華中科技大學

2022

数字电路与逻辑设计
实验报告

专　　业：计算机专业

班　　级：CS2104 班

学　　号：U202115452

姓　　名：刘凯欣

电　　话：13871967628

邮　　件：1474867893@qq.com

完成日期：2022.12.15

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

对于实验中的运动码表，结合实际需求和功能。我们基本需要实现的是图 2.1-1 中的所有内容。

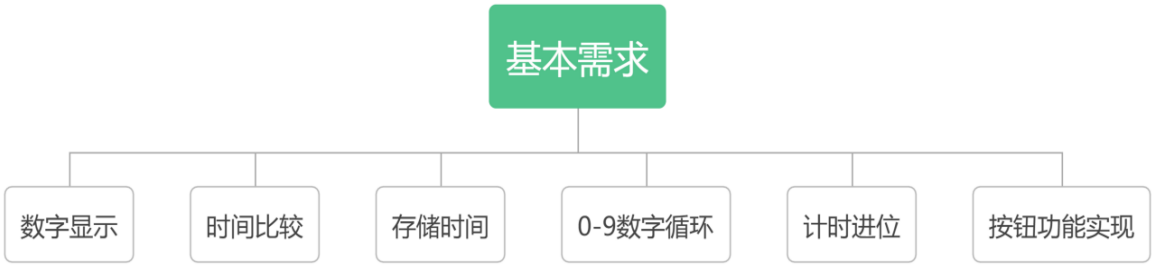


图 2.1-1 实验基本需求

在方案设计部分，我将结合各个部分的功能，围绕实现功能的器件展开，对其原理进行设计和解释。在介绍时，我将采用表达式，真值表，图形对照等多种方式以便更好地解释。

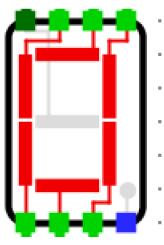
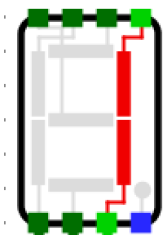
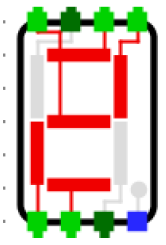
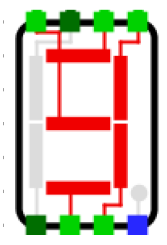
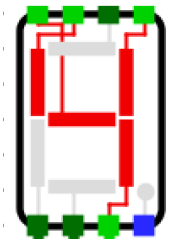
2.1.1数字显示

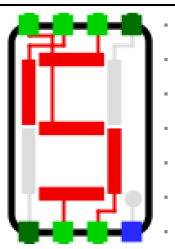
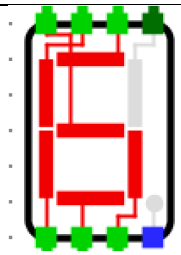
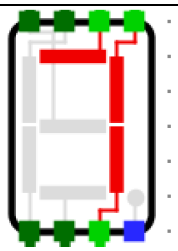
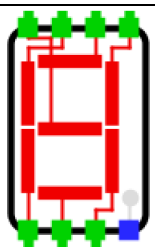
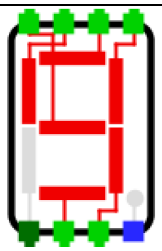
数字显示基本需求是，实现码表的数字显示。其构造包括“七段数码管”和“码表显示”。

(1) 七段数码管

实验中，“七段数码管”执行时间显示的功能，是运动码表时间显示的基本单元。在对应的时间中，分别显示对应的数字，见表格 2.1-1

表格 2.1-1 “七段数码管”基本显示

0	1	2	3	4
				

5	6	7	8	9
				

“七段数码管”中，各个显示部位的对应连接端口，见右图 2.1-2。

其中 Seg_1 至 Seg_7 是数字显示的部分，Seg_8 是小数点显示的部分。

根据表格 2.1-1 的显示，我们使用 4 位二进制数作为输入（$2^4 > 10$，恰好可以容纳所有状态）。其中 Seg_1-Seg7 作为输出，Seg_8 显示小数点（在码表显示驱动中使用）

以此得出数码管电路的真值表如图 2.1-3。其中，计时为 10 至 16 的部分，由 0-9 的状态得出的表达式确定。

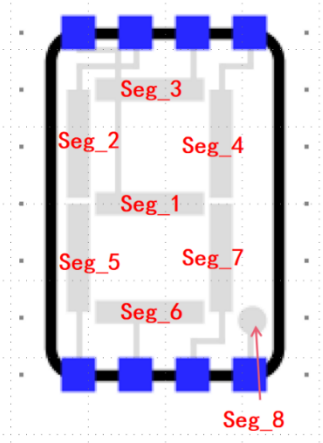


图 2.1-2 七段数码管

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 2.1-3 数码管电路真值表

上述输入输出即为器件的引脚。即 4 个一位输入引脚，8 个一位输出引脚。

(2) 码表显示

“码表显示”是由四个“七段数码管”组合实现的。

单个“七段数码管”是 4 个一位输入和 8 个一位输出，那么“码表显示”经过整合后，是 1 个 16 位输入和 1 个 32 位输出，且使用到了小数点。其中，输入代表的是当前十进制的计时，输出代表的是七段数码管显示部分。

下面图 2.1-4 是码表显示器的基本构造，图 2.1-5 是单个数码管的接线关系。

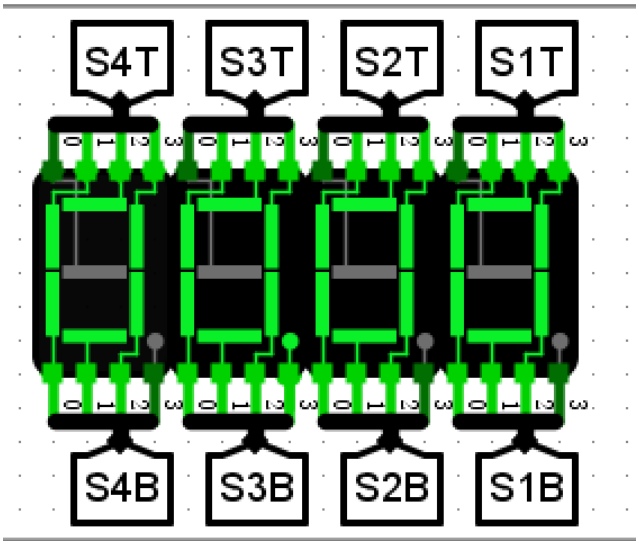


图 2.1-4 数码显示器外观构造

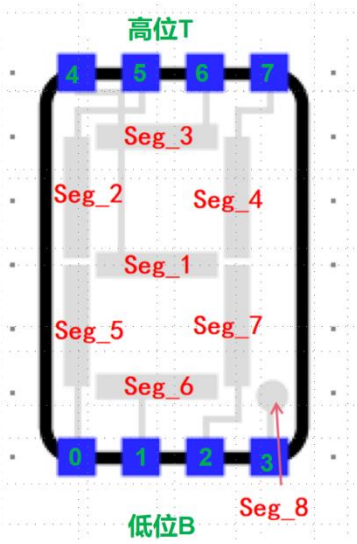


图 2.1-5 数码显示器连接

“码表显示器”与 “七段数码管”不同的，是在接线的对应关系上发生了改变，详细来说，高四位 SiT 分别对应单个“七段数码管”的“Seg_1-Seg_4”，低四位 SiB 分别对应单个“七段数码管”的“Seg_5-Seg_8”，在接线时要注意对应连接。

“码表显示”对应的有两个引脚，一个是 16 位输入引脚，一个是 32 位输出引脚，

2.1.2时间比较

“时间比较”的基本需求是，在码表运行中对相应功能中涉及到的时间大小比较进行抉择，并正确输出。其构造包括“二路选择器”和“无符号比较器”。如在码表中的 store 功能中，“二路选择器”和“无符号比较器”协同进行时间数据的选择、比较与输出。

(1) “二路选择器”

“二路选择器”是时间信号选择的关键器件。其构造包括“二路选择器（1 位）”和“二路选择器（16 位）”。

a. “二路选择器（1 位）”

1 位二路选择器，有 2 个 1 位输入 X0,X1，1 个 1 位选择 Sel，1 个 1 位输出 Out。对应的引脚有四个，与输入输出和选择分别对应。

其功能的基本表达式为$Out = （Sel == 0）?X0:X1$。

b. “二路选择器（16 位）”

