華中科技大學

2022

数字电路与逻辑设计
实验报告

专　　业：计算机科学与技术

班　　级：CS2104

学　　号：U202115424

姓　　名：张森磊

电　　话：15670561689

邮　　件：1833842212@qq.com

完成日期：2022-12-21

实验报告及电路设计评分细则

评 分 项 目	满分	得分	备　注
一、实验报告总分	100		
1、文档格式（段落、行间距、缩进、图表、编号等规范化）	15		
2、设计方案与实验过程	60		实验过程将从电路的复杂度、是否考虑竞争和险象、电路的美观等方面进行评分。
3、遇到的问题及处理	10		
4、设计方案存在的不足	5		
5、实验心得（含思政）	5		
6、意见和建议	5		
二、电路得分(头歌)	100		
三、实验课程总分	**100**		实验报告总分*0.6 + 电路得分（头歌）*0.4
教师签名		日　期	

目　录

1　实验概述

1.1 实验名称

运动码表系统设计。

1.2 实验目的

实验将提供一个完整的数字逻辑实验包，从真值表方式构建 7 段数码管驱动电路，到逻辑表达式方式构建四位比较器，多路选择器，利用同步时序逻辑构建 BCD 计数器，从简单的组合逻辑电路到复杂时序逻辑电路，最终集成实现为运动码表系统。

实验由简到难，层次递进，从器件到部件，从部件到系统，通过本实验的设计、仿真、验证 3 个训练过程使同学们掌握小型数字电路系统的设计、仿真、调试方法以及电路模块封装的方法。

1.3 实验环境

软件：Logisim2.15.0.2 软件一套。

平台：https://www.educoder.net

1.4 实验内容

设计一个运动码表系统，具体内容及要求如下：

输入：4 个按钮，分别为 Start、Stop、Store 和 Reset。

输出：4 个 7 段数码管显示数字，分别显示秒和百分秒。

具体功能：

（1）当按下 Start 时，计时器清零，重新开始计时；

（2）当按下 Stop 时，计时器停止计时，显示计时数据；

（3）当按下 Store 时，若当前计时数据小于系统记录，则更新系统记录，并显示当前计时数据；否则不更新系统记录，但显示系统记录。

（4）当按下 Reset 时，复位，计时=0.00, 系统记录=99.99 秒。

1.5 实验要求

（1） 根据给定的实验包，将运动码表系统切分为一个个实验单元；

（2） 对每一个实验单元，按要求设计电路并使用 Logisim 软件进行虚拟仿真；

（3） 设计好的电路在 educoder 平台上提交并进行评测，直到通过全部关卡。

2　设计方案与实验过程

2.1 方案设计

本实验对运动码表数字系统进行模块化设计，按照以下流程进行系统的构建:

设计需求分析（外部数据，控制输入，输出，显示）-> 拆分为功能模块 -> 功能模块的数据通路（此时不考虑控制信号，只要数据链路通了就行）-> 构建控制单元（如何从外部控制输入转化到内部的控制信号）。

2.1.1　码表功能需求分析

- 输入：四个按钮　输出：四个 7 段数码管
- Start：计时器归零，重新开始计时。
- Stop：停止计时，显示计时数据。
- Store: 尝试更新系统记录，并显示系统记录。
- Reset: 复位，计时=00.00，系统记录=99.99。

2.1.2　码表功能部件设计

（1）运动码表模块划分如表 1

	功能部件	控制信号	输入	输出
1	时间计数器 TM	TM-En，TM-Rst	CLK	时间计数输出 16 位
2	16 位寄存器	SD-En	CLK,Din(16 位)	Q(16 位)
3	数码管显示		Din(16 位)	DisplayInfo(32 位)
4	比较器			
5	2 路选择器	Sel		

表 1

2.1.3 运动码表模块实现

（1）时间计数器

a.四位 BCD 计数器设计

一共存在 0~9 共 10 个状态，对应的状态编码为 0000~1001。由状态转换真值表如图 1 可以得到四位 BCD 计数器状态机。接着进行输出函数的设计，只有当状态为 1001 输出 1，即 Cout=S3&S1 可直接得到如图 2 电路。完成状态机以及输出函数的设计后，借助 4 个 D 触发器完成四位 BCD 计数器的构建。

S3	S2	S1	S0	N3	N2	N1	N0
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	0	0	0	0
1	0	1	0	1	0	1	1
1	0	1	1	0	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	0	1	0	0
1	1	1	0	1	1	1	1
1	1	1	1	1	0	0	0

图 1

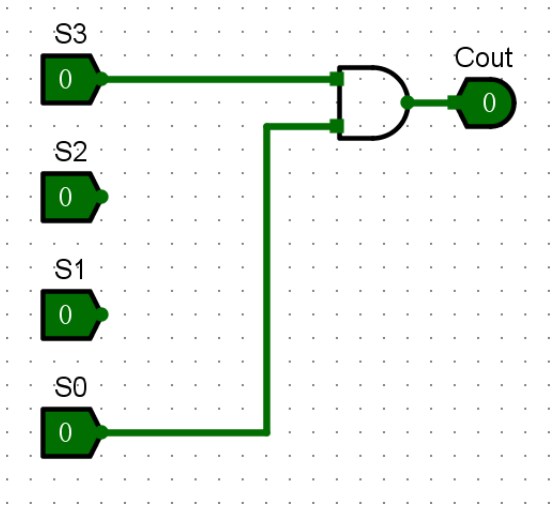

图 2

b. 码表计数器设计

码表计数器一共 16 位输出，包括 4 个 BCD 计数器，分别对应 10 秒，1 秒，1/10 秒，1/100 秒。低位计数到 9 时，相邻高位在时钟到来时加 1。

通过每个 BCD 计数器的进位输出 Carry 以及使能端 en 构建码表计数器。只有当低位都产生进位输出的情况下，高位加 1。因此，第 1 位的 en 由第 0 位的 Carry 过与门得到，第 2 位的 en 由第 0，1 位的 Carry 过与门得到，第 3 位的 en 由第 0，1，2 位的 Carry 过与门得到。

（2）16 位寄存器

如图 3，利用四个 D 触发器，构造 4 位并行加载寄存器。再用四个 4 位并行加载

寄存器构造 16 位并行加载寄存器。

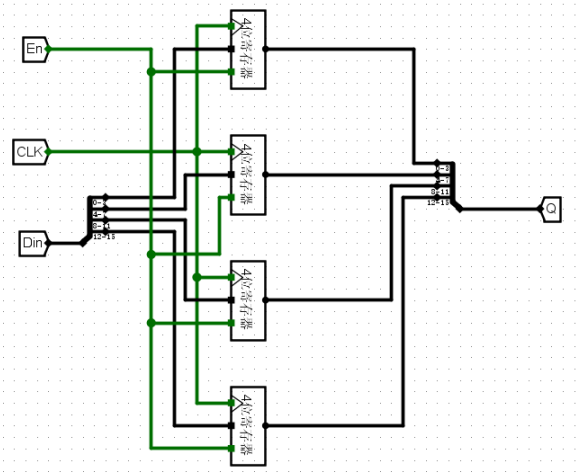

图 3

（3）数码管显示

a.数码管驱动

数码管驱动是将数字 0-9 转化为数码管的 0 - 9 输出；数码管元件有 a - g 七根管加一个小数点共 8 位输出，小数点可以单独处理，我们设计驱动电路的时候 4 输入，7 输出即可。利用图 4 真值表可以在 logisim 中直接得到。

X3	X2	X1	X0	Seg_1	Seg_2	Seg_3	Seg_4	Seg_5	Seg_6	Seg_7
0	0	0	0	0	1	1	1	1	1	1
0	0	0	1	0	0	0	1	0	0	1
0	0	1	0	1	0	1	1	1	1	0
0	0	1	1	1	0	1	1	0	1	1
0	1	0	0	1	1	0	1	0	0	1
0	1	0	1	1	1	1	0	0	1	1
0	1	1	0	1	1	1	0	1	1	1
0	1	1	1	0	0	1	1	0	0	1
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	0	1	1
1	0	1	0	1	0	0	0	1	1	0
1	0	1	1	1	0	0	0	0	1	1
1	1	0	0	1	1	0	1	0	0	0
1	1	0	1	1	1	1	0	0	1	0
1	1	1	0	1	1	0	0	1	1	0
1	1	1	1	0	0	0	0	0	0	0

图 4

b.数码管显示驱动

通过分线器将输入的 Din 分为 16 个数，每 4 位数用一个数码管驱动分出 7 条控制 7 段数码管的线。再用一个分线器和电源，把一共 32 个数值合成一个 32 位数 DispInfo。

（4）无符号比较器

a. 4 位无符号比较器

按照先高位，后低位的比较原则，可以写出输出 Great 的表达式为 X3~Y3+X3⊕Y3 X2 ~Y2+ X3⊕Y3 X2⊕Y2 X1~Y1+ X3⊕Y3 X2⊕Y2 X1⊕Y1X0~Y0；Equal 的表达式为 X3⊕Y3 X2⊕Y2 X1⊕Y1X0⊕Y0；Less 的表达式为~X3Y3+X3⊕Y3 ~X2 Y2+ X3⊕Y3 X2⊕Y2 ~X1Y1+ X3⊕Y3 X2⊕Y2 X1⊕Y1~X0Y0 将表达式分别填入 logisim 中即可得到如图 5 所示电路图。

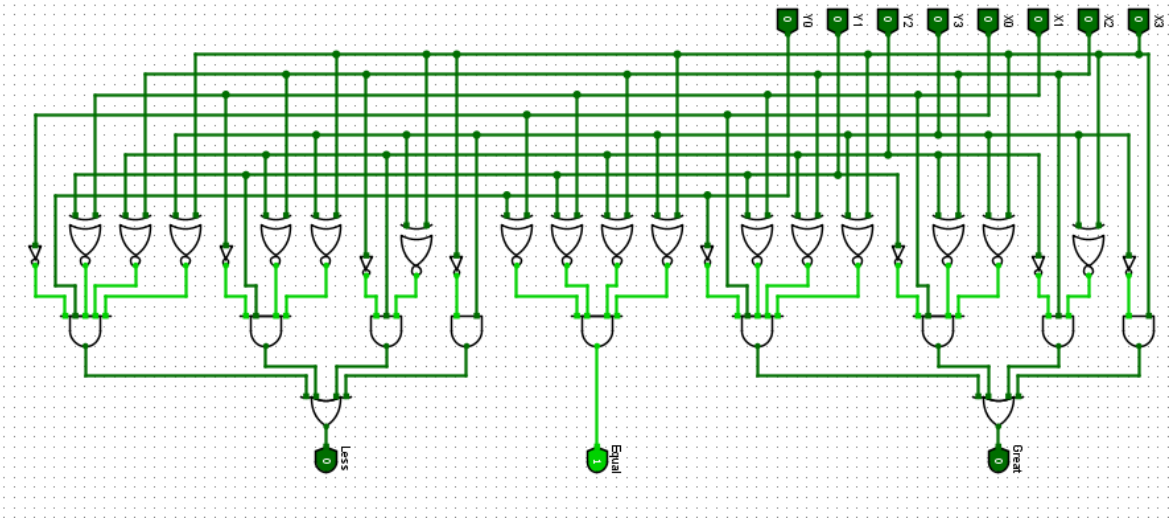

图 5

b.16 位无符号比较器

16 位无符号比较器需要 4 个 4 位无符号比较器，构建思路与 4 位无符号比较器相同，不过一次比较四位。另外利用 Less=~Great~Equal 对电路进行简化。

（5）2 路选择器

a.2 路选择器（1 位）

利用逻辑表达式 Out=SelX1+~SelX2 可以得到二路选择器（1 位）的电路。

b.2 路选择器(16 位)

