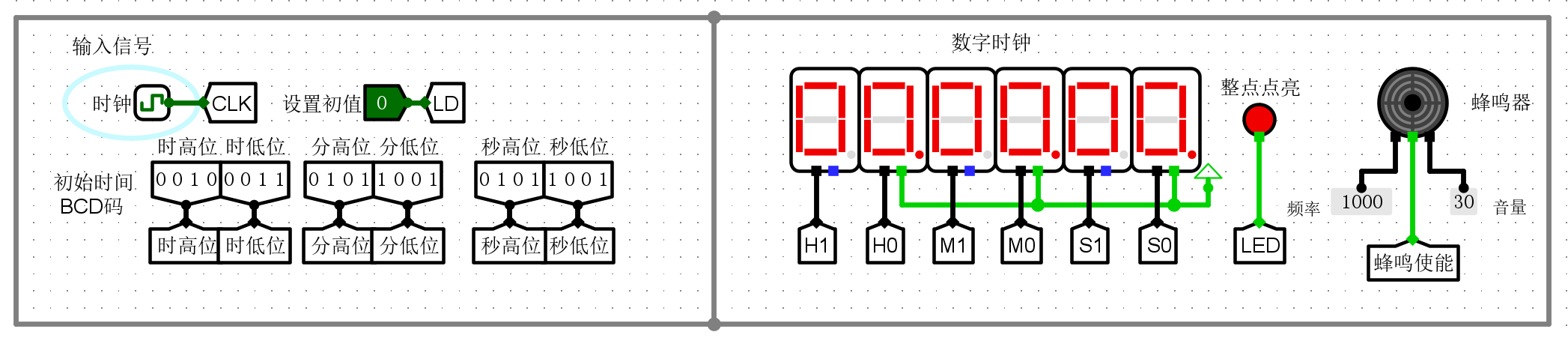
整点亮灯：





24H清零与蜂鸣：

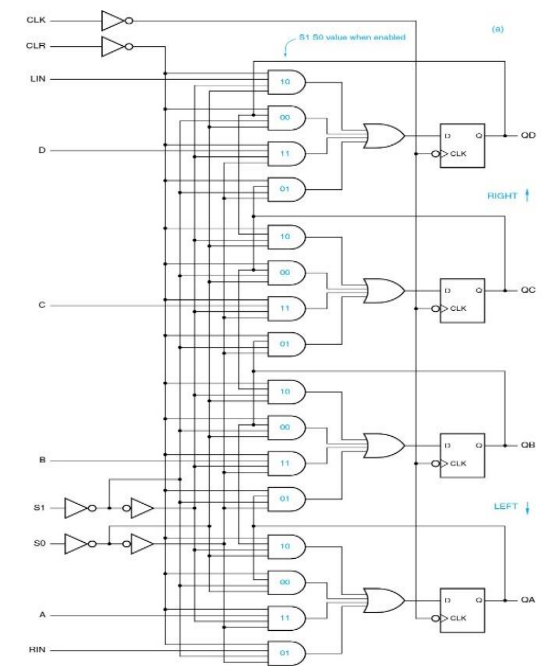
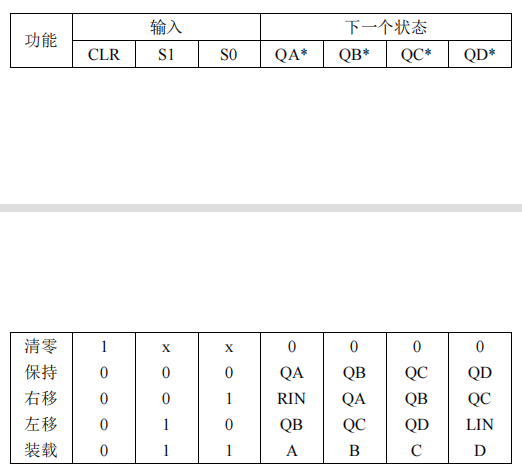




**3.移位寄存器实验**

根据表 11.9 给出的功能描述和图 11.58 给出的电路原理图，构建 4 位通用移位寄存器SHRG4U子电路，利用子电路SHRG4U和少量门电路循环产生15位二进制序列1001 1010