16 位二路选择器，有 2 个 16 位输入 X,Y，1 个 1 位选择 Sel，1 个 16 位输 Out。引脚与输入输出和选择分别对应。

其功能的基本表达式为$Out = （Sel == 0）?X:Y$。

在构建 16 位二路选择器时，沿用 1 位二路选择器，将 X 和 Y 的 16 位输入一一拆分，并进行一一比较选择，最后汇总为 16 位输出。

(2) “无符号比较器”

“无符号比较器”是时间信号比较和输出结果的关键器件。其构造包括“4 位无符号比较器”和“16 位无符号比较器”。

a. “4 位无符号比较器”

在介绍“4 位无符号比较器”前，先介绍一下“1 位无符号比较器”。

“1 位无符号比较器”有 2 个 1 位输入 X 和 Y，3 个 1 位输出“Great”“Equal”“Less”，其功能可表示为“>”“=”“<”。当 X>Y 时，触发功能“>”，输出“Great”为 1。其他情况可以以此类推。

“4 位无符号比较器”相比于“1 位无符号比较器”，输入为 2 个 4 位的二进制数(从高位到低位可记为 X3X2X1X0 和 Y3Y2Y1Y0)，输出仍为 3 个 1 位输出“Great”“Equal”“Less”。

构造电路的表达式见表格 2.1-2。

表格 2.1-2，4 位无符号比较器输出表达式

$Great$	$X3 > Y3 + (X3 = Y3)\&(X2 > Y2) + ((X3 = Y3)\&(X2 = Y2)\&(X1 > Y1)) + ((X3 = Y3)\&(X2 = Y2)\&(X1 = Y1))\&(X0 > Y0))$
$Equal$	$X3 = Y3\&X2 = Y3\&X1 = Y1\&X0 = Y0$
$Less$	$X3 < Y3 + (X3 = Y3)\&(X2 < Y2) + ((X3 = Y3)\&(X2 = Y2)\&(X1 < Y1)) + ((X3 = Y3)\&(X2 = Y2)\&(X1 = Y1))\&(X0 < Y0))$

其中 1 位的比较会用到“1 位无符号比较器”。

引脚与输入输出相对应。

b. “16 位无符号比较器”

“16 位无符号比较器”相比于“4 位无符号比较器”，输入为 2 个 16 位的二进制数(从高位到低位可记为 X15~X0 和 Y15~Y0)，输出仍为 3 个 1 位输出“Great”“Equal”“Less”。

“16 位无符号比较器”的构造之与“4 位无符号比较器”，和“4 位无符号比较器”之与“1 位无符号比较器”相同。故可依据“4 位无符号比较器”的基本构造和表达式去实现“16 位无符号比较器”。期间也需要用到“4 位无符号比较器”。

引脚与输入输出相对应。

2.1.3存储时间

“存储时间”的基本需求是，存放码表运行到某次操作时的时间信息。其构造包括“并行加载寄存器”。

“并行加载寄存器”又分为“4 位并行加载寄存器”和“16 位并行加载寄存器”。

(1) 4 位并行加载寄存器

功能为存放和输出从输入端输入的 4 位二进制数。有 1 个 4 位的输入 Din，1 个 4 位的输出 Q，以及 1 个 1 位的使能端 En。

在电路中，D 触发器是一个具有记忆功能的信息存储器件，我们此处便是使用 D 触发器完成寄存功能。我们对输入 Din 的每一位并行处理，使用四个并行的 D 触发器，以实现 4 位二进制数的存储功能。

引脚与输入、使能、输出一一对应。

(2) 16 位并行加载寄存器

16 位并行加载寄存器是在 4 位寄存器的基础上，对输入和输出数据进行扩大，使得其实现存储 16 位二进制数的功能。其他逻辑与结构基本一致。

有 1 个 16 位的输入 Din，1 个 16 位的输出 Q，以及 1 个 1 位的使能端 En。

引脚与输入、使能、输出一一对应。

2.1.4“0-9”数字循环

“‘0-9’数字循环”是对数字显示功能进一步实现，在计数时，码表的显示数字将在“0-9”之间循环，这就涉及到“4 位 BCD 计数器”。

实现“4 位 BCD 计数器”的功能，我们需要完成两个主要的函数：状态转换；输出函数。

(1) 状态转换

状态转换是对码表计数进行设计。该函数的主要功能是完成 0-9 的数字循环。

由于$2^4 > 10$,故函数是四位输入（S3-S0），四位输出（N3-N0）。

状态转化的真值表如图 2.1-6。

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	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

图 2.1-6 BCD 计数器状态转换

当输入为 0-8 时，输出自动加 1；当输入为 9 时，输出为 0. 余下情况是由 0-9 的表达式的所确定的，在实验中一般不会遇到。

(2) 输出函数

输出函数是当输入为9时进行的附加处理。在码表运行中，一旦某一位运行到“9”，表明这一位的上一位即将进 1。输出函数输出 Cout 便是记录进位。仅当输入为“9”时，Cout 输出 1。

2.1.5计时进位

“计时进位”，是处理各位之间有进位的情况。这就涉及到“码表计数器”。、介绍原理之前，我们先看下一个特殊的进位情况，如下图 2.1-7。

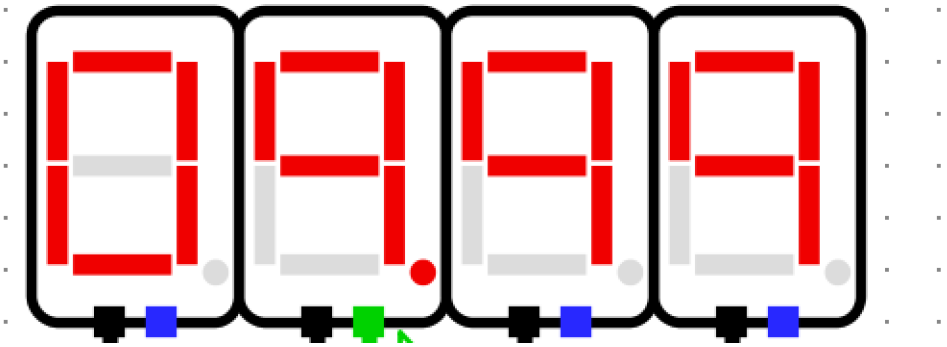


图 2.1-7 特殊进位情况

此时码表运行到 9.99 秒，到下一步的显示应当是 10.00 秒。

这里，百分位的进位是在“0-9”的循环之中产生，这在“0-9 数字循环”中已有处理，由一个“4 位 BCD 计数器”可以实现。

十分位的进位，一是自身出现数字“9”，二是低位即将产生进位。

同理，个位的进位，一是自身出现数字“9”，二是十分位和百分位均即将产生进位。

从而可以得出，高位的进位需要低位均有进位，且高位收到自身数字为“9”的使能信息。

最终，我们给出这个器件的输入输出等引脚：

1 个 16 位的输入，1 个 16 位的输出以及最高位的 1 位进位输出，以及使能段。

2.1.6按钮功能实现

“按钮功能实现”主要是实现码表各个基本功能。涉及到“码表控制器”。

要实现“码表控制器”，主要完成两个部分的函数，一个是“状态转换函数”，一个是“输出函数”。

“状态转换函数”是依据现态和按下的按钮功能，确定出次态；“输出函数”则是将状态转换后的次态对应到码表的实际功能。

（1）“状态转换函数”

要进行状态转移，首先我们需要确定码表需要几个状态。

已知有四个具体功能：

当按下 Start 时，计时器清零，重新开始计时；

当按下 Stop 时，计时器停止计时，显示计时数据；

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。

当按下 Reset 时，复位，计时=0.00，系统记录=99.99 秒。

在任何状态，按下 reset，要实现计时器清零和储存器复位为 99.99。

在任何状态，按下 start，要实现计时器清零且立即开始计时。

结合上述的功能与特性，我们设置有如下的状态转换。

Reset 和 start 的清零，显然不是同一个状态。

于是，我们将 reset 的复位和清零设置为一个状态（000）。

start 的清零和计时分成两个状态（001，010）。

处于 001 时，将接着直接进入 010 状态。当然，将 start 的两个状态合并也可以，此处的设计主要是与头歌测试中的状态转换进行比较。

在 010 状态下，只有按下 stop，才进入停止状态，记为 011.

在 011 状态下，只有按下 store 且 new record 的使能输入为真，进入存储状态 100，随后进入 101.