利用分线器将 16 位的输入分成 16 个一位，分别与 1 位的 2 路选择器连接，最后再通过分线器将输出合成 16 位。

2.1.3 码表数据通路构建

通过构建输入来源表（表 2），按数据流构建数据通路，其中有多个输入来源的需要增设多路选择器。

	功能部件	数据输入	来源
1	时间计数器 TM		
2	16 位寄存器	CLK,Din(16 位)	99.99 或者当前记录
3	数码管显示 DP	Din(16 位)	TM.Q 或 SD.Q
4	比较器		当前计时&SD.Q

表 2

2.2.4 控制单元构建

（1） 状态转换

如图 6，根据码表功能的需要，设计了 000~101 共 6 个状态。

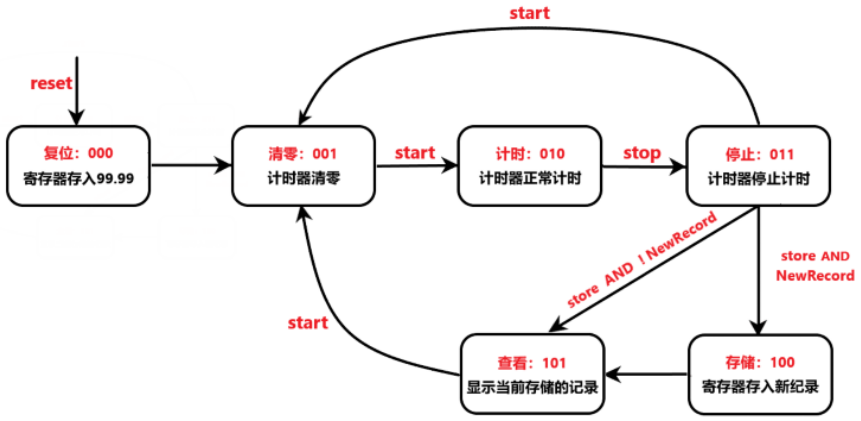

图 6

之后在 excel 中填写状态转移表，可以得到自动生成的逻辑表达式如下：

N2：~start ~reset S2 ~S1 + ~reset S2 ~S1 ~S0 + ~start ~stop store ~reset ~S2 S1 S0 + store ~reset S2 ~S1 + stop ~reset S2 ~S1

N1：~start ~store ~reset ~S2 S1 + ~reset ~S2 S1 ~S0 + stop ~reset ~S2 S1 + start ~stop ~store ~reset ~S2 ~S1 S0 + start store ~reset ~S2 S1

N0：~start ~reset ~S1 + ~reset ~S1 ~S0 + ~store ~reset ~S2 S1 S0 + ~reset ~NewRecord ~S2 S1 S0 + ~reset S2 ~S1 + ~start stop ~store ~reset ~S2 + stop ~reset ~S2 S0 + start store ~reset ~S2 S0

将上述逻辑表达式输入到 logisim 中得到状态转换电路。

（2） 输出函数

输出函数真值表如图 7 所示，利用真值表可以直接生成输出函数电路。

当前状态(现态)					输入信号										
S3	S2	S1	S0	现态10进制	start	stop	store	reset	NewRecord			SDsel	SDen	DPsel	TMen	TMreset
0	0	0	0	0								0	1	1	0	1
0	0	0	1	1								0	0	1	0	1
0	0	1	0	2								0	0	1	1	0
0	0	1	1	3								0	0	1	0	0
0	1	0	0	4								1	1	1	0	0
0	1	0	1	5								0	0	0	0	0

图 7

2.2.5 运动码表设计

运动码表如图 8，初始状态时要保证 16 位寄存器 SD 中储存的数据无穷大，即设为 9999。第一次 Store 的时候比较 9999 与计时器 TM 中值的大小，此时用 SD-SEL 来决定，把小的那个值传入寄存器 SD 中。以后每次 Store 都要比较寄存器 SD 中的值与当前计时器 TM 中值的大小，通过 16 位无符号比较器来完成，取小的那一个存入寄存器 SD 中，并输入码表显示 DP，从而显示在码表上同时修改 NewRecord 的值，此时用 DP-SEL 来决定。

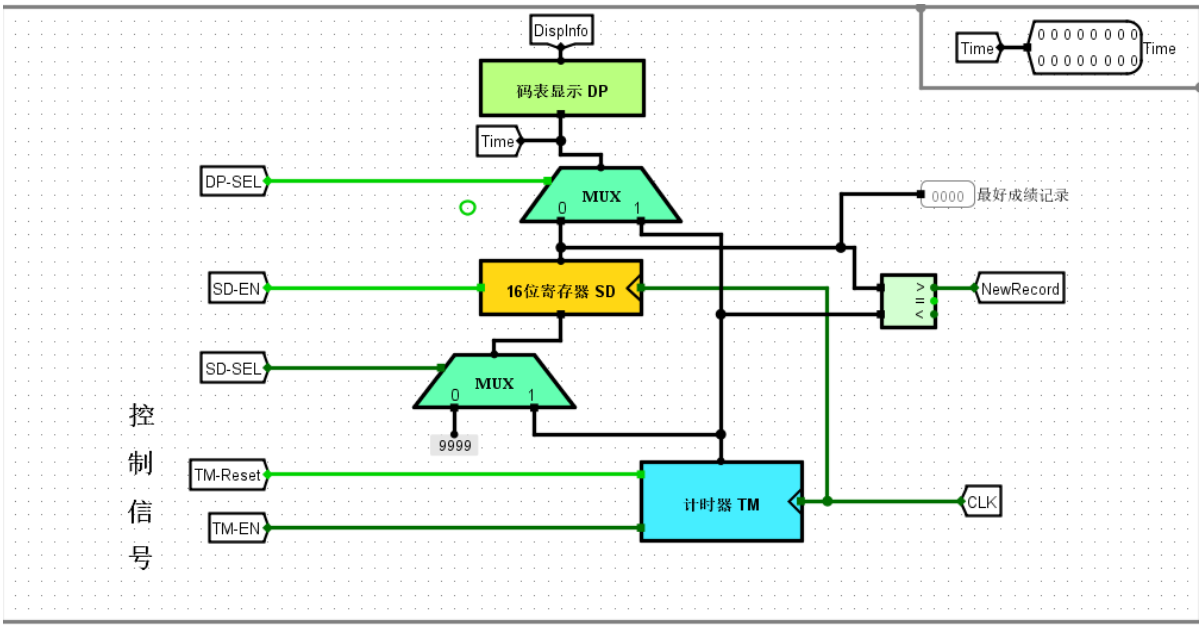

图 8

2.2 实验过程

2.2.1 7 段数码管驱动电路

（1）电路图

a. 内部结构电路如图 9 所示。

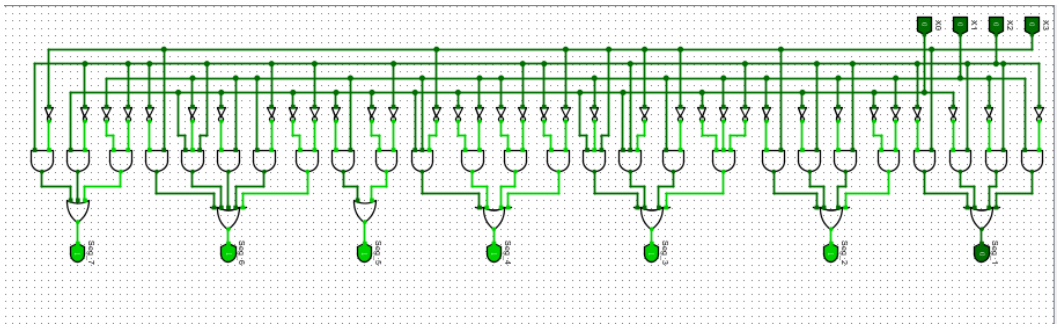

图 9

b. 封装电路图如图 10 所示。

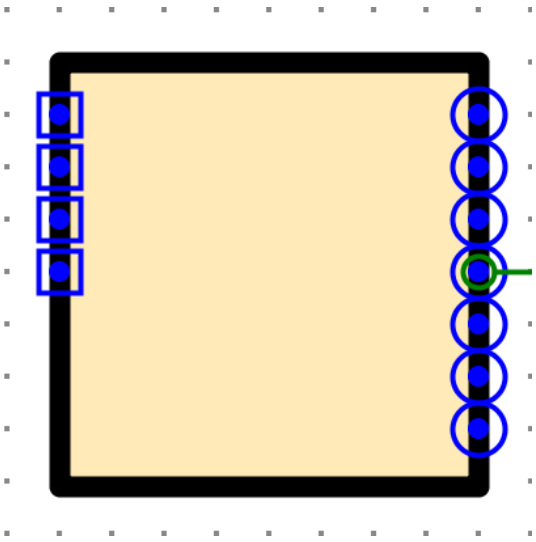

图 10

电路引脚如表 3 所示。

信号	I/O	位宽	说明
X3~X0	输入	1	4 位 BCD 码输入
Seg1~Seg7	输输出	1	7 位数码管驱动信号

表 3

（2）测试图

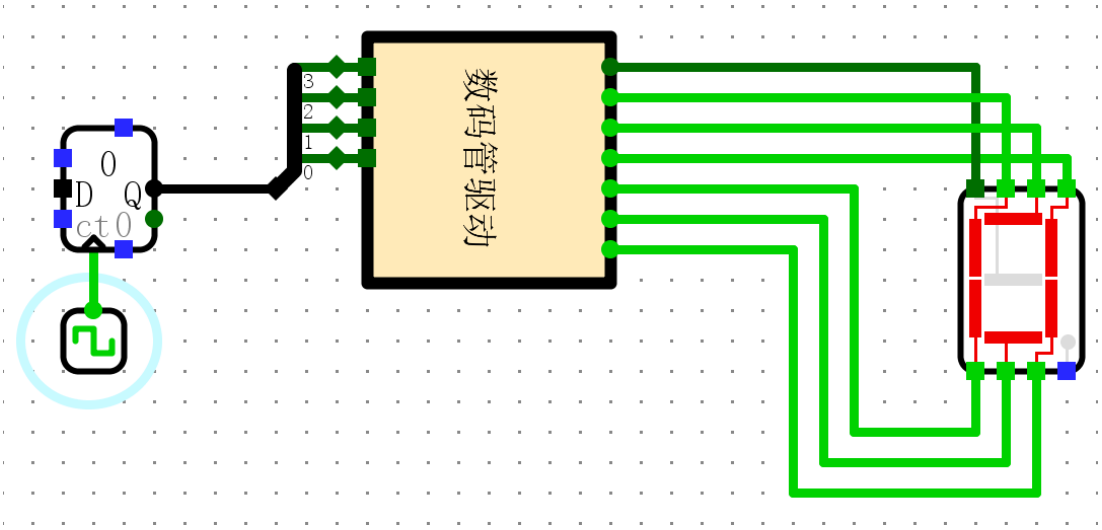

图 11 数码管显示“0”