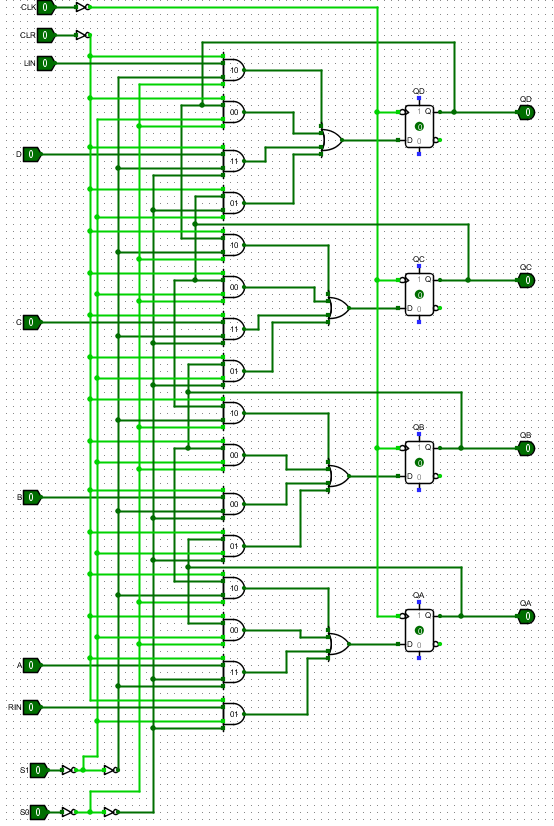
1111 000。



实验步骤：

1） 构建 4 位通用移位寄存器 SHRG4U 子电路。

电路如图：

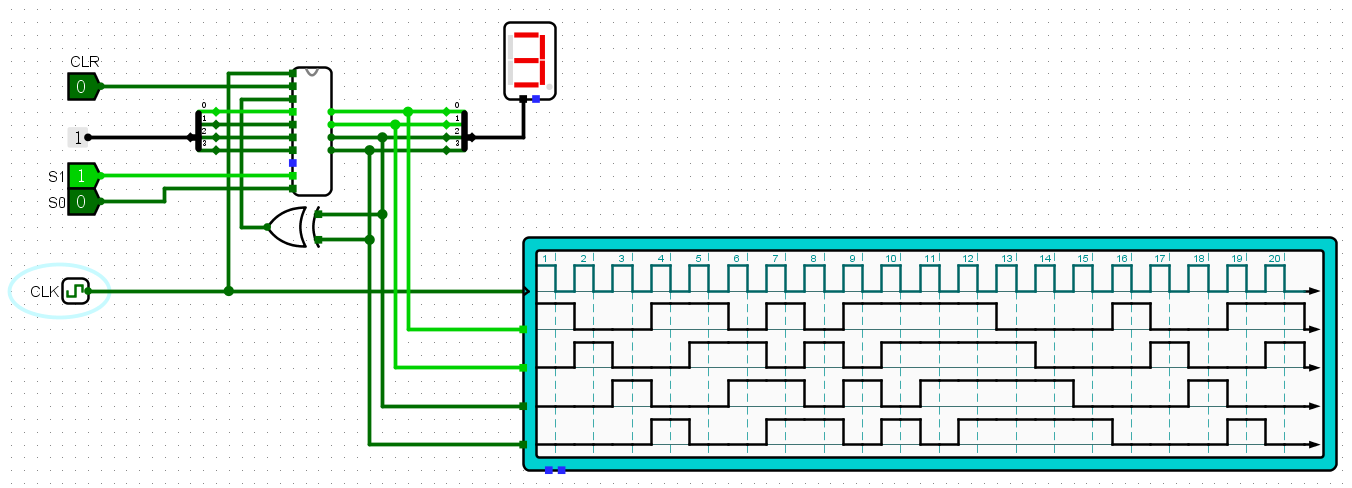


2） 利用该 SHRG4U 子电路和少量门电路设计一个二进制序列生成器，要求生成的二

进制序列为 1001 1010 1111 000。

分析：若采用左移，则反馈方程为 LIN=QA⊕QB。

电路如图：



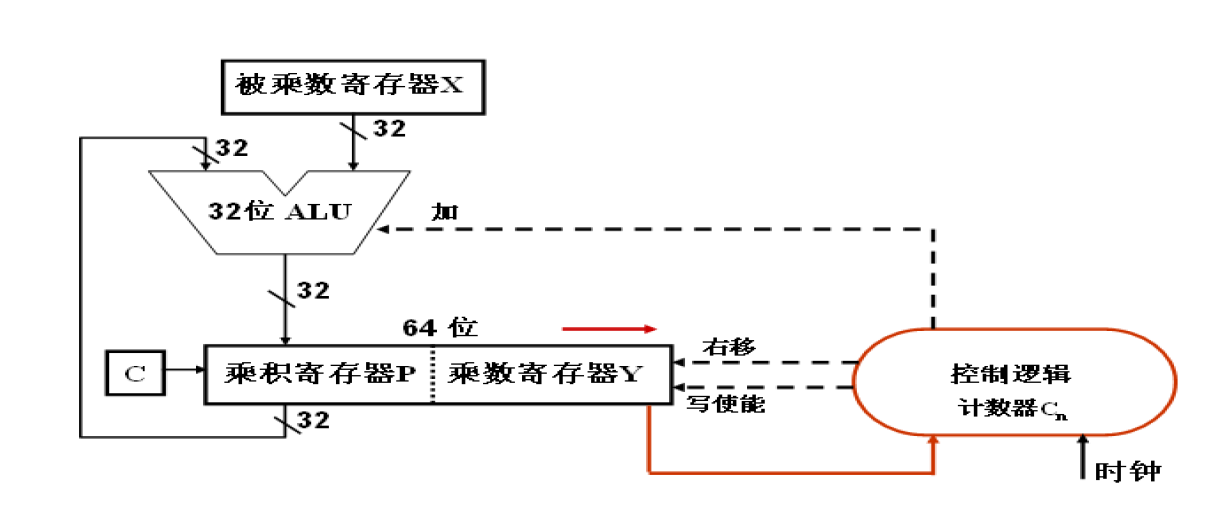
进行仿真检测

数字示波器所显示的是 QD、QC、QB 和 QA 的输出序列，都为 1001 1010 1111 000，前后相差一个时钟周期。16进制数码管按 QA QB QC QD 构成的二进制数值进行循环显示，最初输出为二进制数值 0001 对应的十六进制数 1，输出产生的伪随机数序列为 1, 2, 4, 9, 3, 6, d, a, 5, b, 7, f, e, c, 8, 1, 2, …。符合实验要求