在 011 状态下，只有按下 store 且 new record 的使能输入为否，进入查看状态 101.

整体的状态转换可见图 2.1-8.

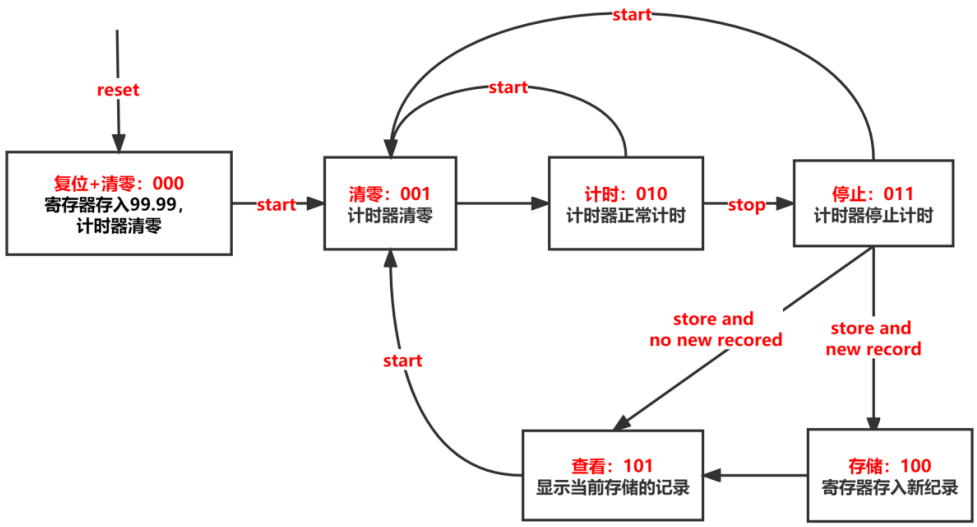


图 2.1-8 状态转换图

由此可见，一共有 6 种状态，用 3 位二进制数即可表示出来($2^3 > 6$)，同时，我

们需要有五个输入“start”“stop”“store”“reset”“new record”，均为 1 位二进制数。

对应的，输出也有 6 种状态，用 3 位二进制数即可表示。

从而输入为：1 个 3 位二进制数和 5 个 1 位二进制数。

输出为：1 个 3 位二进制数。

对应的状态转换的真值表如图 2.1-9

				输入信号								下一状态（次态）			
S2	S1	S0	现态10进制	start	stop	store	reset	NewRecord	In6	In7	In8	次态10进制	N2	N1	N0
				0			1					0	0	0	0
0	0	0	0	1			0					1	0	0	1
0	0	1	1	0			0					2	0	1	0
0	1	0	2	0	1	0	0					3	0	1	1
0	1	1	3	0		1	0	1				4	1	0	0
0	1	1	3	0		1	0	0				5	1	0	1
1	0	0	4	0			0					5	1	0	1
1	0	1	5	0			0					5	1	0	1
0	1	1	3	1		0	0					1	0	0	1
0	0	1	1	1			0					1	0	0	1
0	1	0	2	0	0		0					2	0	1	0
0	1	1	3	0		0	0					3	0	1	1
1	0	1	5	1	0	0	0					1	0	0	1
0	1	0	2	1	0		0					1	0	0	1

图 2.1-9 状态转换表

（2）“输出函数”

这里我们只需要对应好各个状态的下的码表功能即可。

有 5 个输出，见表格 2.1-3

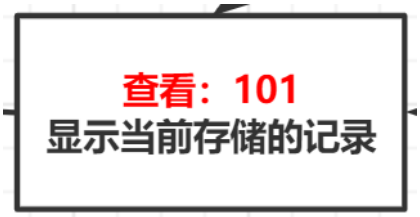
表格 2.1-3 各输出名称、位宽与功能

名称	位宽	功能
SDsel	1	最好成绩记录选择信号
SDen	1	保存最好成绩记录的寄存器的使能信号
DPsel	1	显示计时成绩记录的选择信号
TMen	1	码表计时器使能信号
TMreset	1	码表计时器复位信号

注：DPsel 为 1 时，显示当前计时；否则显示存储器计时。

各状态下输出为真的输出部分见表格 2.1-4，

表格 2.1-4 输出函数对应

	SDen：存储器更新 DPsel：计时器显示 TMreset：计数器清零
	DPsel：计时器显示 TMreset：计数器清零
	DPsel：计时器显示 TMen：计时使能信号
	DPsel：计时器显示
	SDsel：产生新成绩 SDen：存储器更新 DPsel：计时器显示信号
	无

这里的每个状态对应的输出函数部分，可以结合 2.2.7 运动码表电路图来看，更方便理解。

真值表见图 2.1-10.

当前状态(现态)				输入信号							输出						
S2	S1	S0	现态10进制	start	stop	store	reset	NewRecord			SDsel	SDen	DPsel	TMen	TMreset	Out6	Ou
0	0	0	0									1	1		1		
0	0	1	1										1		1		
0	1	0	2										1	1			
0	1	1	3										1				
1	0	0	4								1	1	1				
1	0	1	5														