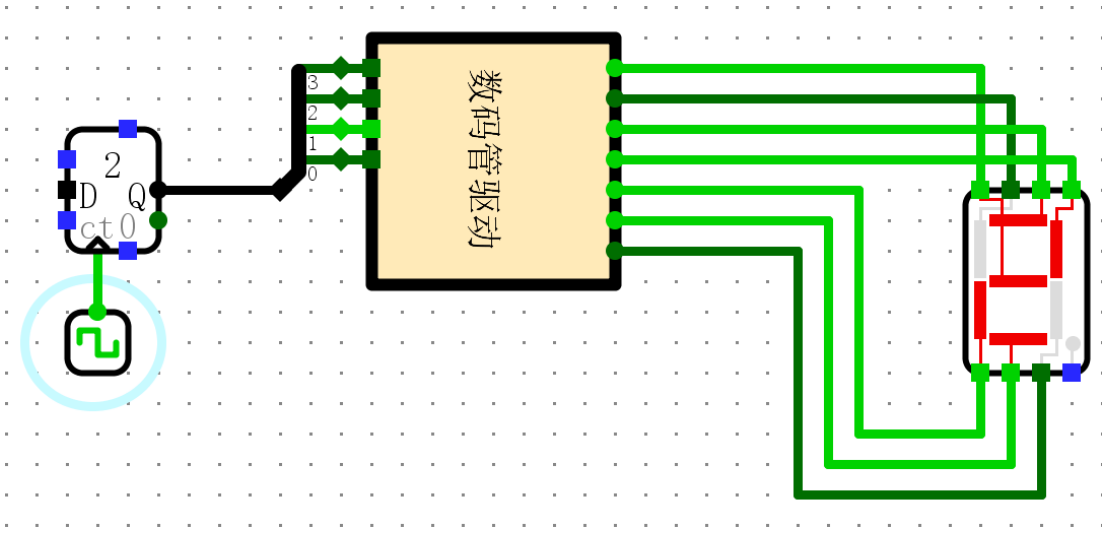

图 12 数码管显示“2”

（3）测试分析

该 7 数码管驱动电路可以正确将 4 位 BCD 码显示。

2.2.2 二路选择器（16 位）

（1）电路图

a. 内部结构电路如图 14 所示。

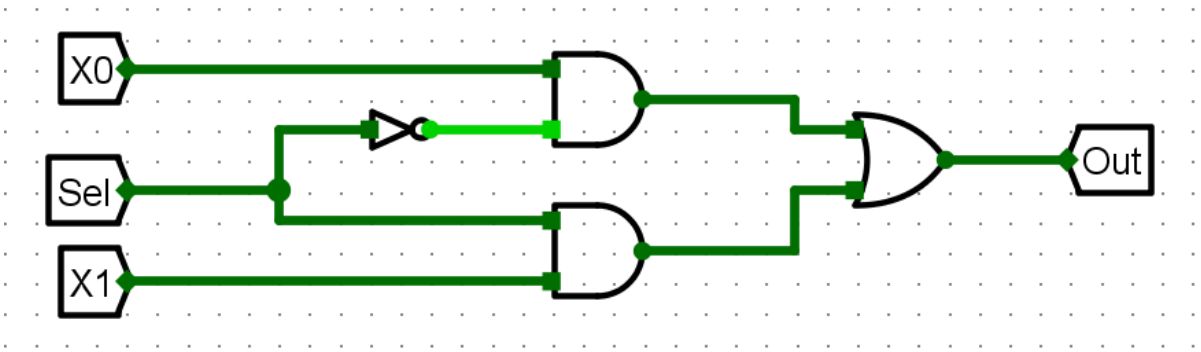

图 13 2 路选择器（1 位）电路

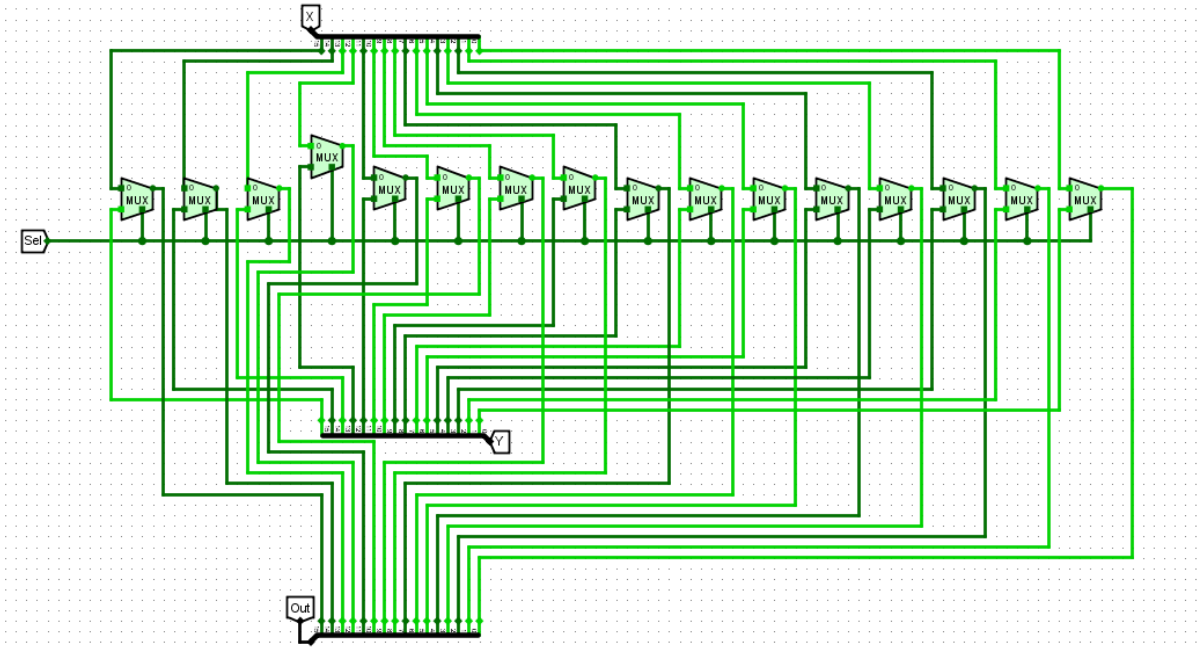

图 14 2 路选择器（16 位）电路

b. 封装电路图如图 15 所示。

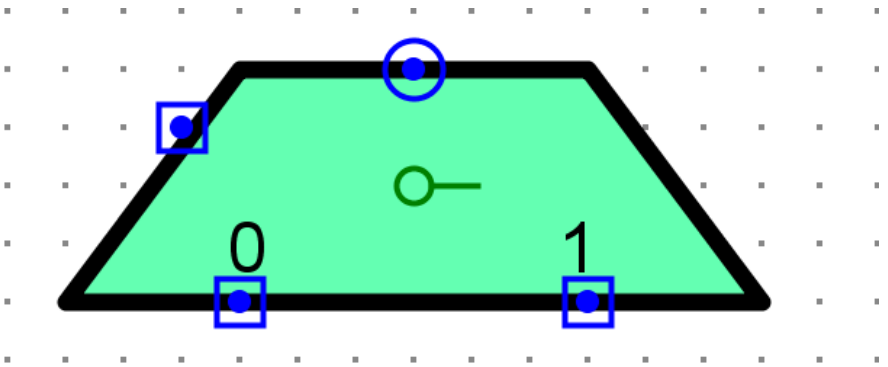

图 15

电路引脚如表 4 所示。

信号	I/O	位宽	说明
X0	输入	16	2 路输入之一路
X1	输入	16	2 路输入之一路
Sel	输入	16	选择控制端
Out	输出	16	Out=(Sel==0)?X0:X1;

表 4

（2） 测试图

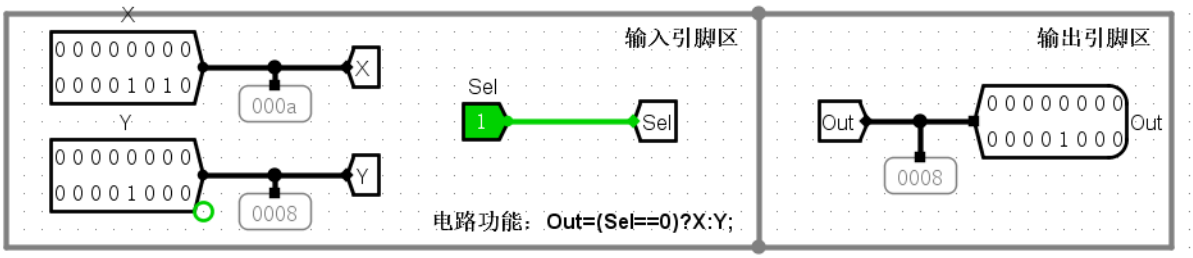

图 16

（3） 测试分析

此时电路输入 X 为 000a、Y 为 0008，Sel 为 1，输出 Out=Y=0008。

2.2.3 16 位无符号比较器

（1）电路图

a. 内部结构电路如图 17 所示

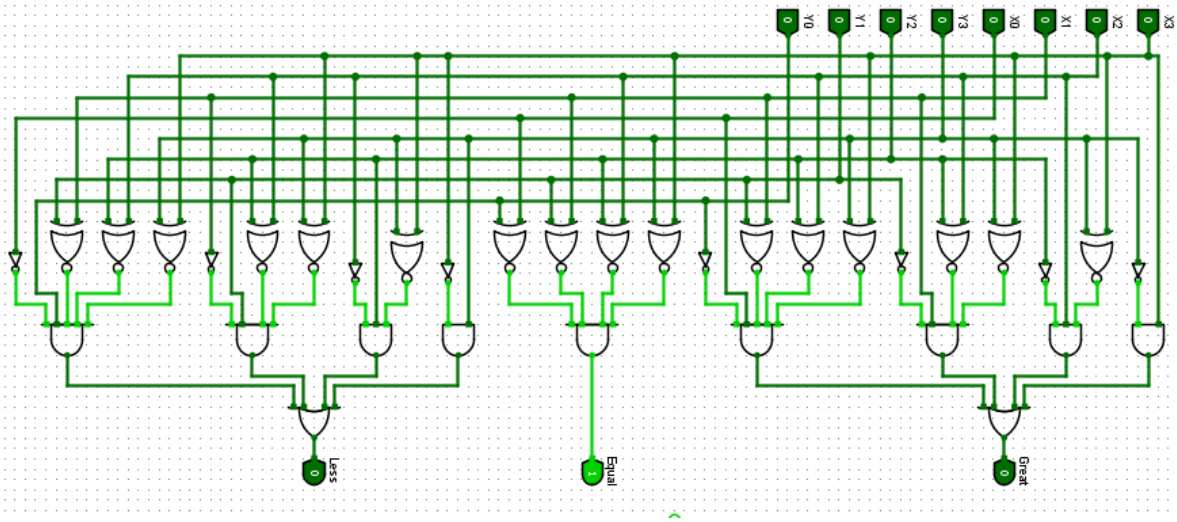

图 17

b. 封装电路图如图 18 所示。

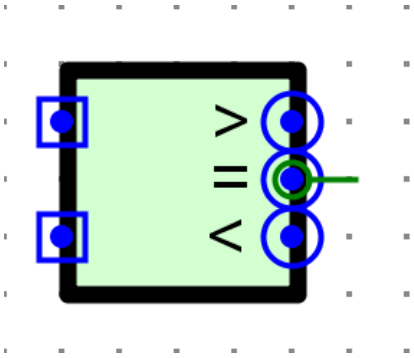

图 18

电路引脚如表 5 所示。

信号	I/O	位宽	说明
X	输入	16	输入 X
Y	输入	16	输出 Y
Great	输出	1	X 大于 Y
Equal	输出	1	X 等于 Y
Less	输出	1	X 小于 Y