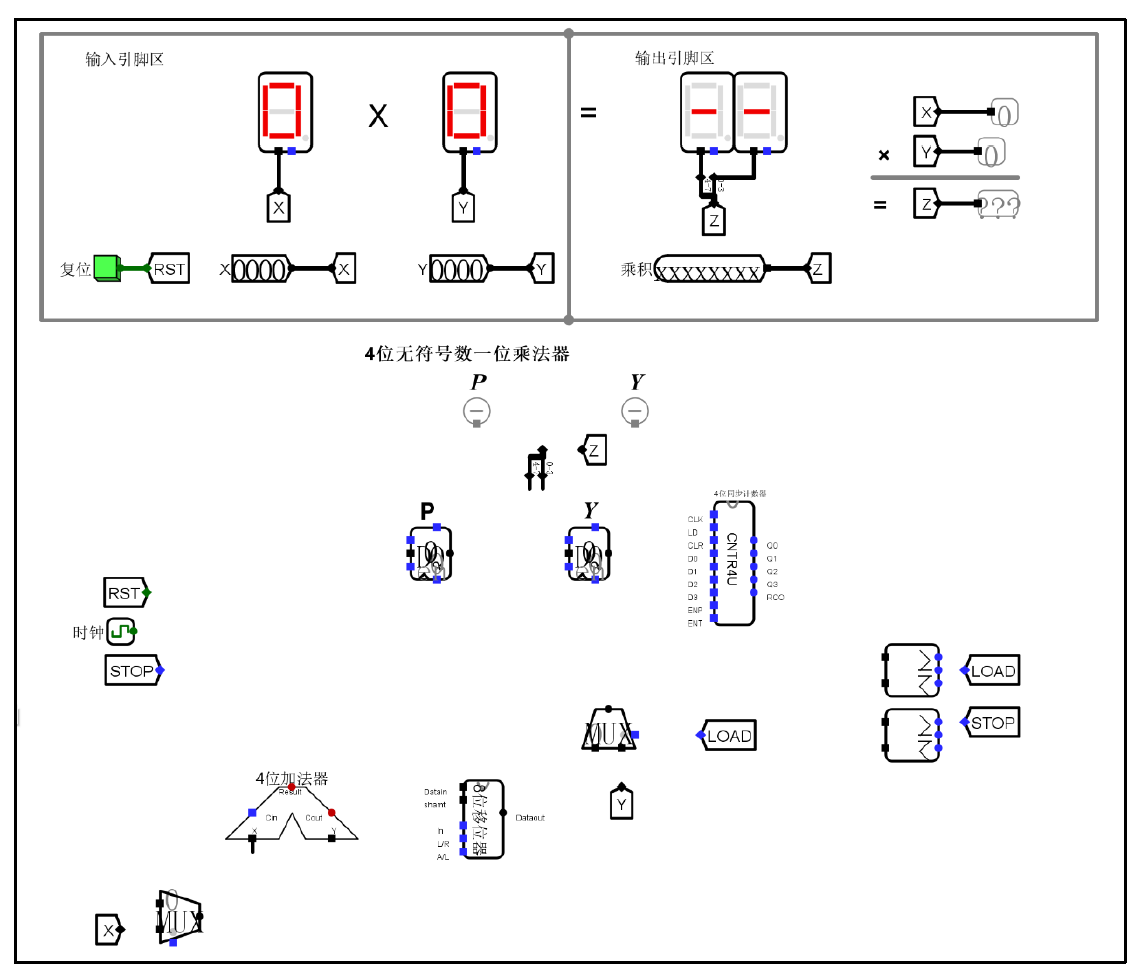
**4.无符号乘法器实验**

1）实验要求**：**实验将实现两个四位二进制无符号数相乘的功能，并通过数码管将其转换成十六进制显示出来。

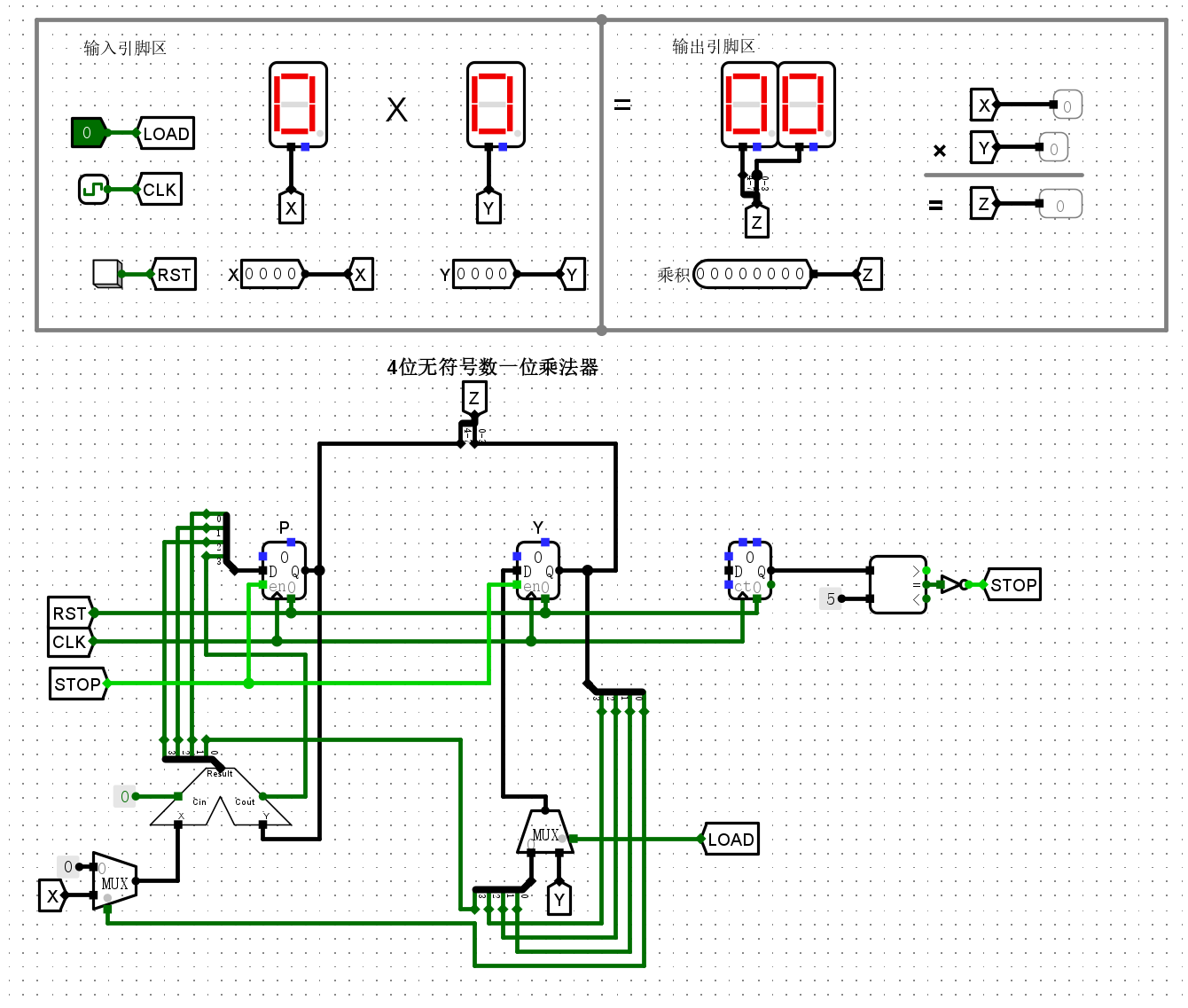
2）原理图：（图例为32位乘法器）



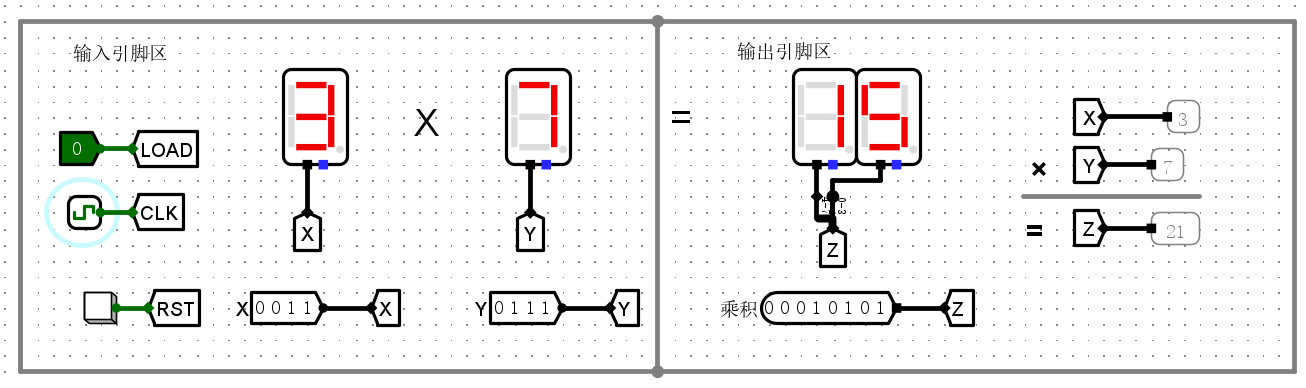
3）按需取用实验原件，如下摆放：



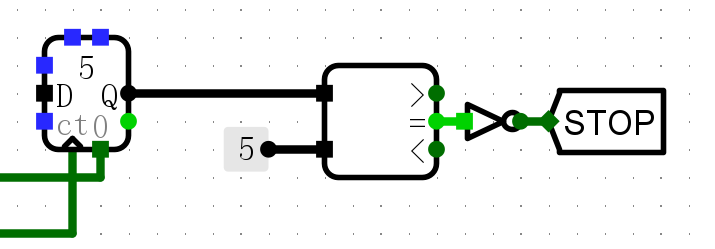
4）连接电路如下：



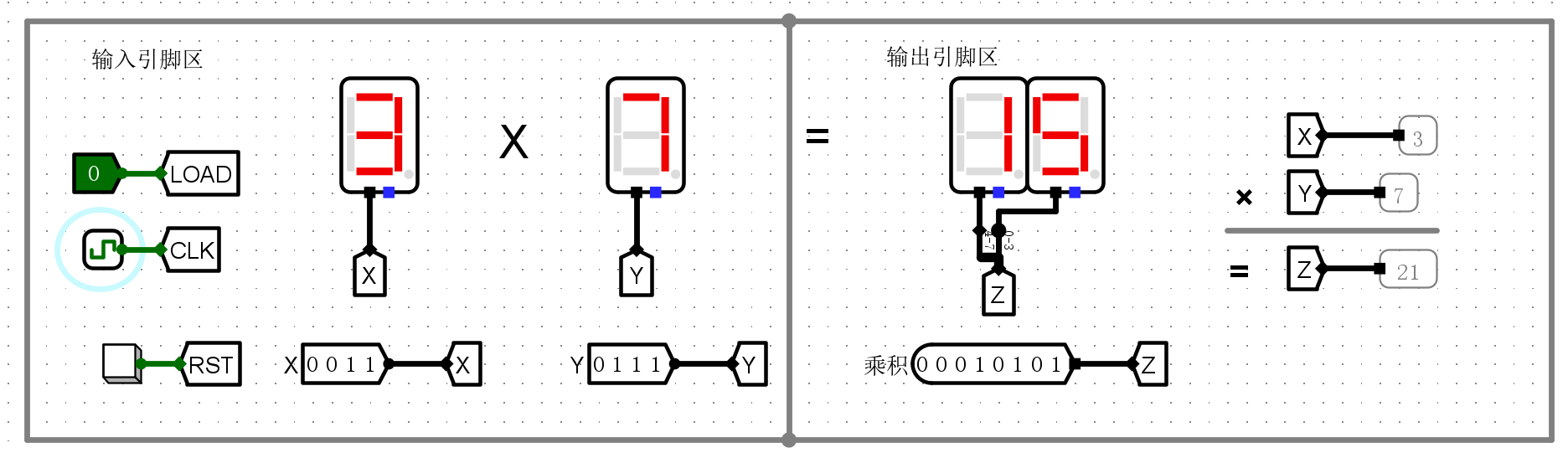
5）仿真测试电路功能

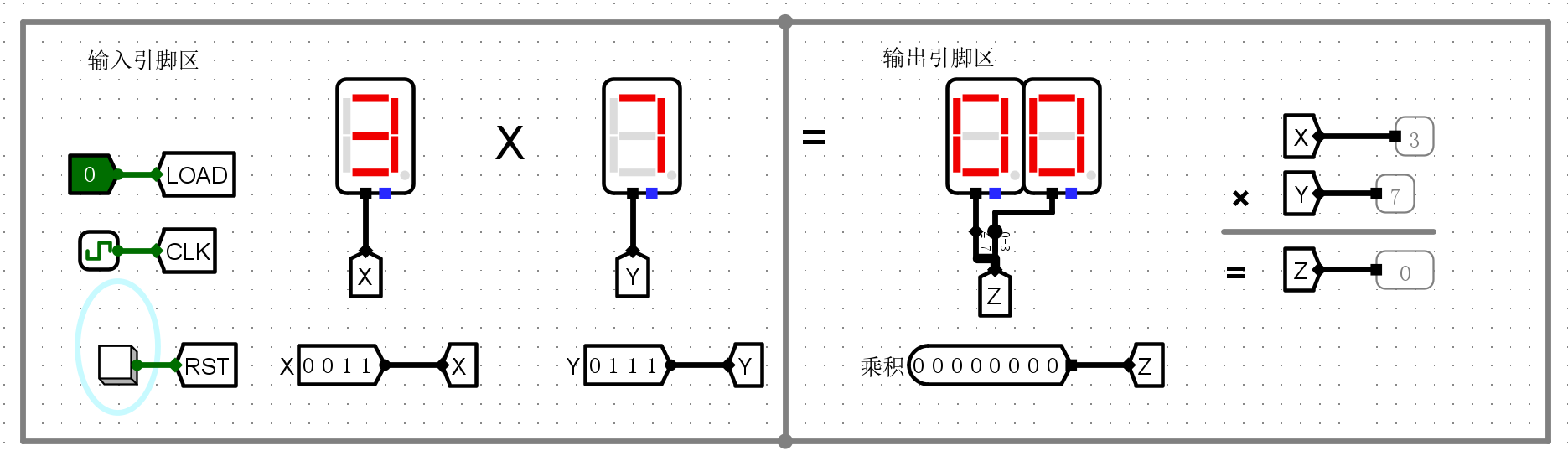
“3x7=21” 