图 2.1-10 输出函数真值表

从上述图表中可以得出，输入为 1 个 3 位二进制数，输出为 5 个 1 位二进制数。

2.1.7运动码表汇总实验

首先看一下上述设计方案的总逻辑关系，见图 2.1-11。

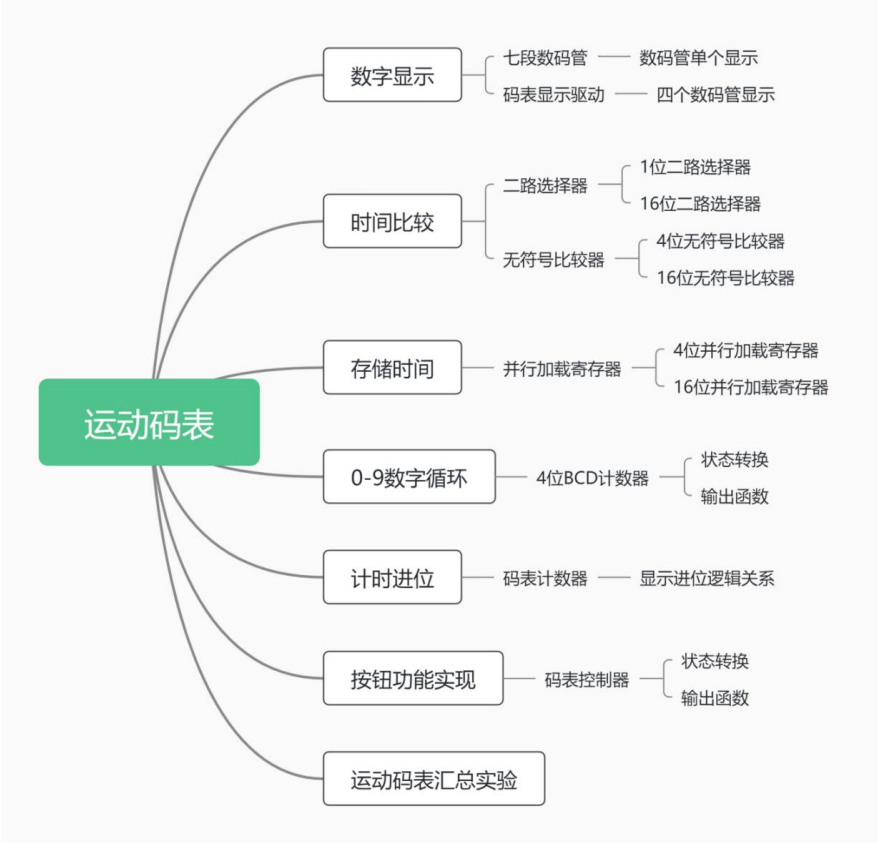


图 2.1-11 实验设计逻辑关系

运动码表的汇总，就是将前面实现的“码表显示驱动”“码表控制器”“无符号比较器”“并行加载寄存器”“码表计数器”拼装在一起即可，具体部分参照电路图。

2.2 实验过程

本部分将结合设计部分，展示设计的电路图，器件封装图，并进行测试。

2.2.1数字显示

（1）七段数码管

（a）电路图

电路图和封装器件各个引脚部分如下图。

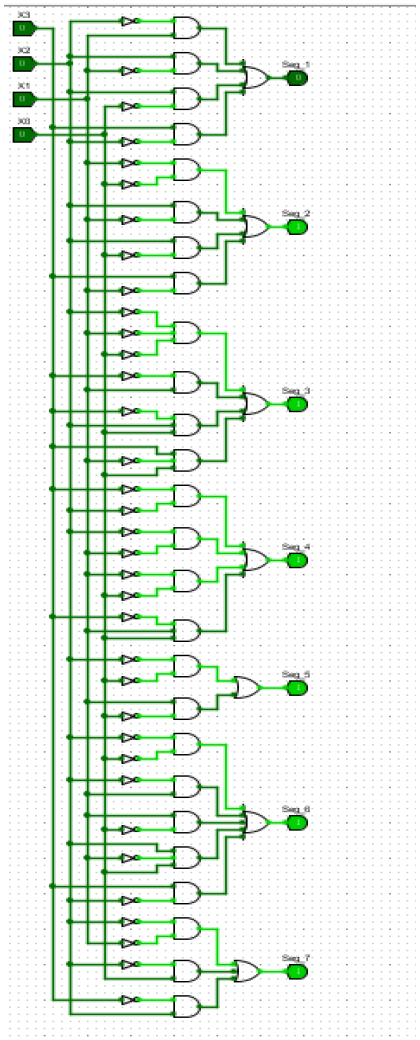


图 2.2-1 七段数码管电路图

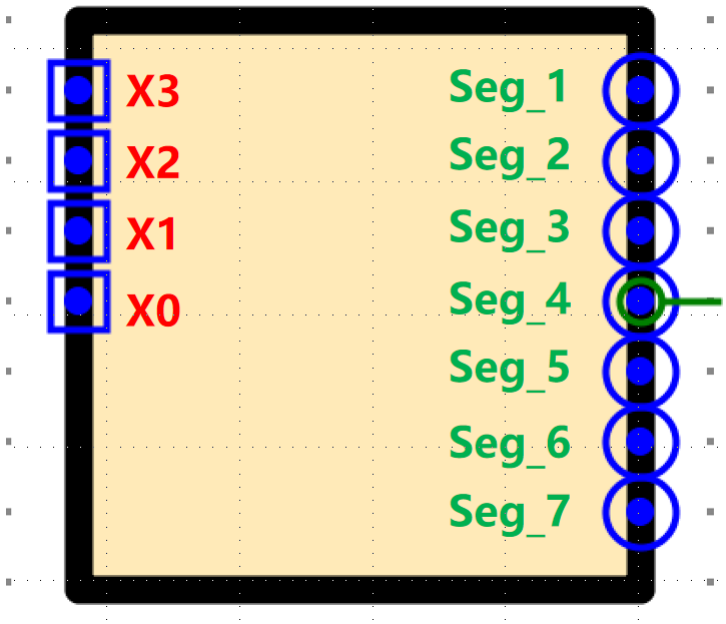


图 2.2-2 七段数码管封装图

（b）测试图

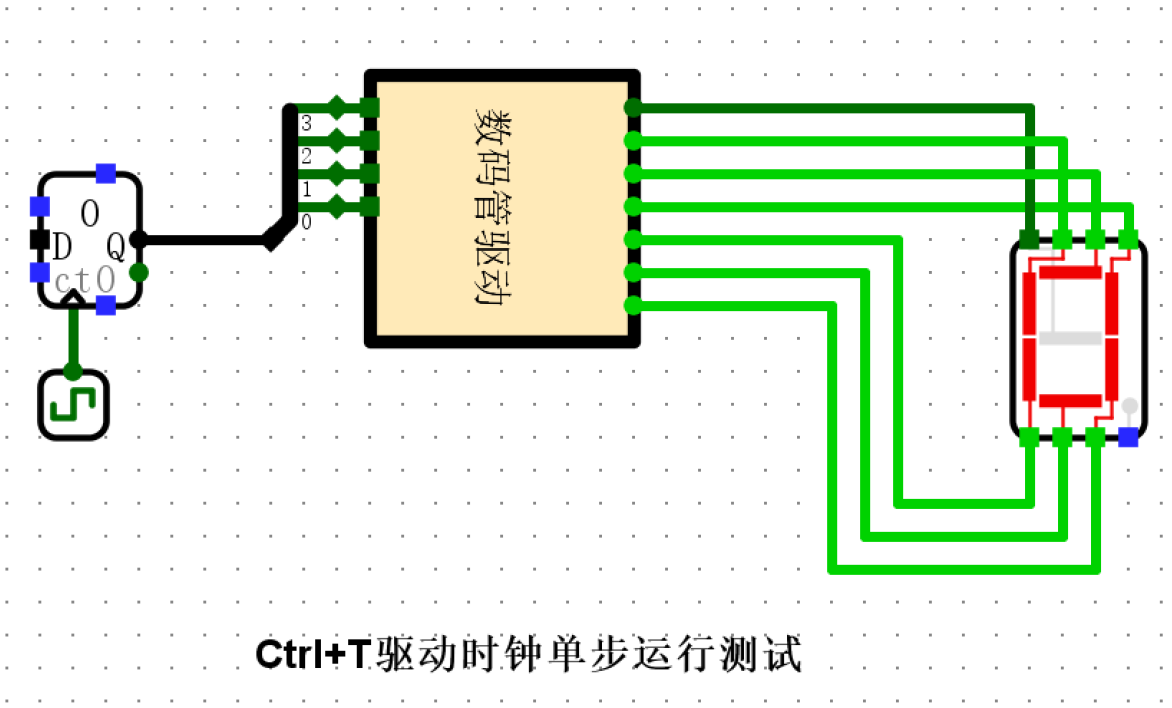


图 2. 2-3 七段数码管测试图

在 logisim 平台上，对七段数码管进行驱动测试，随着时钟脉冲信号的输入，观察七段数码管上的数字变化。

（c）测试分析

随着 clk 时钟使能端的脉冲输入，在表格 2.2-1 的显示基础上，七段数码管还会多出 6 个基本状态，这是由“0-9”状态表达式得出来的结果。这六个状态由后面的“0-9 数字循环”进一步处理，使其不会出现。这也进一步验证了七段数码管的正确性。

（2）码表显示驱动

（a）电路图

码表显示为 1 位 16 位输入和 1 位 32 位输出。由图 2.1-5 的对应关系，重新排版输出引脚中位数的分布，使得接线更为简洁美观。同时要注意到的是，码表显示中，只有个位右下角的小数点始终处于亮着的状态，于是这个小数点使能始终为 1，其他三个小数点使能始终为 0。