表 5

（2）测试图

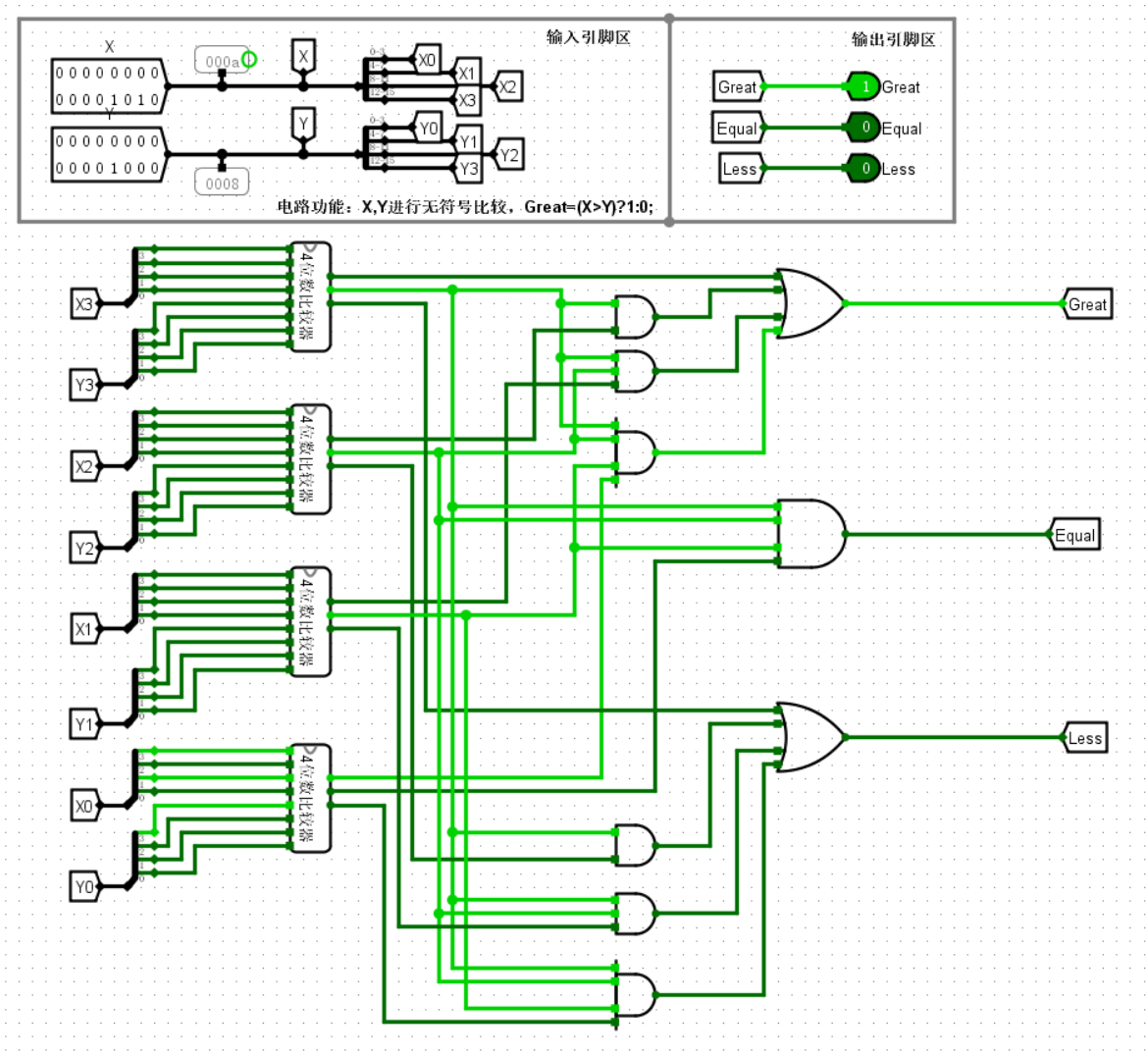

图 19

（3）测试分析

电路输入 X 为 000a、Y 为 0008，输出 Great=1，Equal=Less=0。结果正确。X=Y，X<Y 情况同理。

2.2.4 并行加载寄存器

（1）电路图

a. 内部结构电路如图 21 所示。

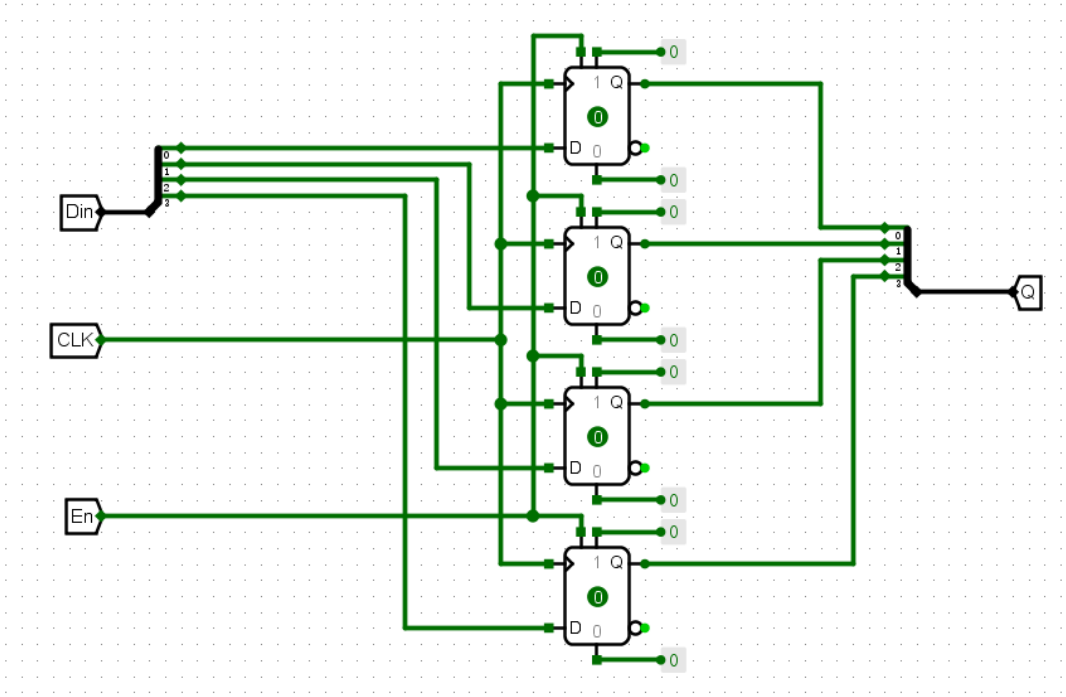

图 20 4 位并行加载寄存器

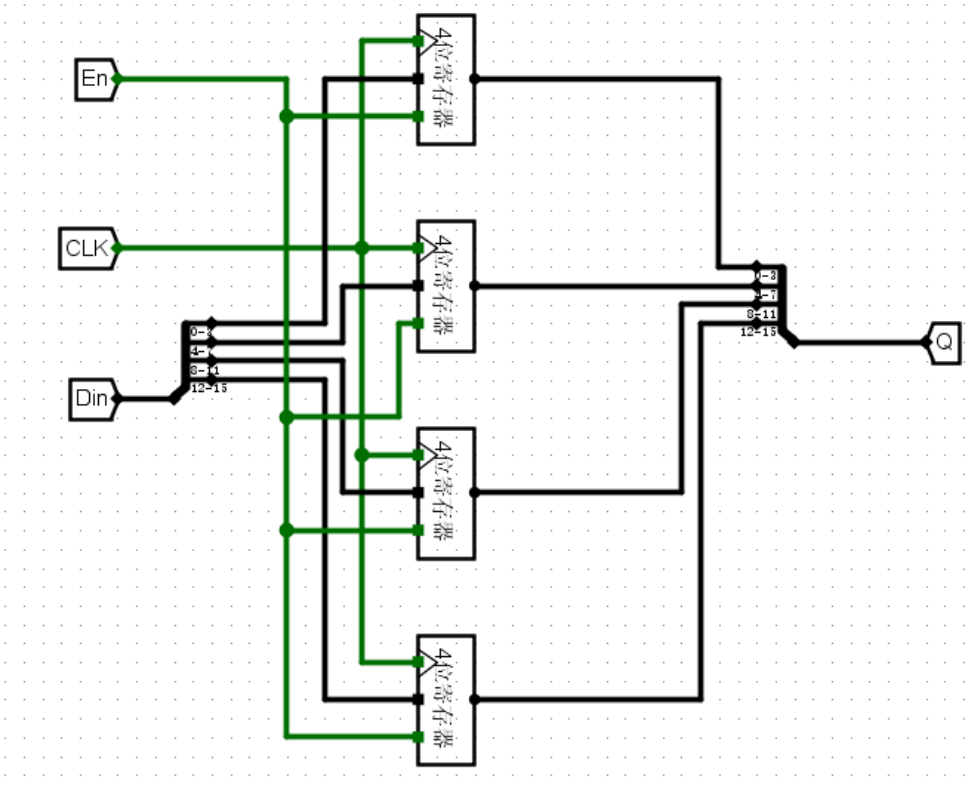

图 21 16 位并行加载寄存器

b. 封装电路图如图 22 所示。

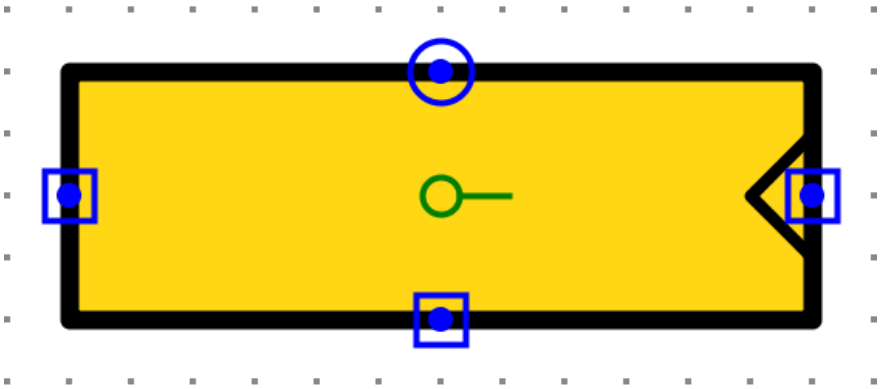

图 22

电路引脚如表 6 所示。

信号	I/O	位宽	说明
Clock	输入	1	时钟脉冲，上升沿有效
Din	输入	16	寄存数据输入端
En	输入	1	使能输入端，高电平有限
Q	输出	16	使能输入端，高电平有限

表 6

（2）测试图

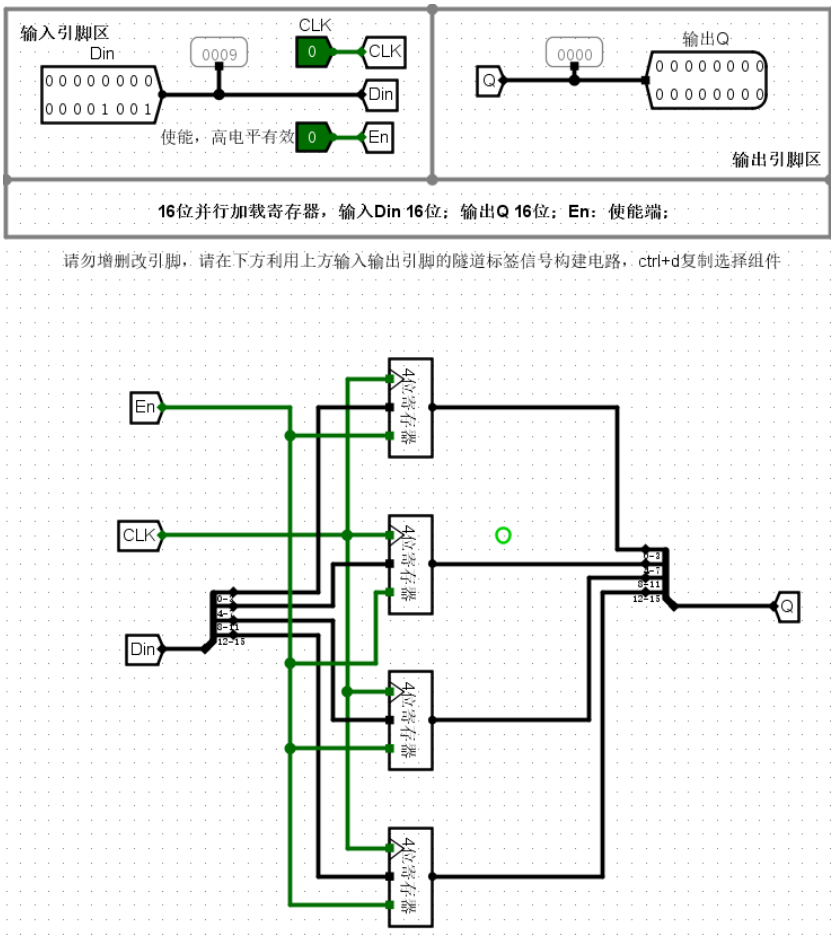
输入引脚区
Din
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1
0009
CLK
0
CLK
Din
使能，高电平有效
0
En
0000
Q
输出Q
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
输出引脚区
16位并行加载寄存器，输入Din 16位；输出Q 16位；En：使能端；
请勿增删改引脚，请在下方利用上方输入输出引脚的隧道标签信号构建电路，ctrl+d复制选择组件
En
CLK
Din
4位寄存器
4位寄存器
4位寄存器
4位寄存器
Q