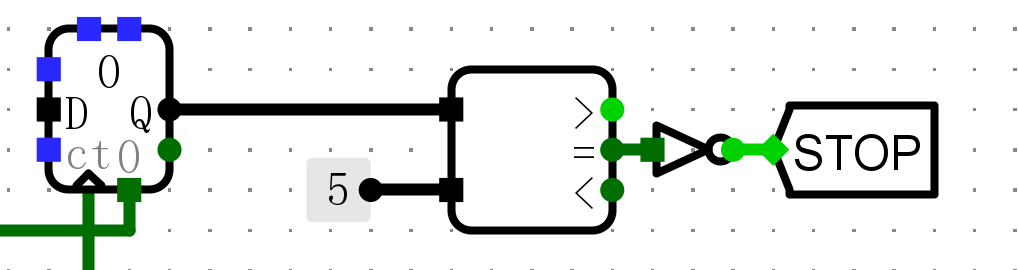
自动截止：



STOP信号为1，寄存器屏蔽时钟信号

复位：

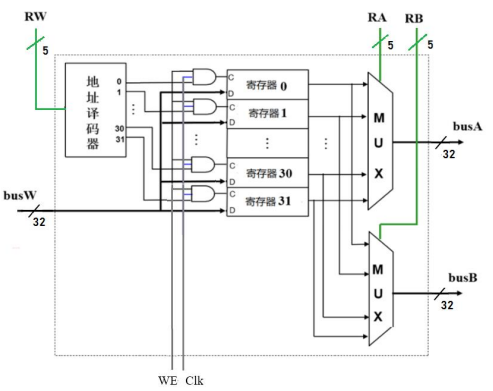
此时按下RST按钮，计算结果和STOP信号计数器清零



**5.寄存器堆实验**

根据图11.62中寄存器堆原理图，构建含有32个32位寄存器的寄存器堆Regfile的读写电路，包含两个读数据端口和一个写数据端口，并封装成子电路。寄存器堆的读操作属于组合逻辑操作，无须时钟控制，即当寄存器地址信号RA或RB到达后，经过一个“读取时间”的延迟，读出的数据输出到端口busA或busB上。寄存器堆的写操作则属于时序逻辑操作，需要时钟信号的控制，即在写使能信号（WE）有效的情况下，有效时钟触发边沿到来时开始将端口busW上的信息写入RW所指定的寄存器中。

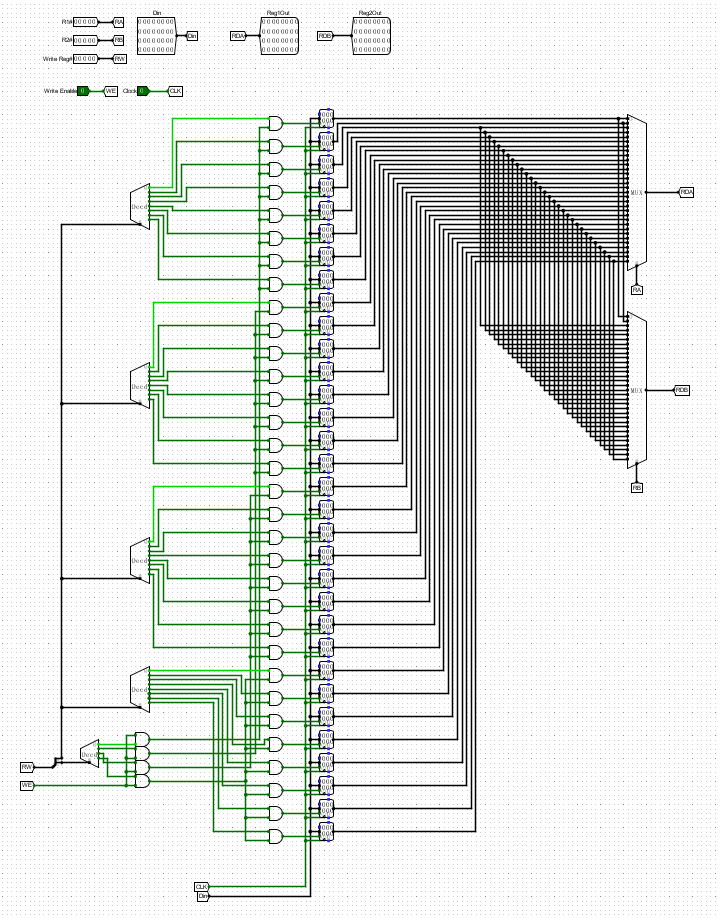
实验原理图：



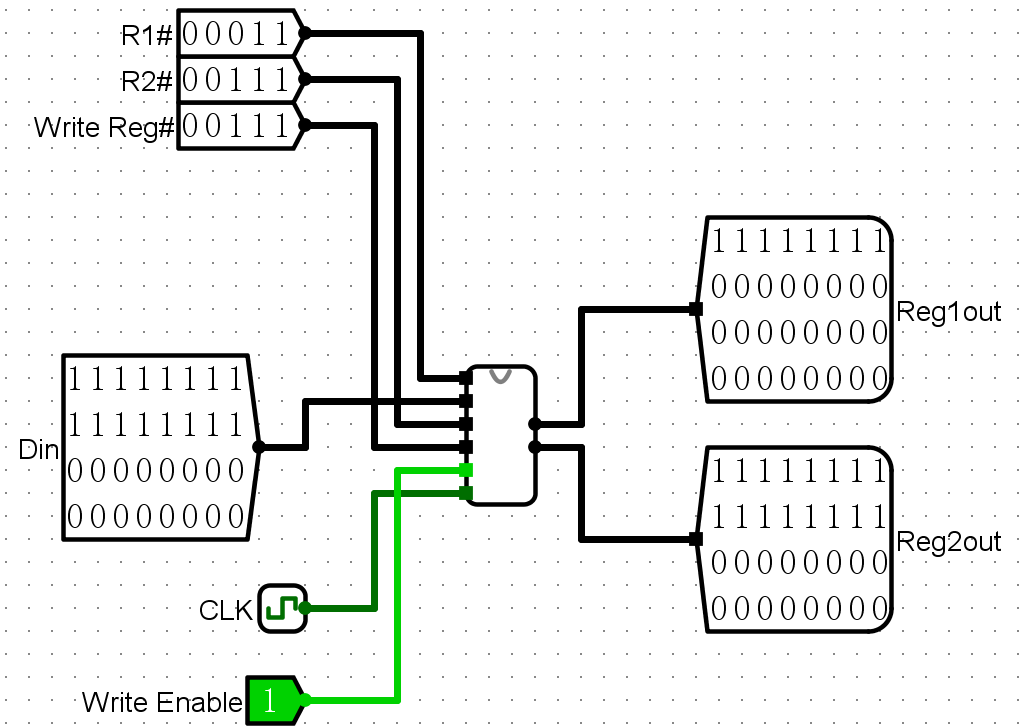
实验步骤：

1. 创建寄存器堆子电路。封装为子电路，命名为“regfiles”