具体电路图见下图，图 2.2-4。

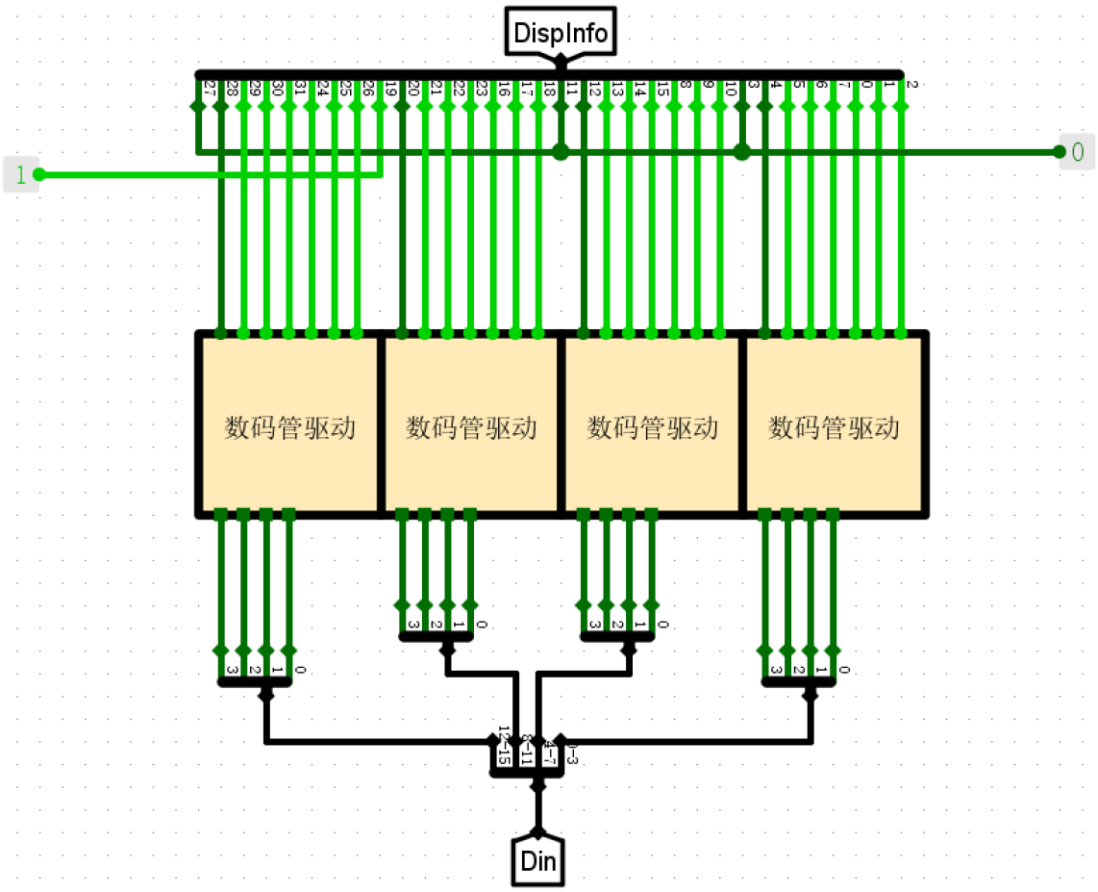


图 2.2-4 码表显示驱动电路图

电路的封装图如下图。

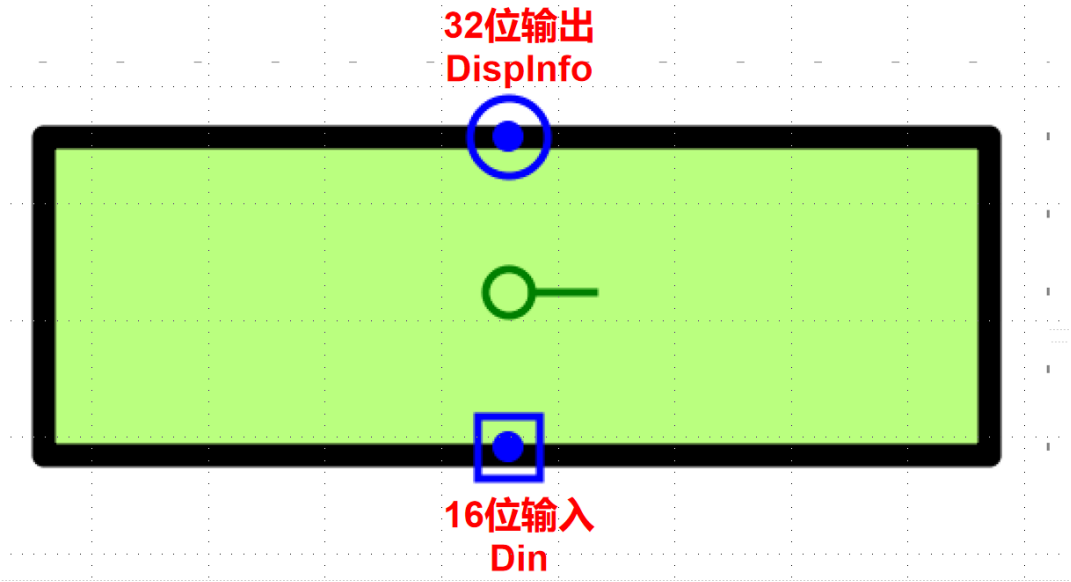


图 2.2-5 码表显示驱动封装图

（b）测试图

测试图码表显示的直观测试，应当是直接观察运动码表的运作情况，而这一部分需要涉及到最后运动码表的组合电路测试。（给定输入 Din 的序列，观察并验证输出 Displnfo 的序列也可，但在检验中较难核实其准确性）。

测试图见下图。

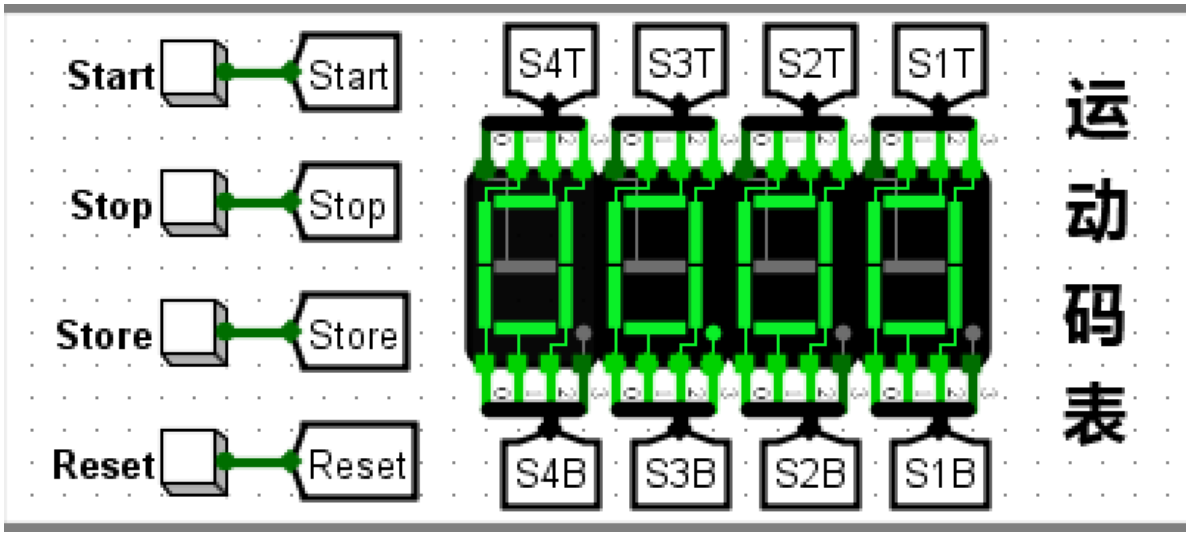


图 2. 2-6 码表显示 运动码表观察测试

在 logisim 平台上，按下 start 进行驱动测试，随着时钟脉冲信号的输入，观察码表显示上数字显示变化。

（c）测试分析

在 logisim 上运行时，运动码表上每一位都可正常顺序显示“0-9”，表明无误。

2. 2. 2时间比较

（1）2 路选择器（1 位）

（a）电路图

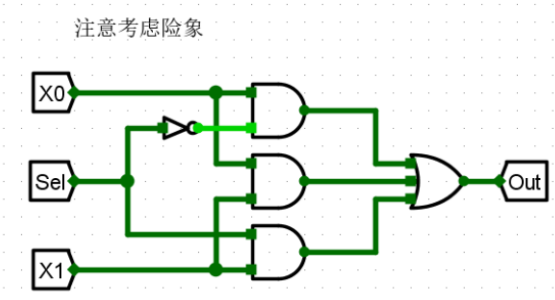


图 2. 2-7 2 路选择器（1 位）电路图

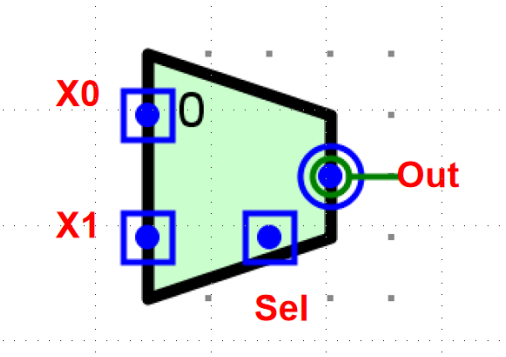
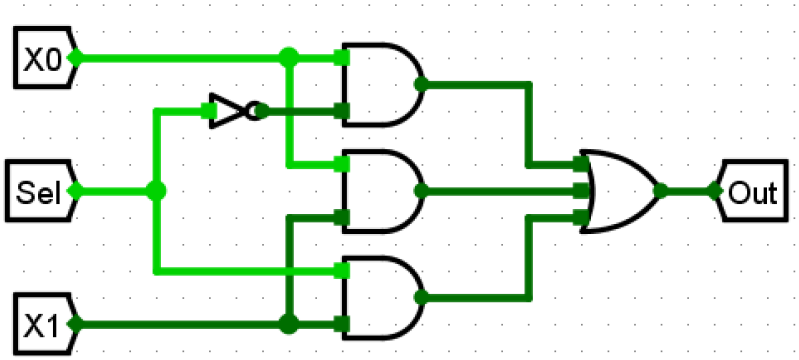