图 23

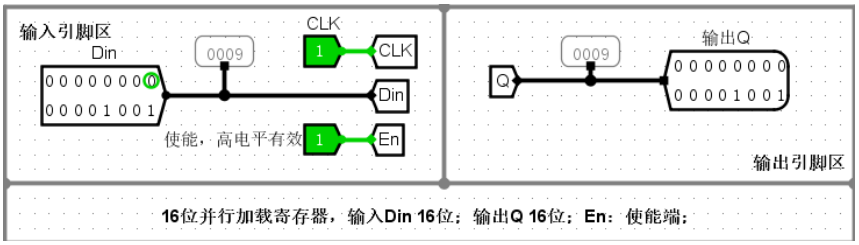
输入引脚区
Din
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1
0009
CLK
1
CLK
Din
使能，高电平有效
1
En
0009
Q
输出Q
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1
输出引脚区
16位并行加载寄存器，输入Din 16位；输出Q 16位；En：使能端；

图 24

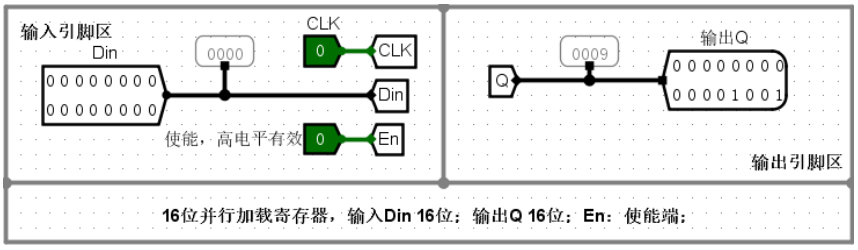
输入引脚区
Din
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0000
CLK
0
CLK
Din
使能，高电平有效
0
En
0009
Q
输出Q
0 0 0 0 0 0 0 0
0 0 0 0 1 0 0 1
输出引脚区
16位并行加载寄存器，输入Din 16位；输出Q 16位；En：使能端；

图 25

（3）测试分析

输入 Din=0009,En,CLK 均为 0 时，输出 Q 为 0；

保持 Din=0009 不变，En，CLK 均为 1 时，输出 Q=0009；

之后输入 Din 变为 0000，但 En，CLK 均为 0 时，输出 Q 仍为 0009；

综上电路功能正确。

2.2.5 四位 BCD 计数器

（1）电路图

a. 内部结构电路如图 26 所示。

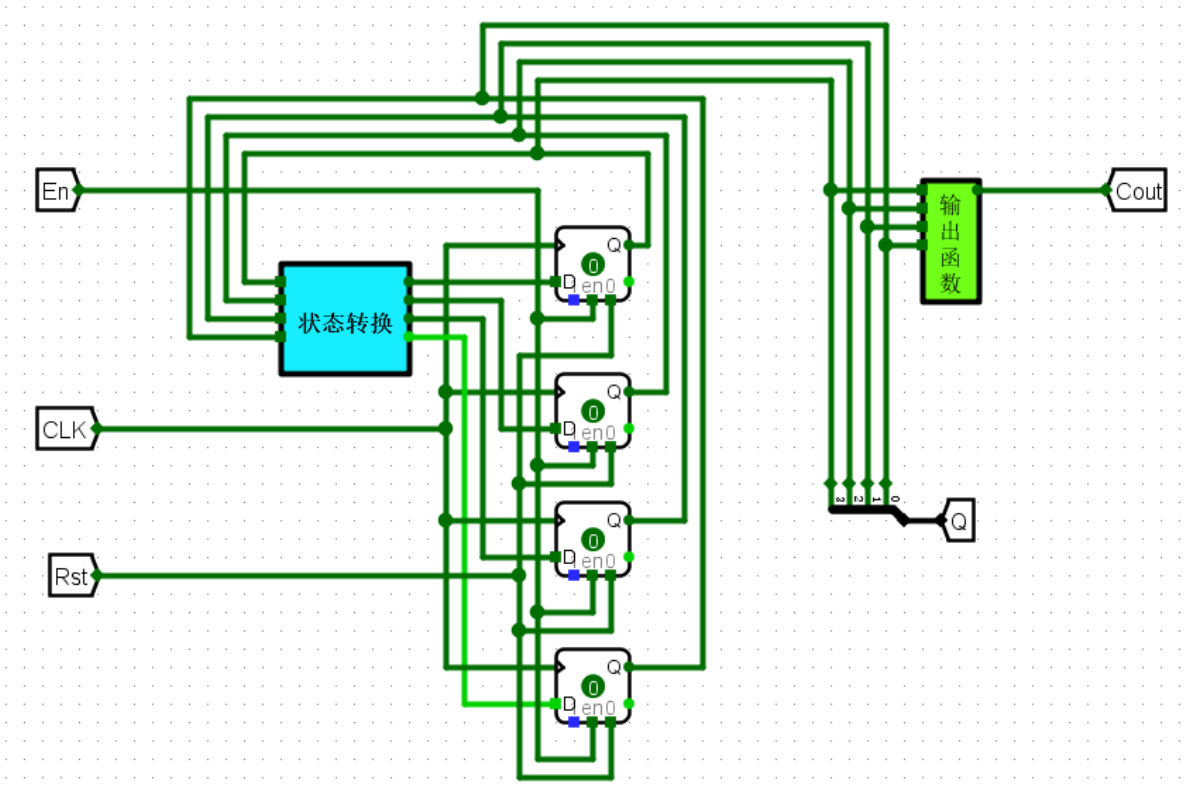

图 26

b. 封装电路图如图 27 所示。

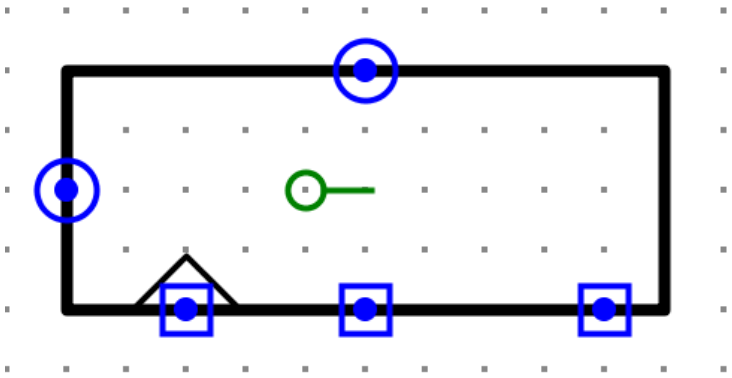

图 28

电路引脚如表 7 所示。

信号	I/O	位宽	说明
CLK	输入	1	时钟输入
Rst	输入	1	异步复位信号，为 1 时 Q=0
En	输入	1	使能端，为 1 时进行计数
Q	输出	4	计时器计时输出
Cout	输出	1	进位输出，数到 9 时输出为 1

表 7

（2）测试图

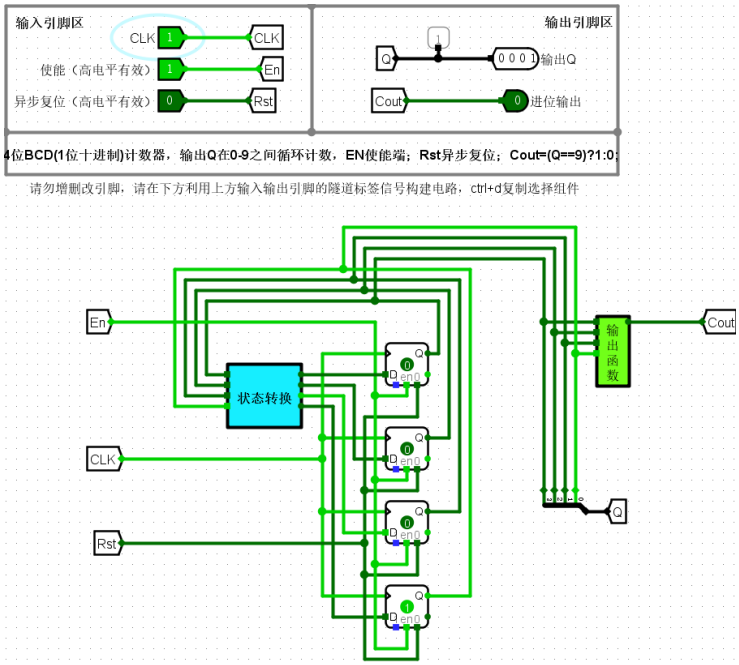

图 29

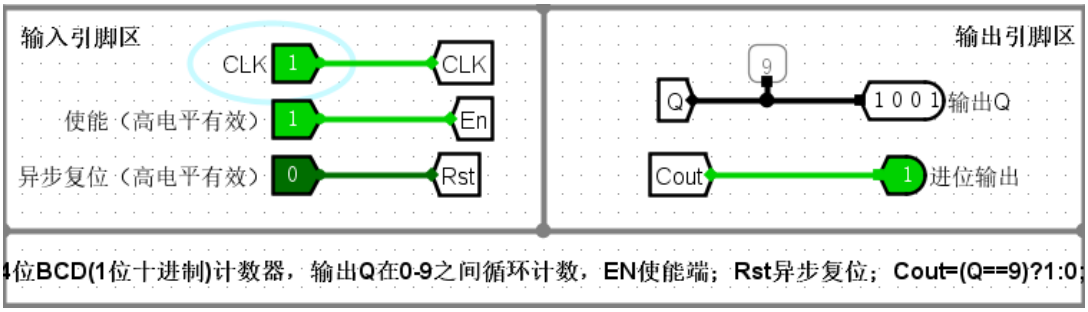

图 30

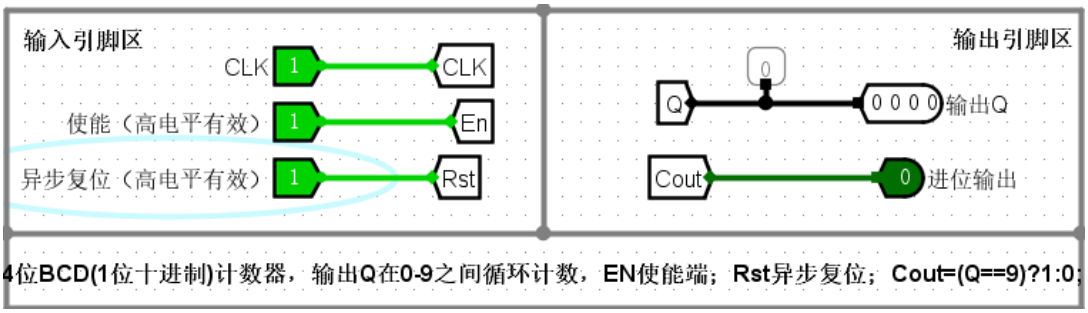

图 31

（3）测试分析

输入 En 为 1，Rst 为 0，计数器于每个时钟周期输出 Q=Q+1，当 Q=9 时，进位 Cout=1，之后设置异步复位 Rst=1，电路还原，输出 Q=0。