电路如图：

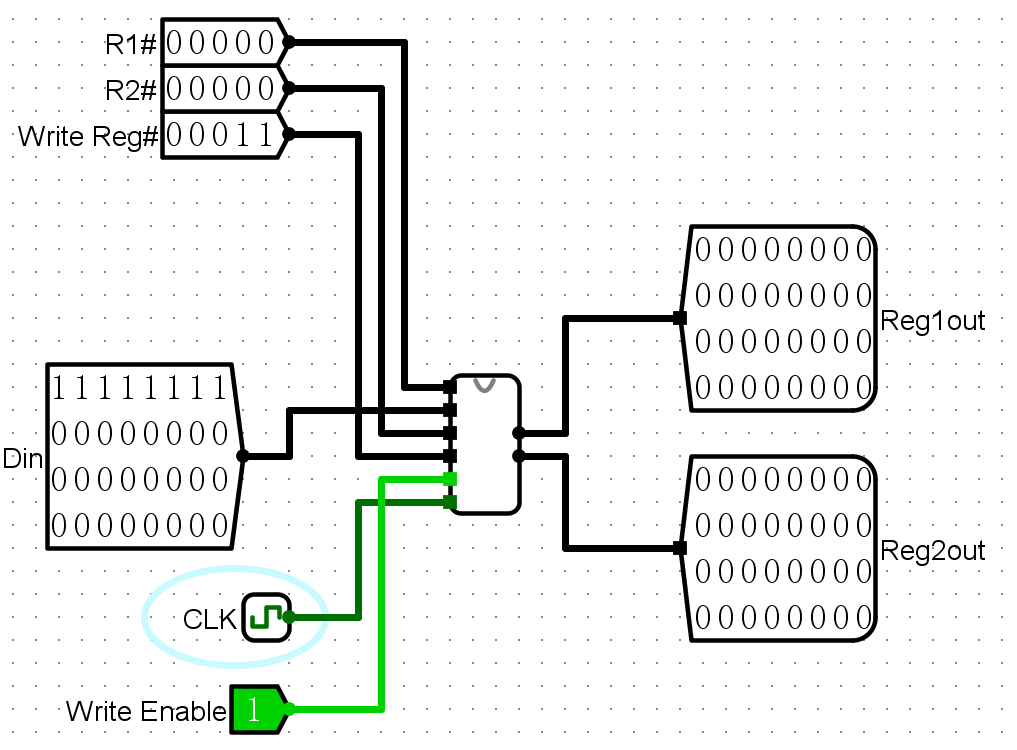


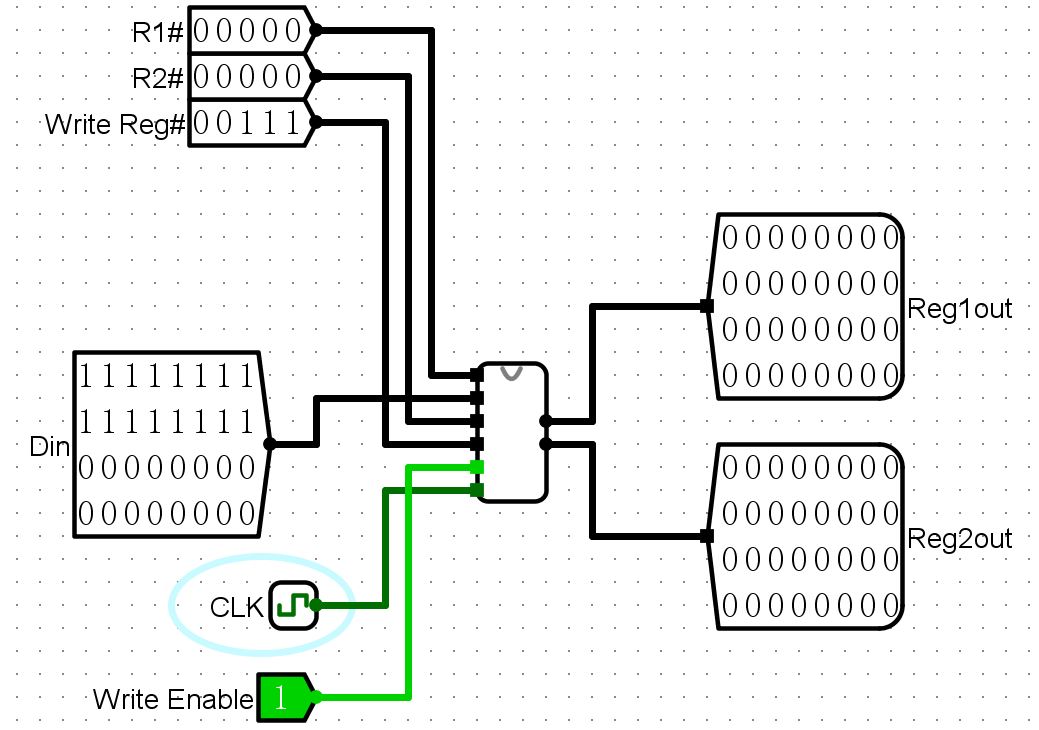
1. 在工程中添加一个名为“寄存器堆读写”的子电路，并完成电路

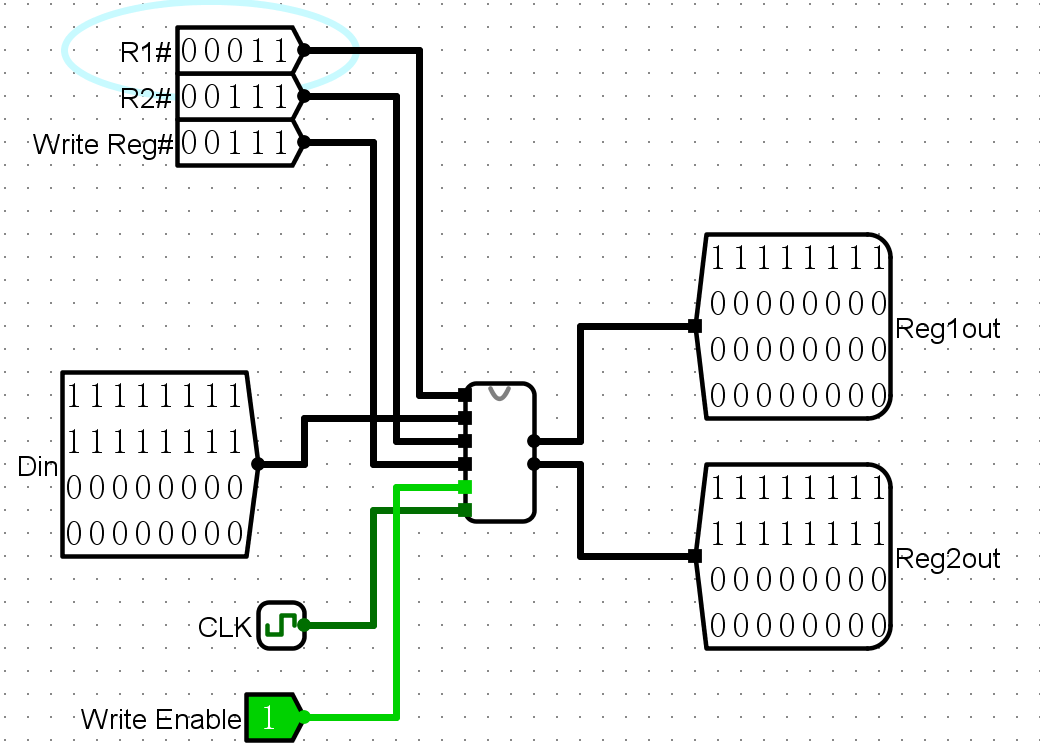


3）仿真测试：

读数取数检测







与预期结果相符

**四、实验中的错误**

实验过程中没有遇到什么难以解决的错误

**五、思考题**

1、在数字时钟实验中，如果需要增加闹钟功能，电路中需要如何修改？

·基本思路：额外设计闹钟模块的子电路，核心功能是比较设定时间和当前时间是否相同。

·考虑到闹钟的实际功能，闹钟模块应该具有以下功能：

1）具有写入设定时间的功能，且写入后保持不变。

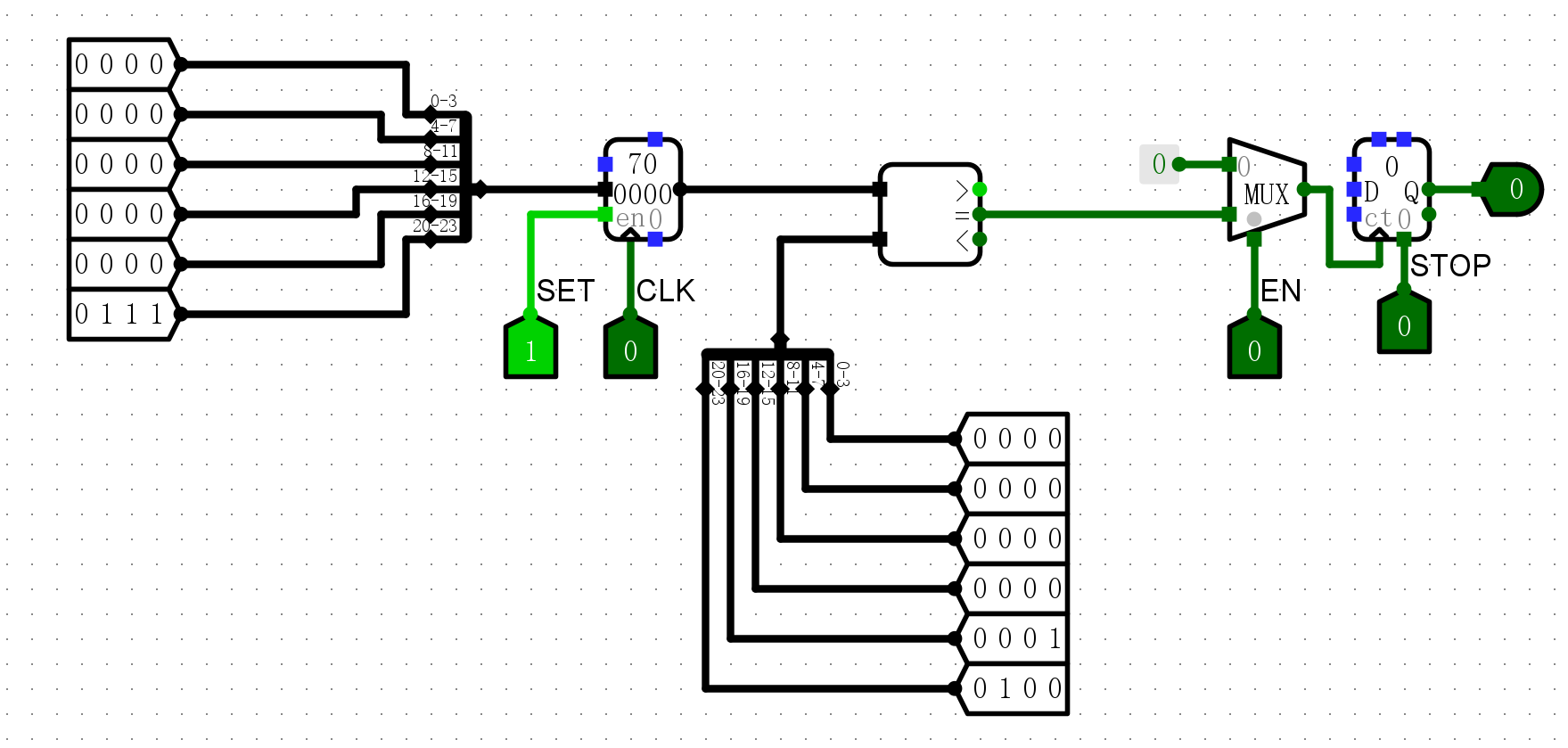
2）具有开关功能，当打开时，每当时间与设定值相同，闹铃响。当关闭时，即使时间与设定值相同，闹铃也不响。