图 2. 2-8 2 路选择器（1 位）封装图

（b）测试图

由于 2 路选择器结构和逻辑上十分简单，测试只给出其中一种情况作为代表。



此时，$X0 = 1$，$Sel = 1$，$X1 = 0$，输出$Out = 0$

（c）测试分析

2 路选择器功能满足$Out =$ （$Sel == 0$）?$X0$:$X1$。

（2）2 路选择器（16 位）

（a）电路图

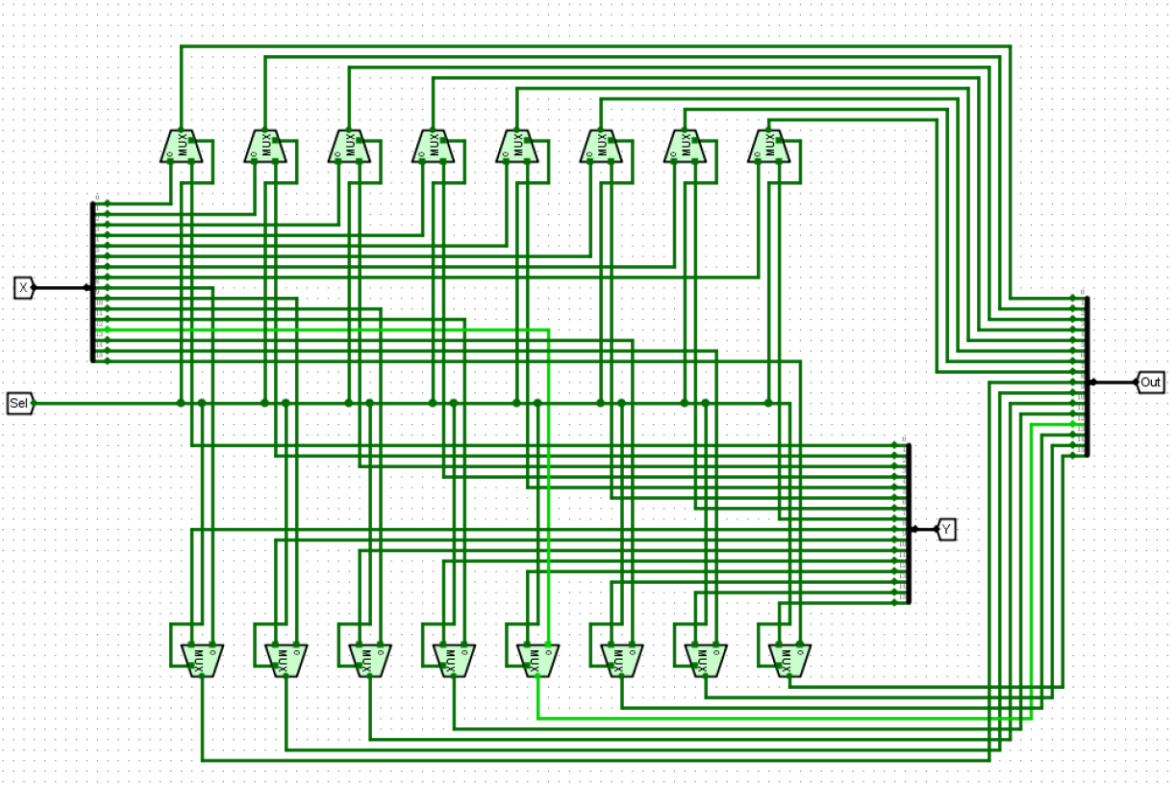
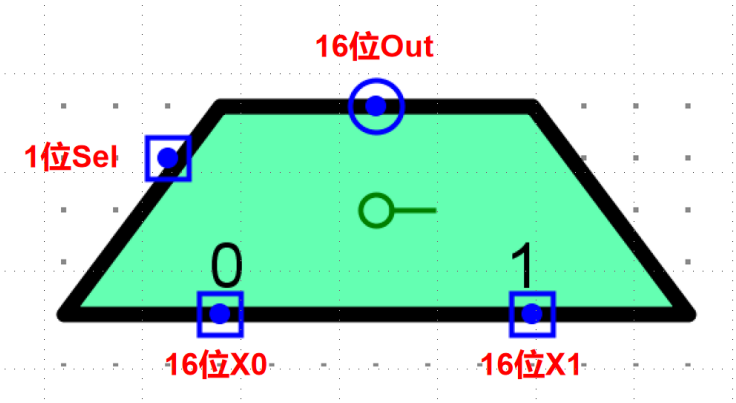


图 2. 2-10 2 路选择器（16 位）电路图



与 1 位的 2 路选择器相比，16 位的电路图是对输入输出的每一位进行 1 位的 2 路选择，最后汇总到一起。两个器件的封装是基本一样的。

图 2.2-11 2 路选择器（16 位电路图）封装图

（b）测试图

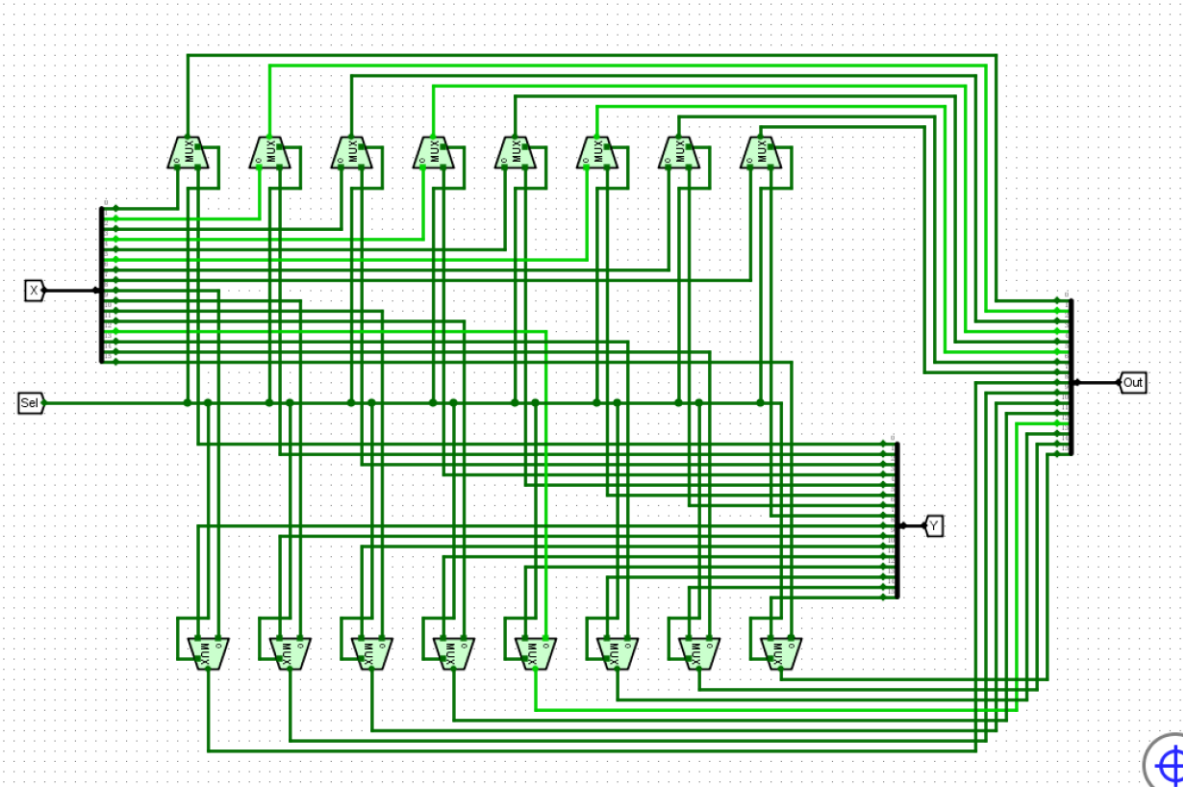


图 2.2-12 2 路选择器（16 位电路图）测试图

$Sel = 0$，$X1$中 1，3，5，13 位为 1，对应Out1，3，5，13 位为 1.