2.2.6 码表计数器

（1）电路图

a. 内部结构电路如图 32 所示。

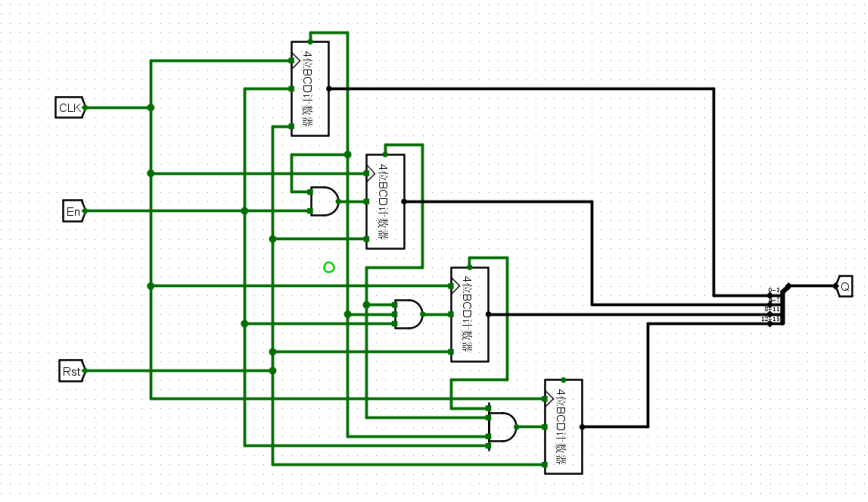

图 32

b. 封装电路图如图 33 所示。

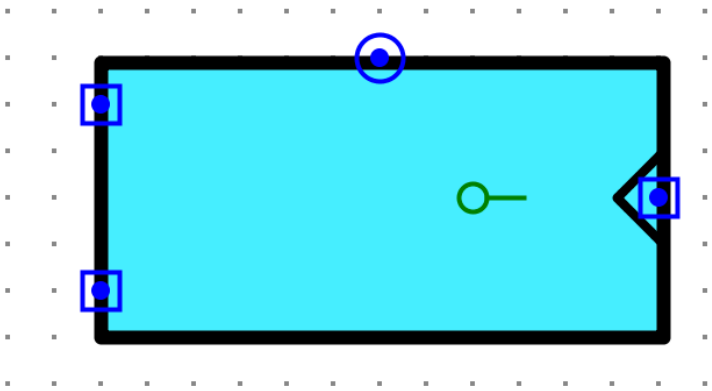

图 33

电路引脚如表 8 所示。

信号	I/O	位宽	说明
CLK	输入	1	时钟输入

Rst	输入	1	异步复位信号，为 1 时异步清零
En	输入	1	使能端，为 1 时进行计数
Q	输出	4	计时器计时输出

表 9

（2）测试图

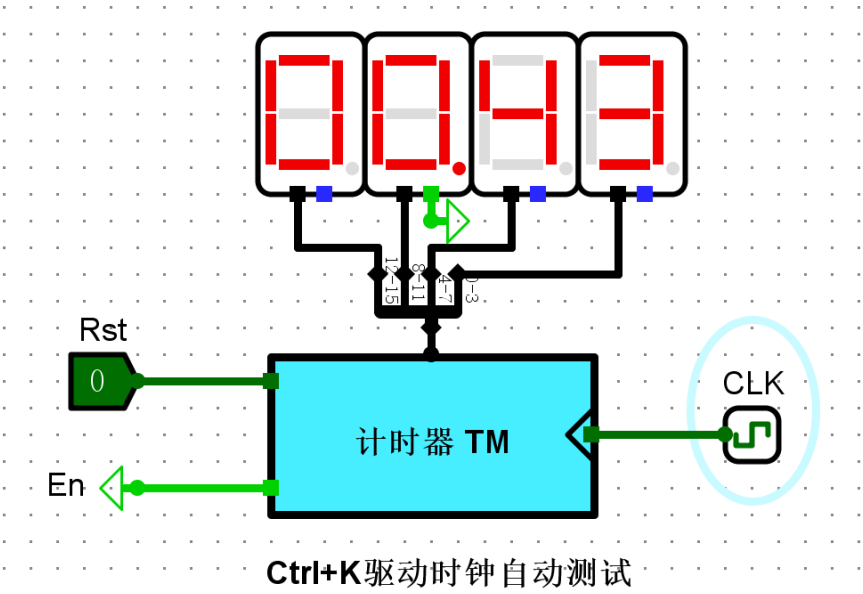

图 34

（3）测试分析

在码表测试器自动电路中，设置时钟为 Ticks Enabled，经观察，该码表测试器可以正确显示 00.00-99.99 的任意时间数据，并可以不断循环，从 99.99 再变回 00.00。由以上可以确定电路构建成功。

2.2.7 码表显示驱动

（1） 电路图

a. 内部结构电路如图 35 所示。

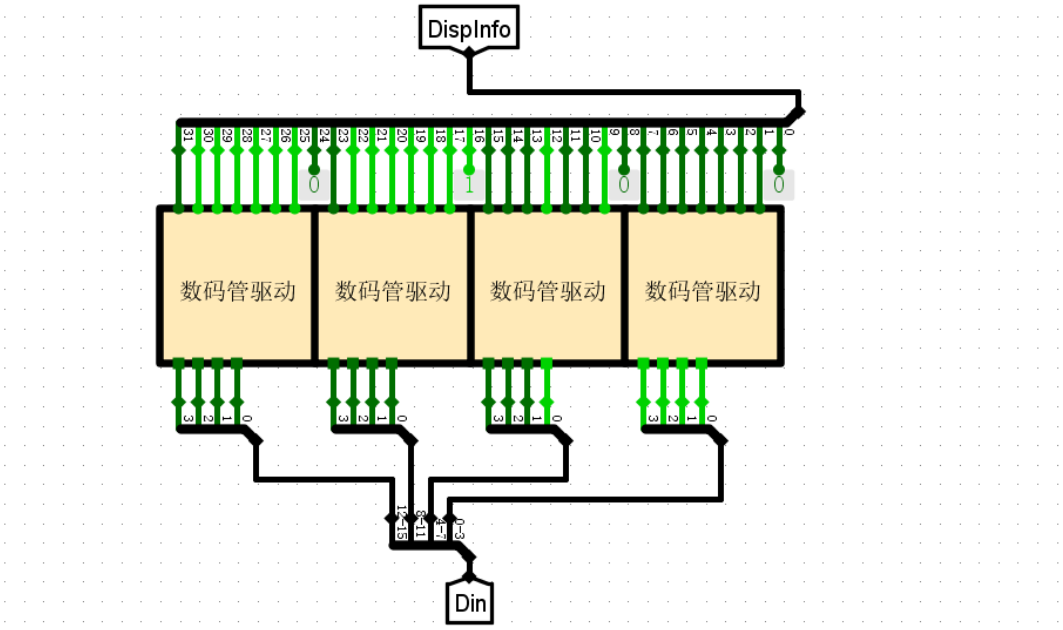

图 35

b. 封装电路图如图 36 所示。

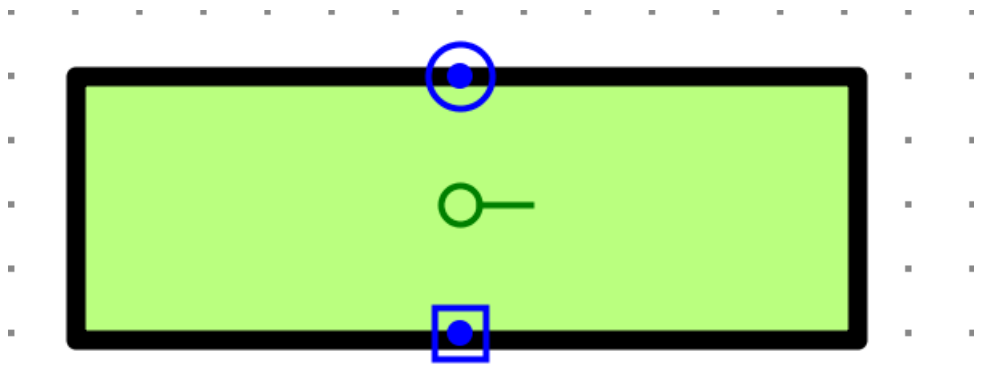

图 36

电路引脚如表 9 所示。

信号	I/O	位宽	说明
Din	输入	16	4 位十进制（16 位 BCD）
DispInfo	输出	32	4 个 8 段（7 段+小数点）数码驱动信号

表 9

（2）测试图

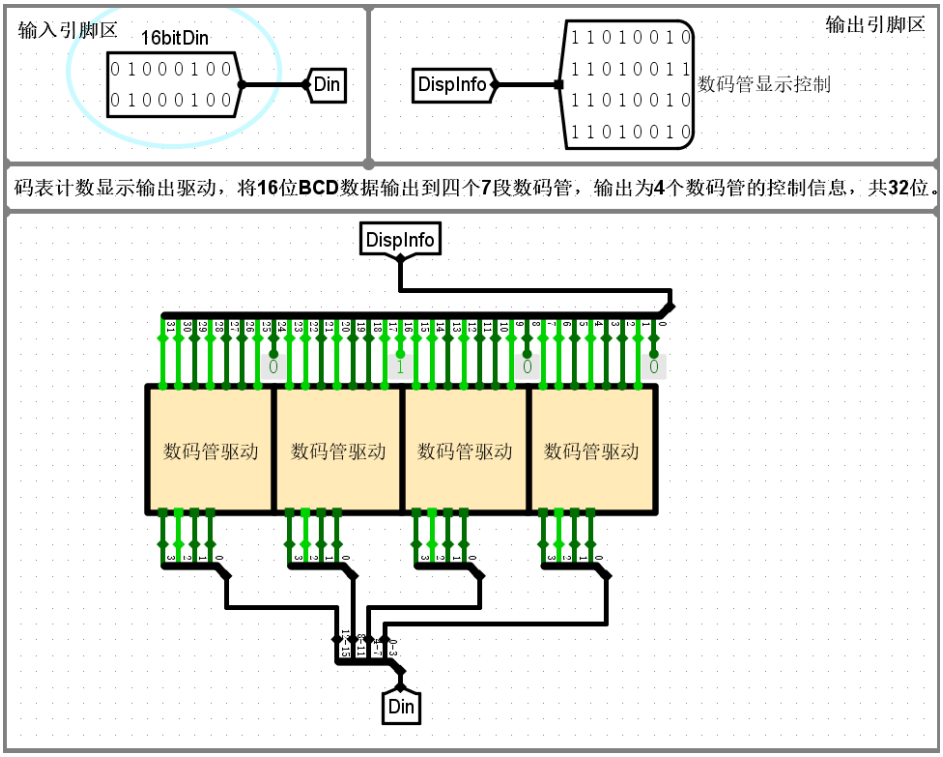

图 37

（3）测试分析

输入 Din 为 0100-0100-0100-0100 时，输出 DispInfo 为 11010010-11010011-11010010-11010010，结果正确。