3）闹铃响起后应当持续，无论时间是否继续流动。

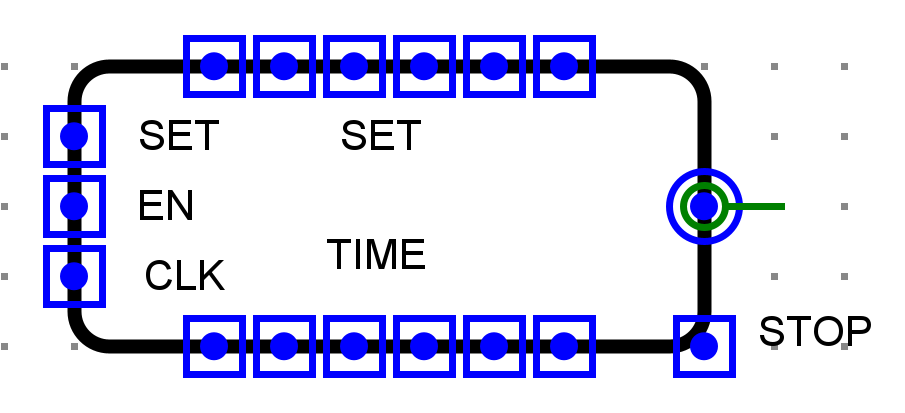
4）闹铃响起时，按下关闭按钮，闹铃停止。

5）蜂鸣频率应当较低，区别于整点报时

闹钟模块子电路如下：

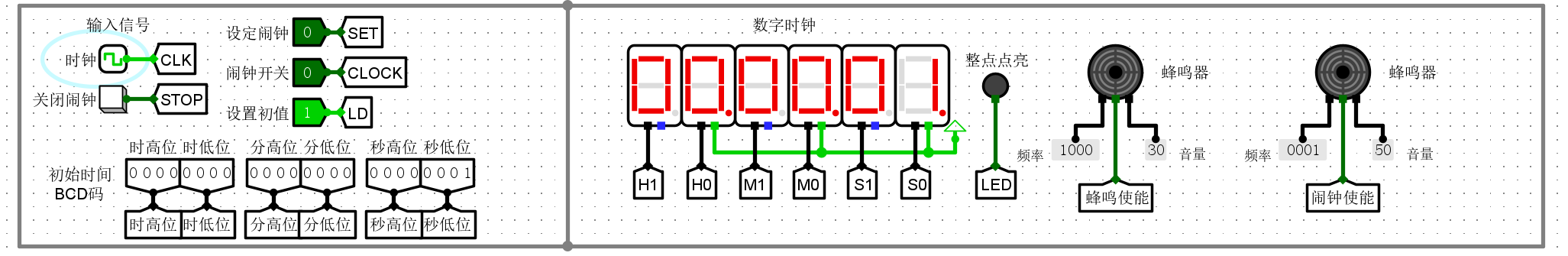


电路封装如下：

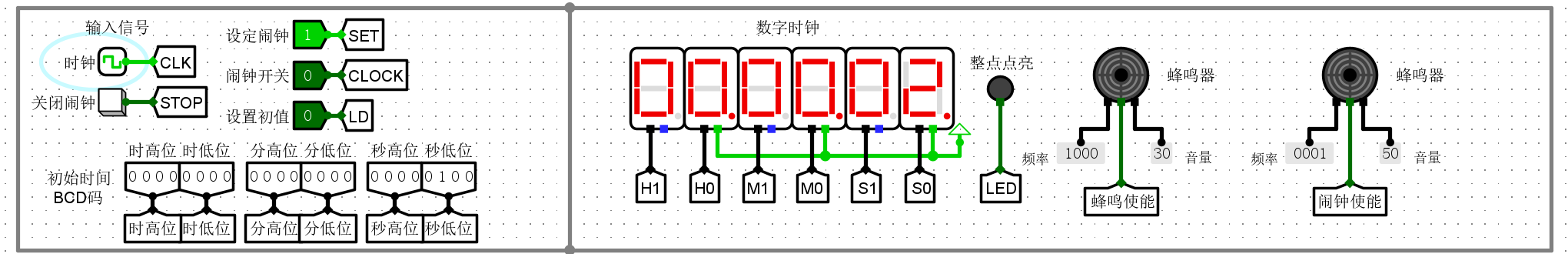


仿真测试：写入当前时间00.00.01，写入闹钟时间00.00.04

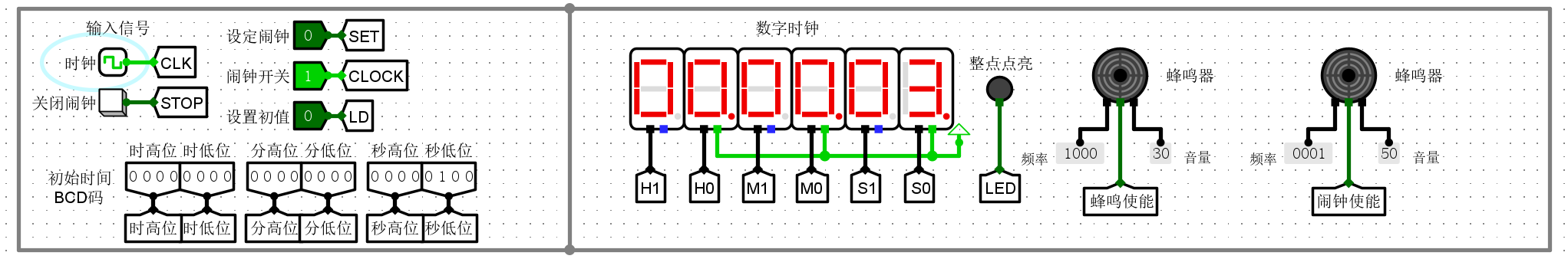
写入当前时间：

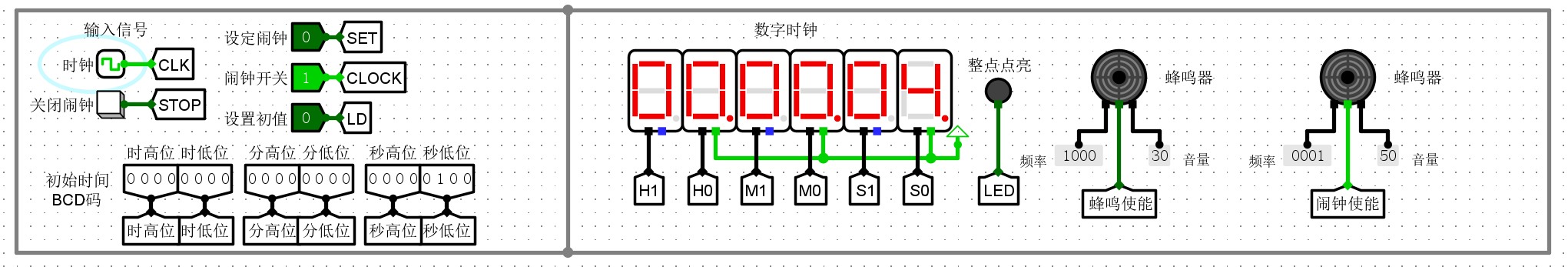


写入闹钟时间：

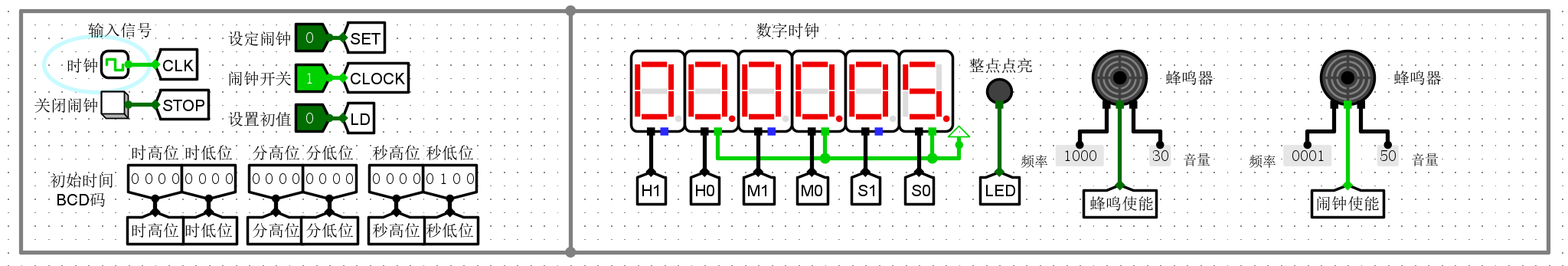


打开闹钟开关，等待铃响：

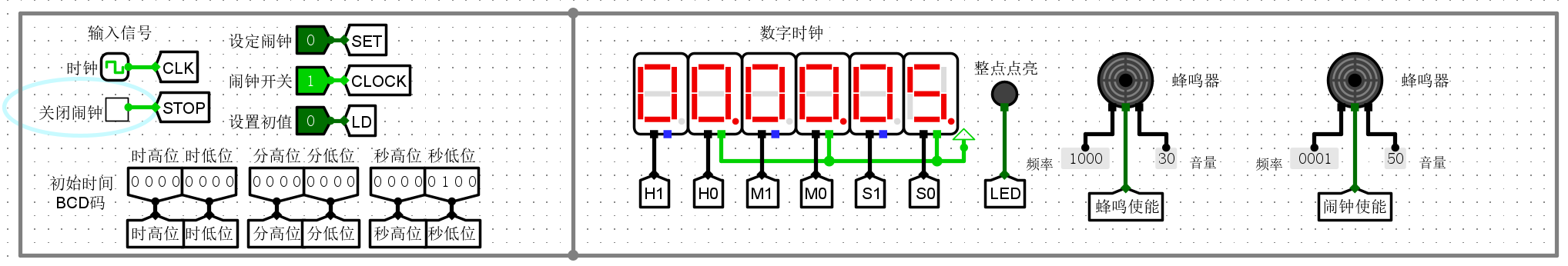




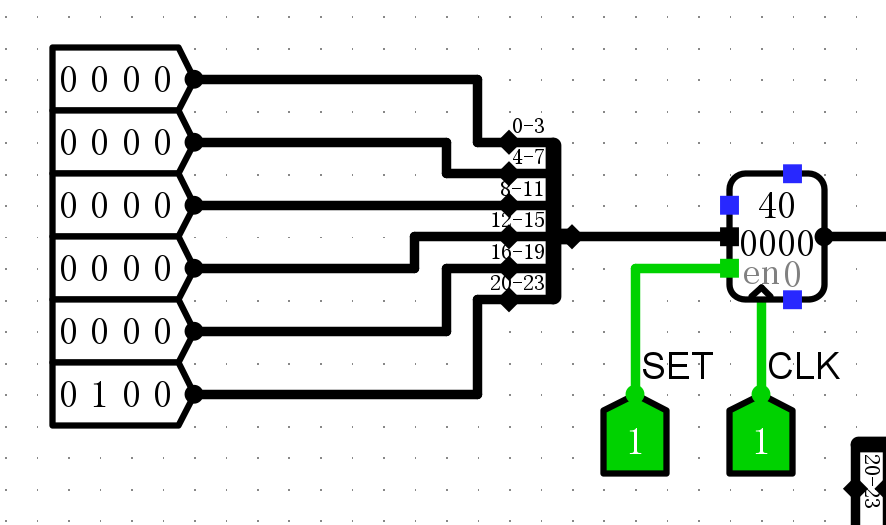
时间流动，闹铃不止：



按下停止按钮，闹铃停止：