（c）测试分析

实际上，16 位 2 路选择器是 16 个 1 位 2 路选择器的并行，只要 1 位 2 路选择器没有问题，16 位的也没问题。

（3）4 位无符号比较器

（a）电路图

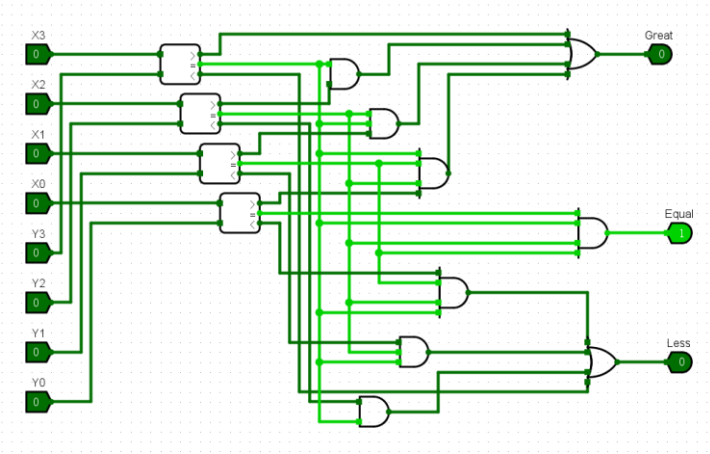


图 2. 2-13 4 位无符号比较器电路图

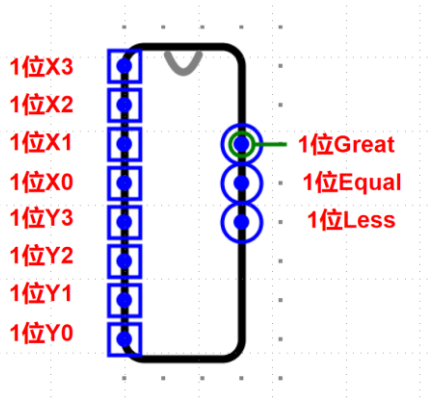


图 2. 2-14 4 位无符号比较器封装图

（b）测试分析

4 位无符号比较器是参照公式连接的，故检查公式的正确性即可检查电路的正确性。故不设置测试图。

公式如下：

$Great$	$X3 > Y3 + (X3 = Y3)\&(X2 > Y2) + ((X3 = Y3)\&(X2 = Y2)\&(X1 > Y1)) + ((X3 = Y3)\&(X2 = Y2)\&(X1 = Y1))\&(X0 > Y0))$
$Equal$	$X3 = Y3\&X2 = Y3\&X1 = Y1\&X0 = Y0$
$Less$	$X3 < Y3 + (X3 = Y3)\&(X2 < Y2) + ((X3 = Y3)\&(X2 = Y2)\&(X1 < Y1)) + ((X3 = Y3)\&(X2 = Y2)\&(X1 = Y1))\&(X0 < Y0))$

（4）16 位无符号比较器

（a）电路图

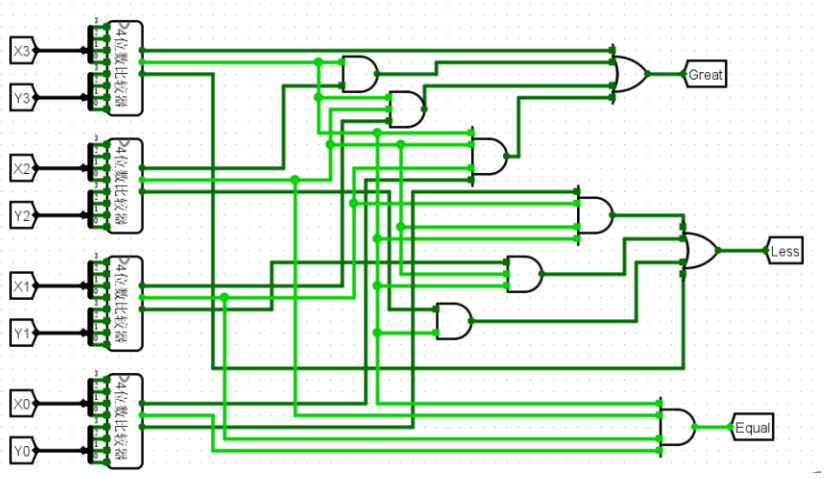


图 2. 2-15 16 位无符号比较器电路图

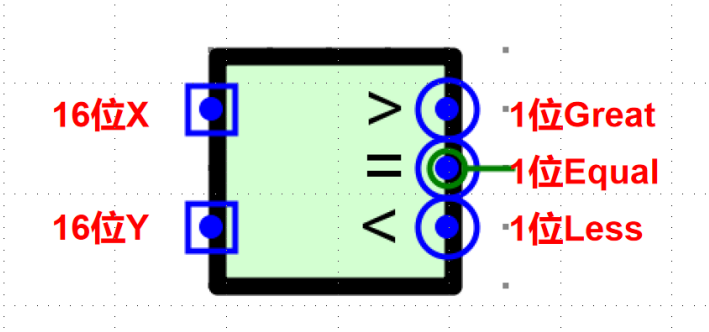


图 2.2-16 16 位无符号比较器封装图

（b）测试分析

16 位无符号比较器与 4 位无符号比较器的逻辑构成完全一致，只需将 16 的输入拆分为 4 个 4 位二进制输入，再对 XY 每个对应 4 位输入进行 4 位无符号比较器的比较即可，故不设置测试图。