2.2.10 码表控制器

（1）电路图

a. 内部结构电路如图 40 所示。

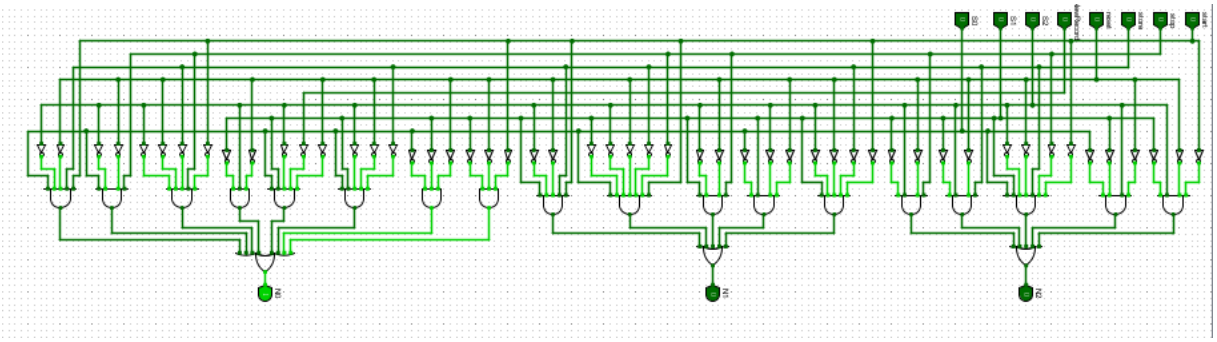

图 38 码表控制器状态转换

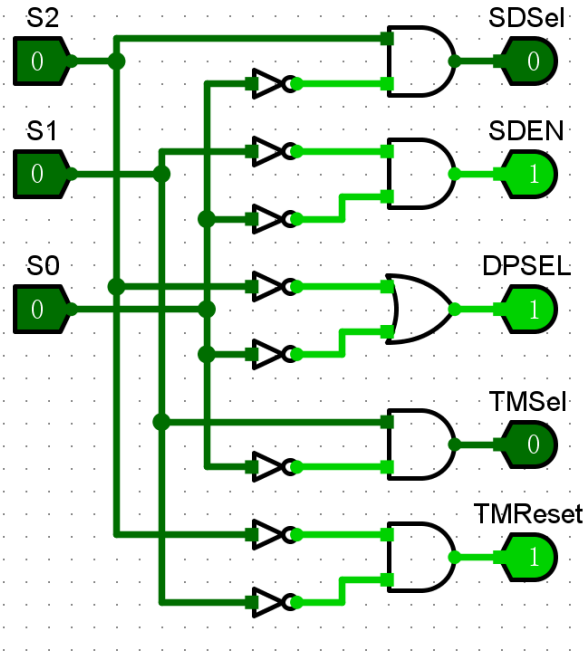

图 39 码表控制器输出函数

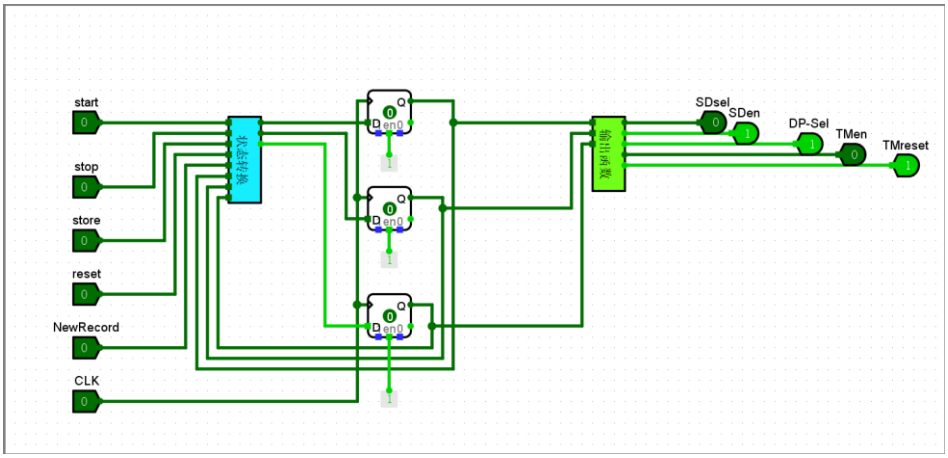

图 40 码表控制器

b.封装电路图如图 41 所示。

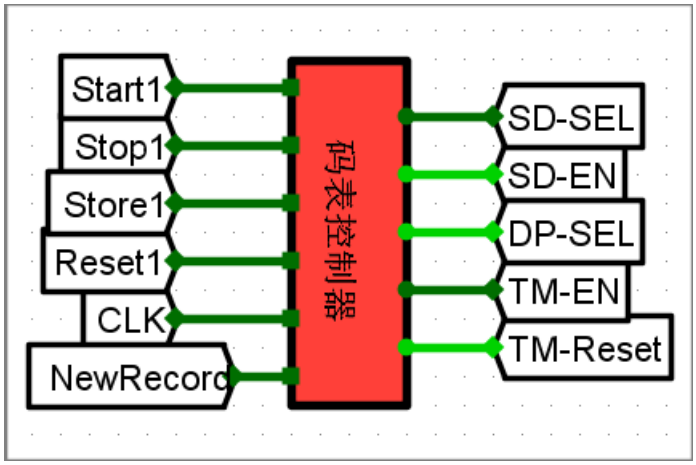

图 41

电子引脚如表 10 所示。

信号	I/O	位宽	说明
CLK	输入	1	时钟脉冲
Start	输入	1	开始计时信号
Stop	输入	1	停止计时信号
Store	输入	1	存储计时记录信号
Reset	输入	1	即使复位信号，记录恢复为 99.99
Newrecord	输入	1	新的最好成绩记录信号
Sdel	输出	1	最好成绩记录的选择信号
Sden	输出	1	保存最好成绩记录的寄存器复位信号
Dpsel	输出	1	显示计时成绩记录的选择信号
Tmen	输出	1	码表计数器的使能信号
TMreset	输出	1	码表计数器的复位信号

表 10

2.2.10 运动码表

（1） 电路图

a. 内部结构电路如图 42 所示。

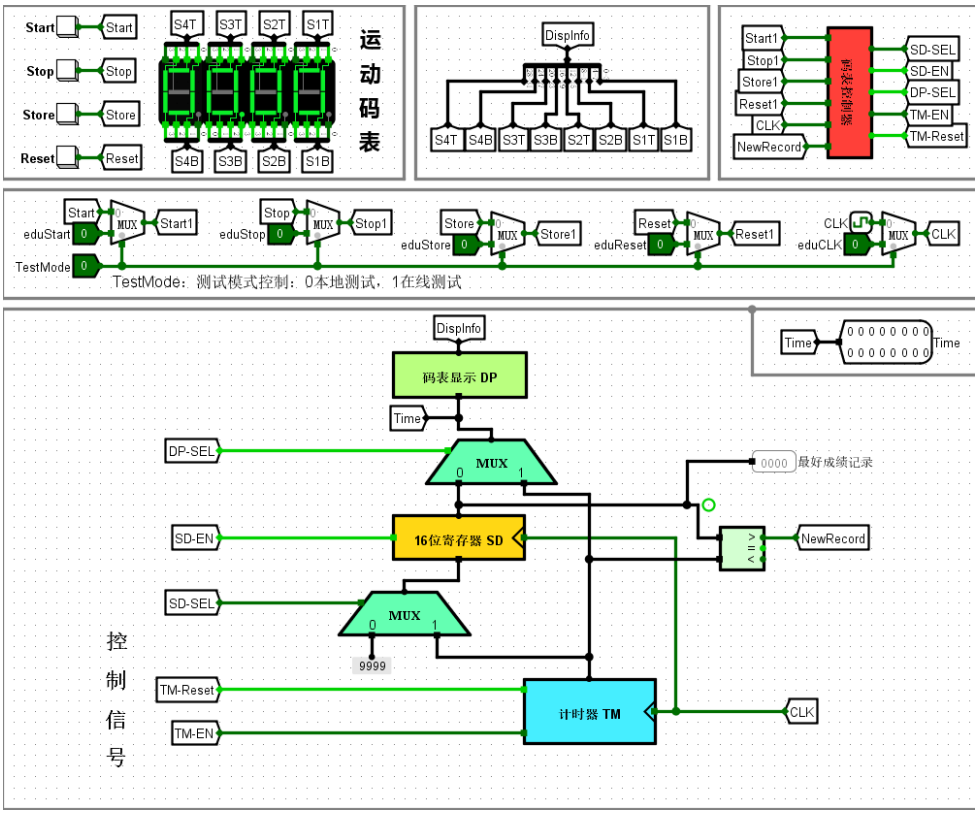

图 42

b. 封装电路图如图 43 所示。

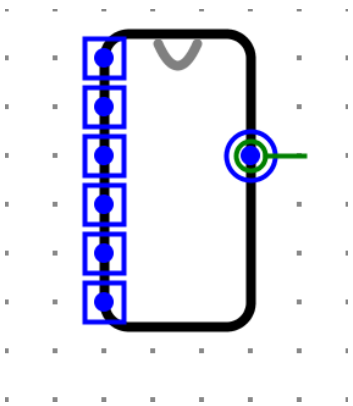

图 43

电路引脚如表 11 所示

信号	I/O	位宽	说明
CLK	输入	1	时钟脉冲
Start	输入	1	开始计时信号
Stop	输入	1	停止计时信号
Store	输入	1	存储计时记录信号
Reset	输入	1	即使复位信号，记录恢复为 99.99
Time	输出	16	计时成绩或者成绩记录