检查闹铃模块，储存的时间正常，下一次闹铃将正常响起：



2、如何修改4位无符号数乘法器电路，使其实现4位Booth乘法器的功能？

·基本思路：大体构造无须改动。根据布斯算法，修改每次加到乘积寄存器内的部分积即可。

·布斯算法（补码一位乘法）：得到Pi后，判断乘数（Y）中的连续两位（YiYi-1）来确定Pi+1：

01：2-1（Pi+X）

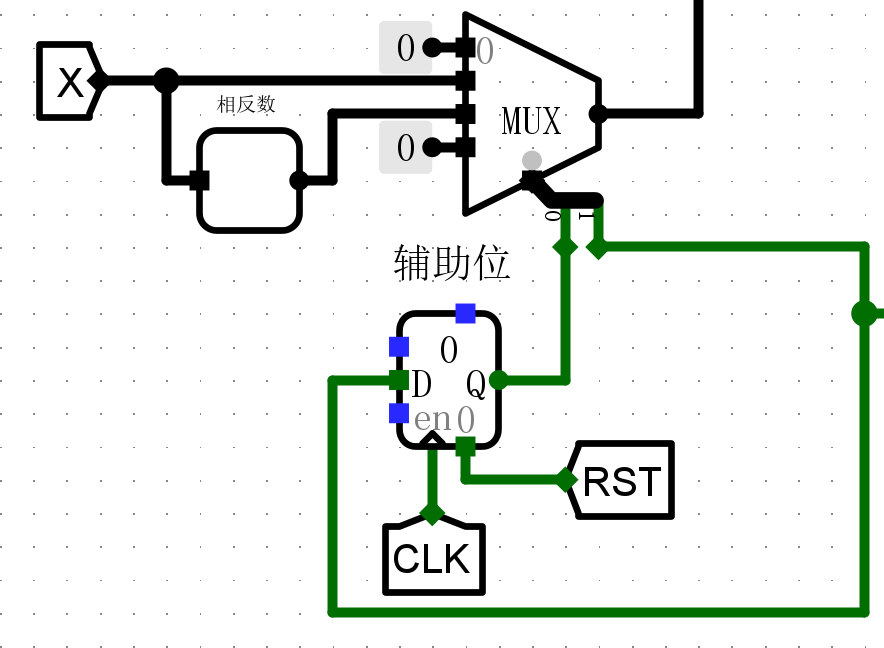
10：2-1（Pi-X）

11/00：2-1（Pi）

其中2-1可以通过补码右移（带符号数的算术右移）实现，-X通过+（-X）实现

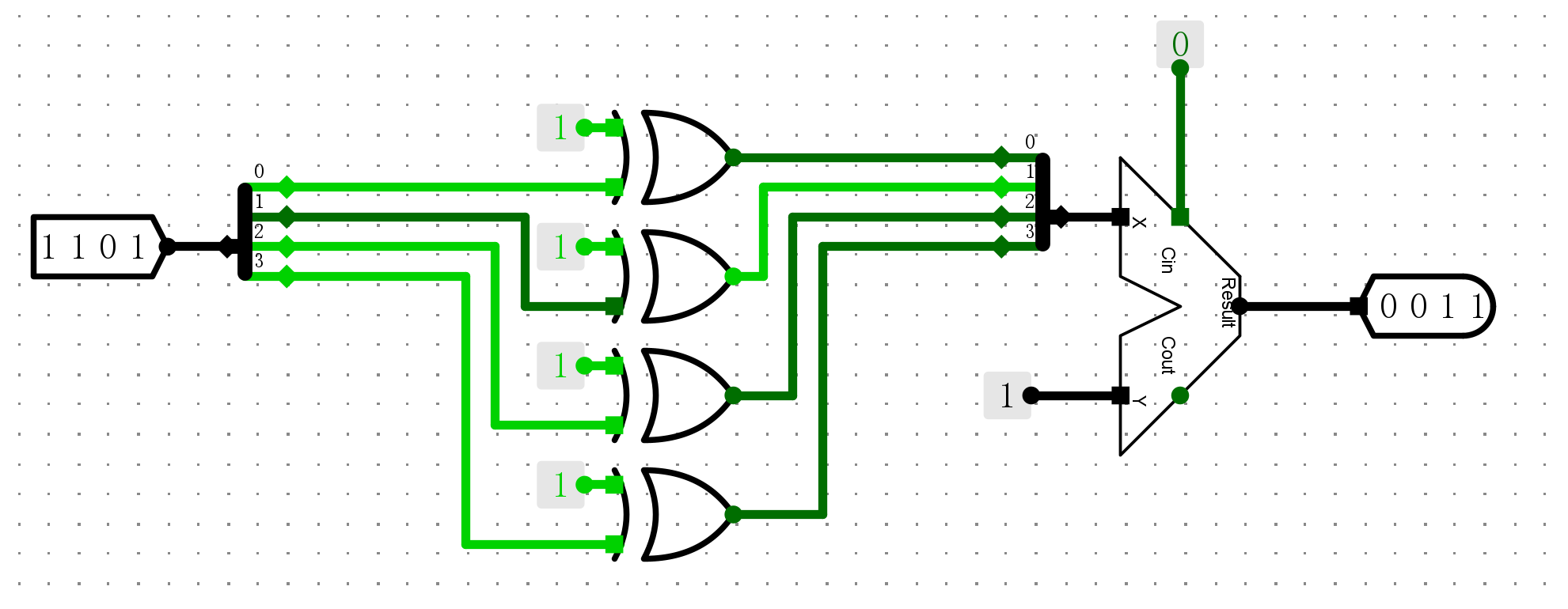
需要在乘数的最低位增加一个辅助位Y-1=0

·首先，修改加法器输入端的多路选择器如下：



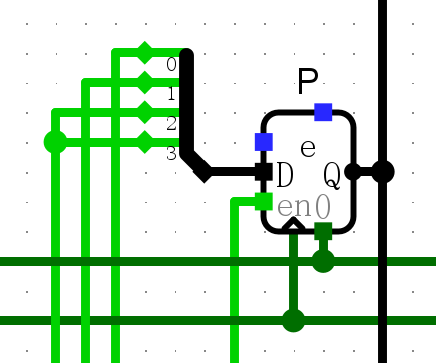
在这里，我们增加一个1位寄存器作为辅助位，同时引入乘数寄存器的最低位，二者即为YiYi-1，由此确定交给加法器的加数是X，-X，还是0。-X的实现上，我们再构建一个子电路：“相反数”，用来求X的相反数-X。

·子电路“相反数”实现电路如下：

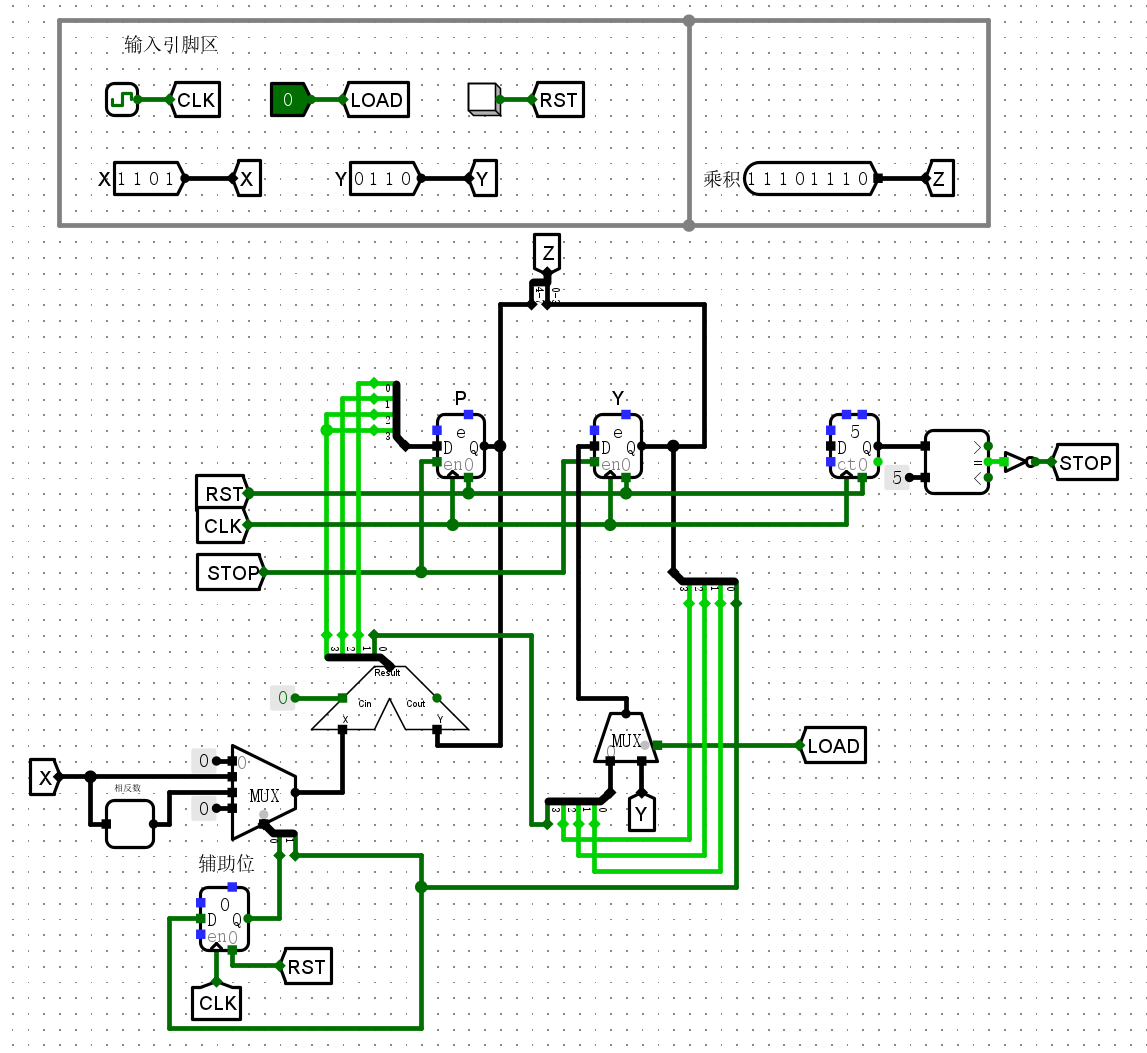


与1异或达到“各位取反”效果；与1相加得到“末位加一”。从而求出了-X的补码表示。

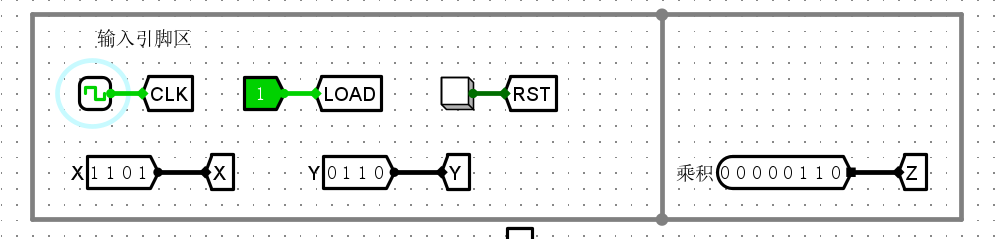
·然后，我们修改乘积寄存器的数据输入端，使之配合补码运算以及算数右移的性质：



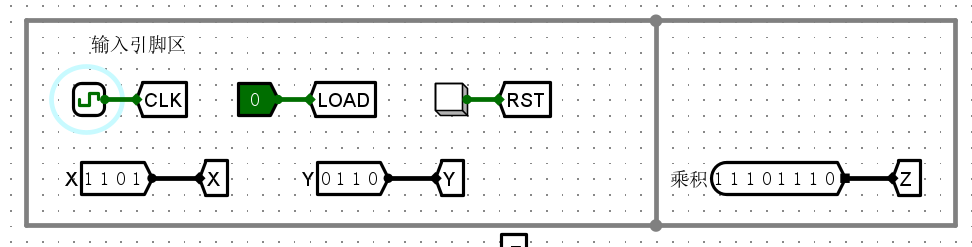
·最终电路实现如下：



仿真测试：计算X=1101，Y=0110，XxY=11101110

载入乘数：

计算结果（四次时钟周期后结果稳定不变）：



3、如何实现4位无符号数除法器？

（口述作答）

·主要思路：参照手算除法的原理，完成和乘法器结构类似的除法器。（采取恢复余数法）

第一步：搭建基本结构：一个4位寄存器用来存放除数Y；“串联”2个4位寄存器，用来存放余数和商。用4位减法器和比较器配合，作为核心计算部件。

第二步：搭建逻辑控制结构：每次时钟周期，将余数/商寄存器中的数左移，最高位移出，最低位变为上商。

第三步：完善预处理：

除数和被除数均为0，报“除法错误”异常，不计算

除数为0，被除数不为0，报“除数为0”异常，不计算

被除数为0，除数不为0；或被除数小于除数，直接商0，余被除数，不计算

·计算流程：

1）预处理。报错的两种情况单独处理，另一种情况直接将被除数填入余数寄存器，商寄存器全0，表示余被除数，商0。

2）完成除数和被除数的载入：除数写入4位除数寄存器，保持不变。被除数写入商寄存器，余数寄存器全部补0。

3）试商：用比较器比较余数寄存器中的余数和被除数。若前者更大，用二者的差作为中间余数填入余数寄存器，上商1，然后余数/商寄存器左移一位；反之，上商0，然后余数/商寄存器左移一位。

4）停止：用计数器计数试商的次数，达到四次以后，锁定所有寄存器，完成计算。、

·左移的实现：余数寄存器的最高位输出连接商寄存器的最低位；商寄存器最高位的输出连接余数寄存器的最低位。

·试商的实现：比较器配合减法器。比较器的结果决定填入余数寄存器的是原余数还是减法器得到的差。

·预处理的实现：一旦检测到导致报错的输入数据，直接锁定所有寄存器，保持报错信号。