表 11

（2）测试图

点击 Start，运动码表开始计时。

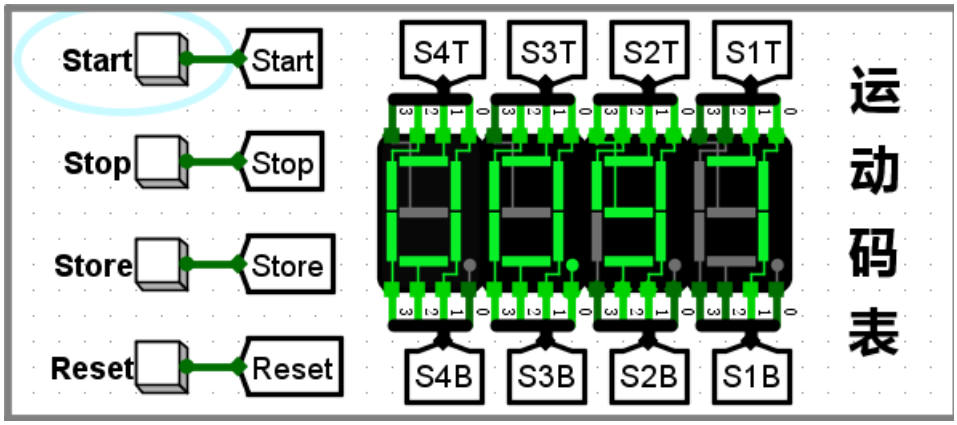

图 44 开始计时

点击 Stop，运动码表暂停计时。此时最好成绩记录为 9999。

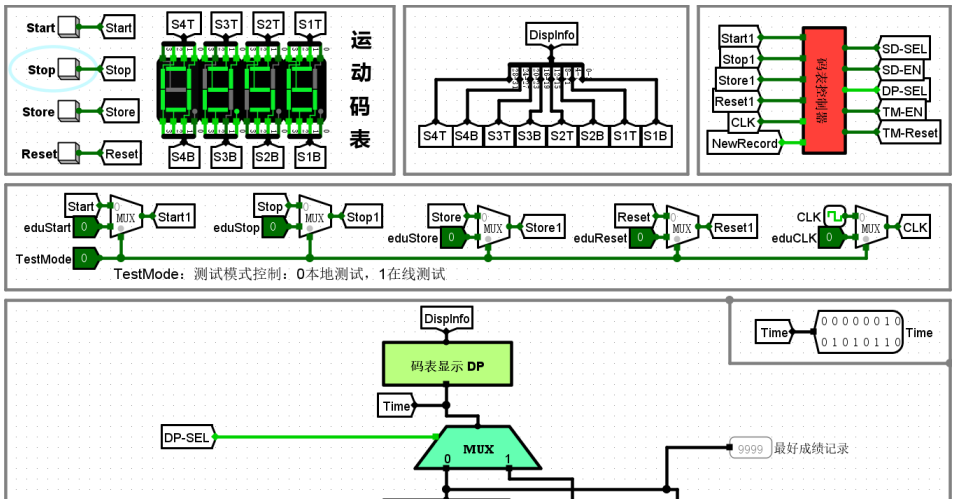

图 45 停止计时

点击 Store，运动码表的值存储到寄存器 SD 中，同时最好成绩记录发生改变。

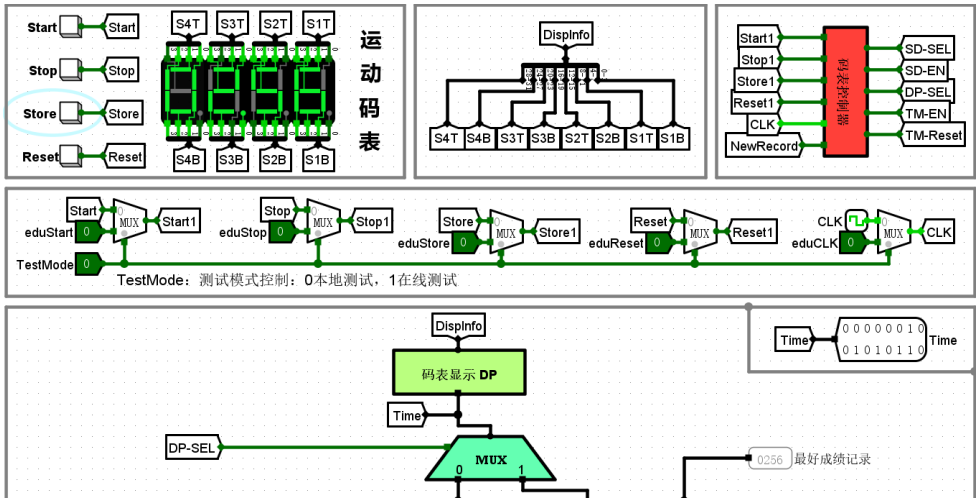

图 46 存储更新记录

再点击 Start，运动码表再次开始计时。

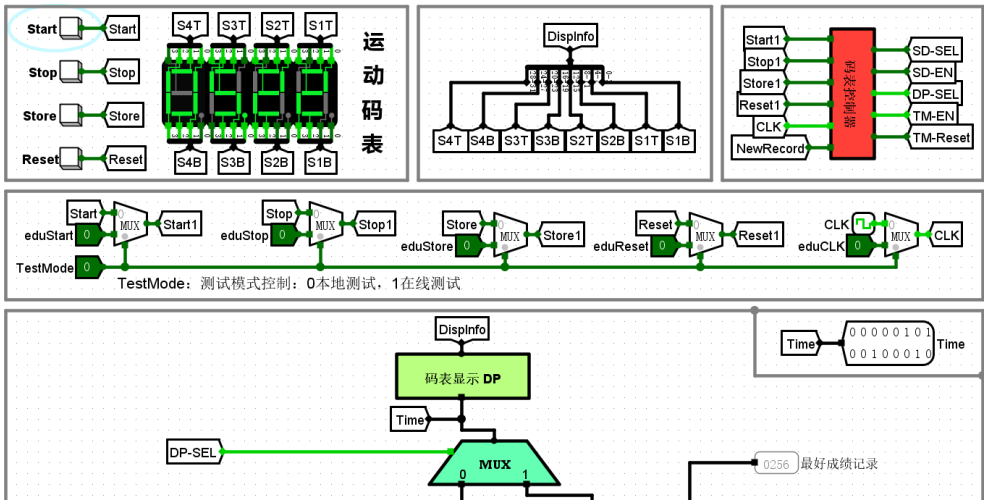

图 47 再次计时

再次点击 stop,码表停止计时。

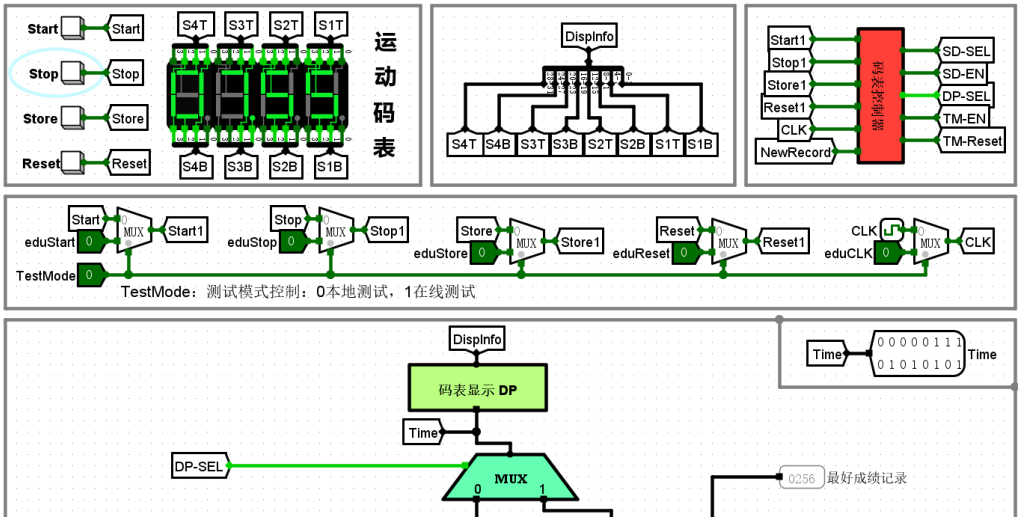

图 48 再次停止计时

点击 Store，此时运动码表显示的值比最好成绩记录大，所以运动码表显示最好成绩记录。

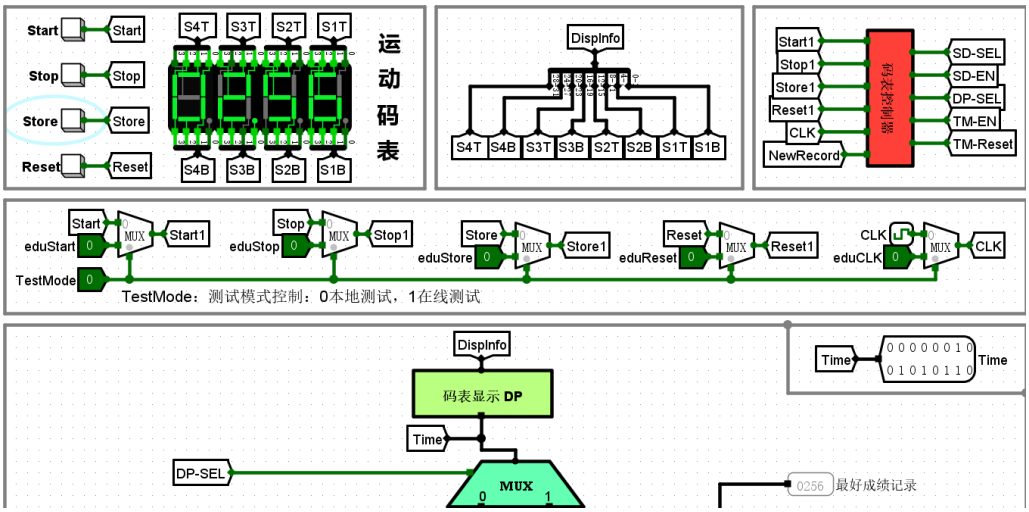

图 49 存储不更新记录

点击 Rst 码表恢复初始状态。

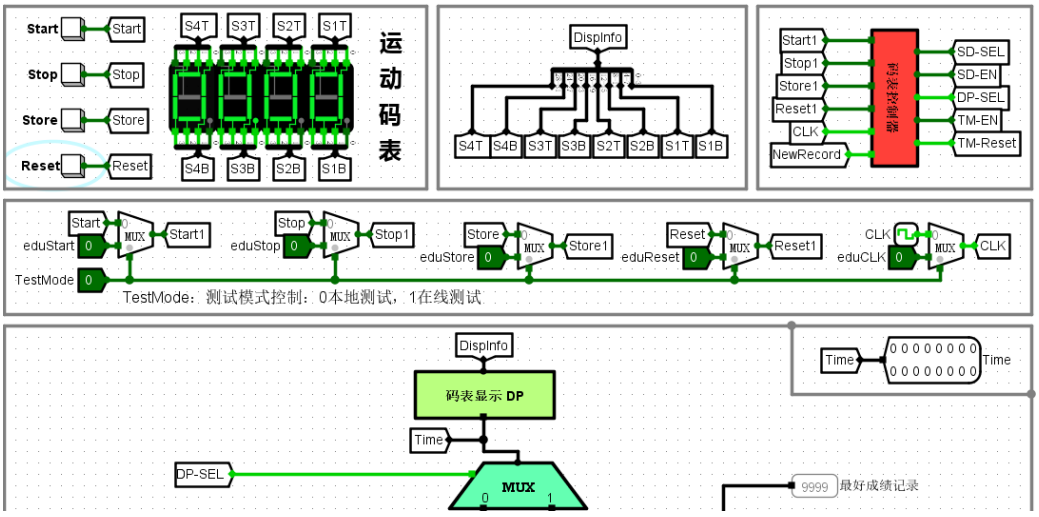

图 51 恢复初始状态

（3）测试分析

通过以上验证，可以确定该运动码表电路完成了实验所需的所有要求。

3.设计总结与心得

3.1 实验总结

通过实验，我将课本学习的组合逻辑电路，同步时序逻辑电路，异步逻辑时序电路等知识在 logosim 平台上进行了运用，完成了一个小型数字系统运动码表的构建。

3.1.1 遇到的问题及处理

在进行 16 二路选择器电路连线时，由于连线过多，导致排线混乱，造成了一线线路短接。自己起初没有在部件之间预先留有充足的空间来排线，导致后面线路挤在一起。将各部件分开重新布局后解决了这个问题。并且在此之后的实验里面，我都会先检查部件的布局之后在进行排线，减去了不少麻烦。

在设计码表控制器的状态转换时，一共设计了 6 个状态。但是起初自己在考虑状态转换时考虑并不全面，导致出错，同时我认为这也是整个实验最难的一部分。最后，我重新分析了一遍 6 个状态之间的转换关系，最终解决了问题。

3.1.2 设计方案存在的不足

实际生活中运动码表具有记录和显示若干骑行数据的功能，比如当前速度、骑行里程、骑行时间等。

本次实验中只实现了运动码表最基本的计时功能。并且码表的测量范围过小，只有一分钟。此外，码表只能存储当前最小时间，不能存储多组数据，现实中码表一般具有存储多组数据的功能，比如体育老师在体育课上测量一组学生的跑步成绩时可以用到。

此外，可以结合传感器实现当前速度测量等一些更为复杂的功能。

3.2 实验心得

在整个实验过程中，数字码表系统被拆分为一个个小的模块单独实现，循序渐进，自己在做实验的过程中体会到了从零开始，完成一个数字码表的乐趣。

Logism 可以根据真值表或者逻辑表达式生成电路图，利用好这一功能，能够为电路设计免去不少的麻烦。同时，选择怎样生成电路图也很重要。比如在设计比较器时，直接写出 Great，Equal，Less 的逻辑表达式要比利用真值表生成电路图要更加省事。

在电路图连线的过程中需要我们的细心与耐心，尤其是像 16 位的二路选择器这种线路密集的电路，更需要认真排线，需要我们耐心将线路连接正确同时兼具你美观性。

在面对码表控制器的转移时，起初会觉得无从下手。但是遇到这种情况，不能简单的直接放弃。而是要联系课本，由课本电路设计的方法进一步思考，慢慢的便会有了思路。

3.3 意见与建议

本次实验采用分层设计，模块设计构造运动码表数字系统。难度从易到难，将课本中一些经典的数字电路如寄存器，7 段数码管显示器实现了 logisim 的仿真。不过个人认为数字码表设计的数字电路还是有限的，可以在数字码表前补充一些数字电路的实验，扩大实验的覆盖面。

数字码表实验中，码表控制器状态转换相对前面的几个部分难度较大，状态很容易考虑不全面，希望老师可以在状态设计这一部分给予一定的指导。