2.2.3存储时间

（1）4 位并行加载寄存器

（a）电路图

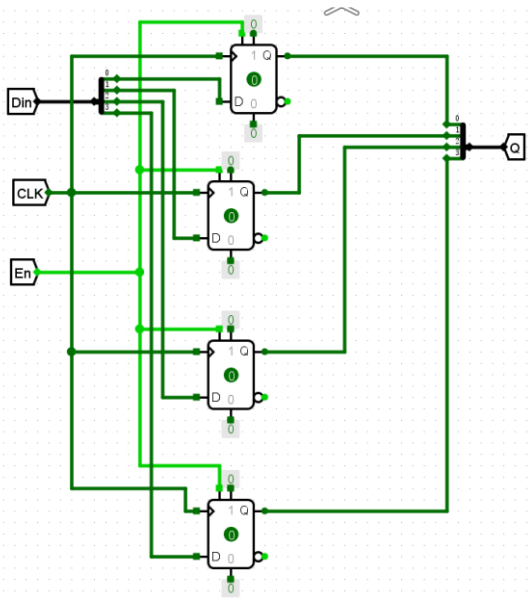


图 2.2-17 4 位并行加载寄存器电路图

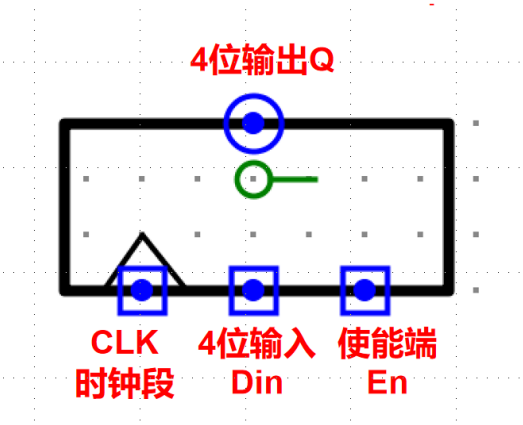


图 2.2-18 4 位并行加载寄存器封装图

（b）测试图

使能端 En 始终为 1，初始 CLK 为 0，将 Din 设置为“0100”，点击 CLK 模拟时钟脉冲信号。变化如下图。

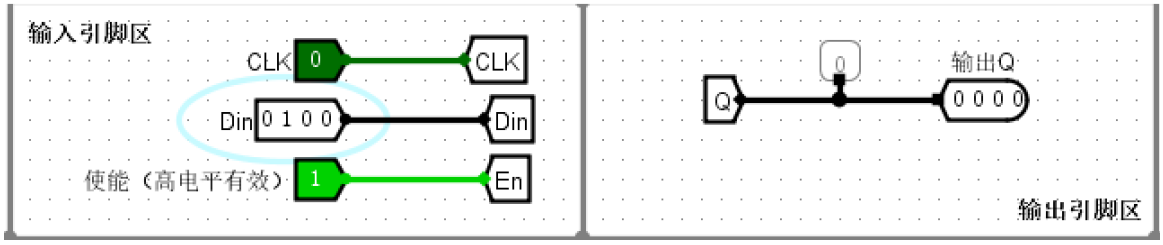


图 2. 2-19 初始状态

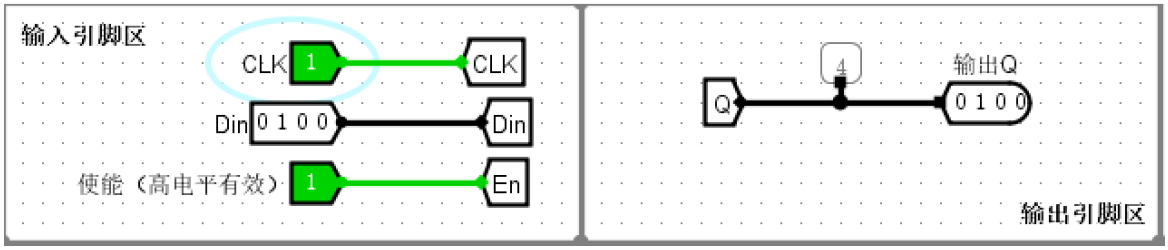


图 2. 2-20 点击后

（c）测试分析

完成了存储功能。

（2）16 位并行加载寄存器

（a）电路图

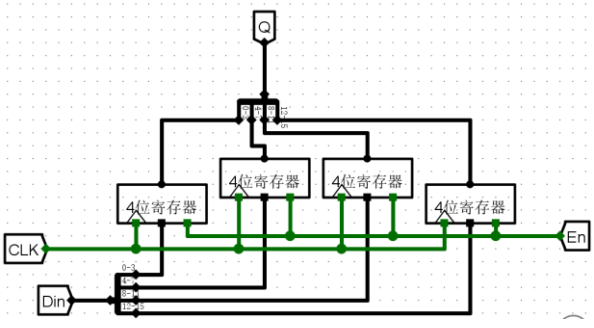


图 2. 2-21 16 位并行加载寄存器电路图

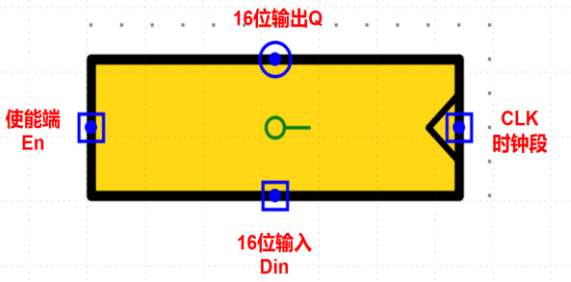


图 2. 2-22 16 位并行加载寄存器封装图

（b）测试图

测试信息从图中可知晓。

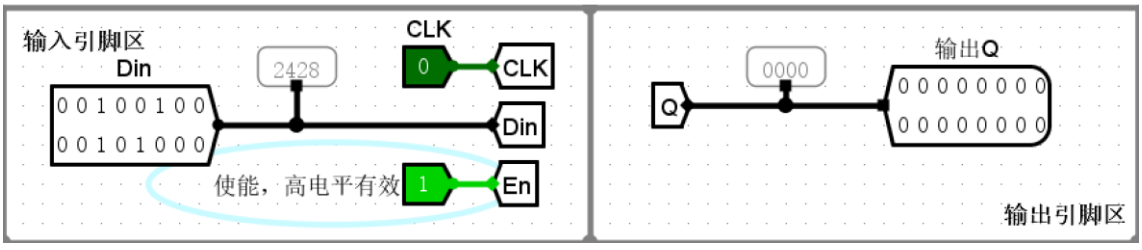


图 2. 2-23 初始状态

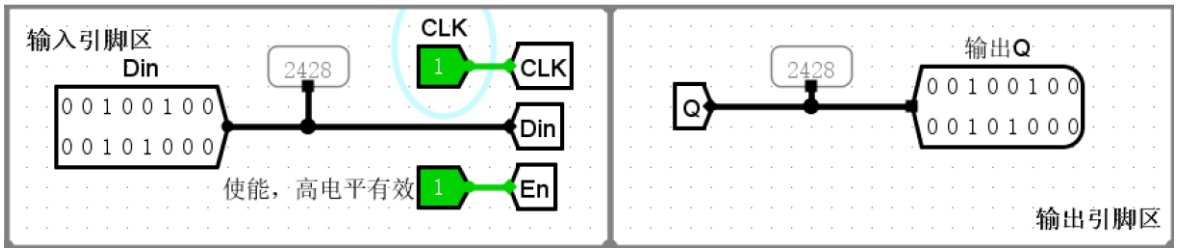


图 2.2-24 点击后

（c）测试分析

存储成功，功能实现。

2.2.4 “0-9”数字循环

4 位 BCD 计数器。

（1）电路图

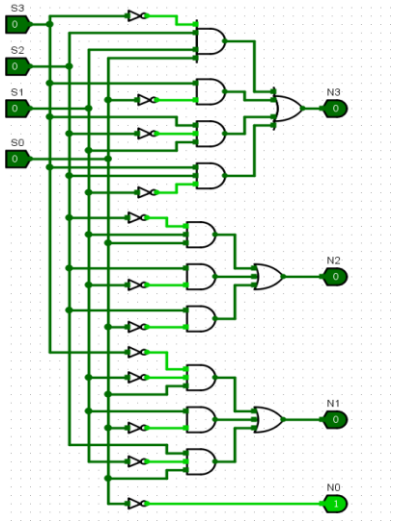


图 2.2-25 BCD 状态转换电路图

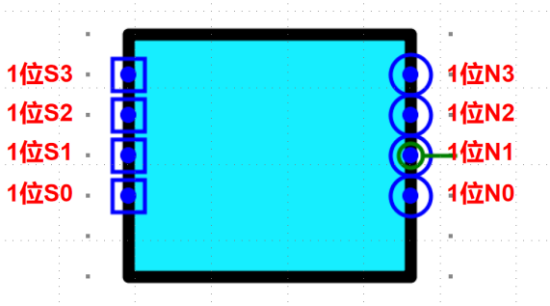


图 2.2-26 BCD 状态转换封装图

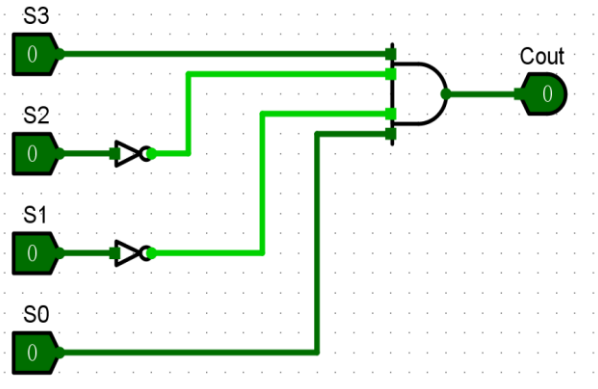


图 2.2-27 BCD 输出函数电路图

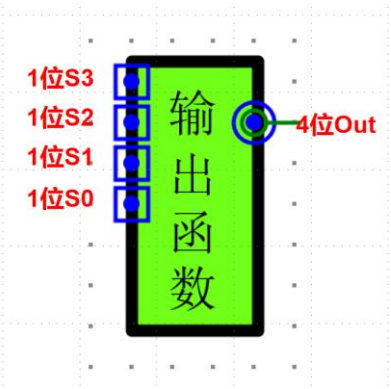


图 2.2-28 BCD 输出函数封装图

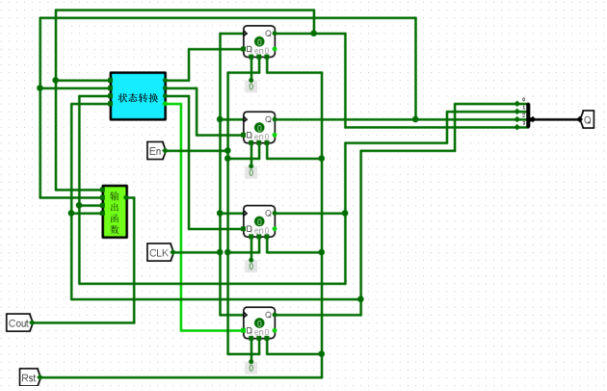


图 2.2-29 BCD 电路图

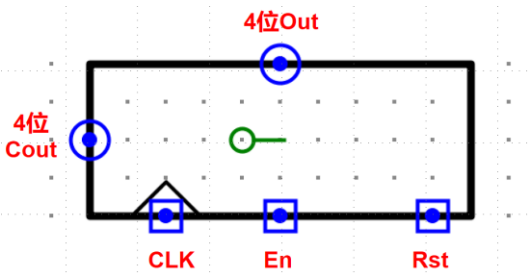


图 2.2-30 BCD 封装图

（2）测试图

4 位 BCD 的测试主要是看数字是否实现了“0-9”的数字循环。

直接观察运动码表的运作情况（这一部分需要涉及到最后运动码表的组合电路测试）。

测试图见下图。

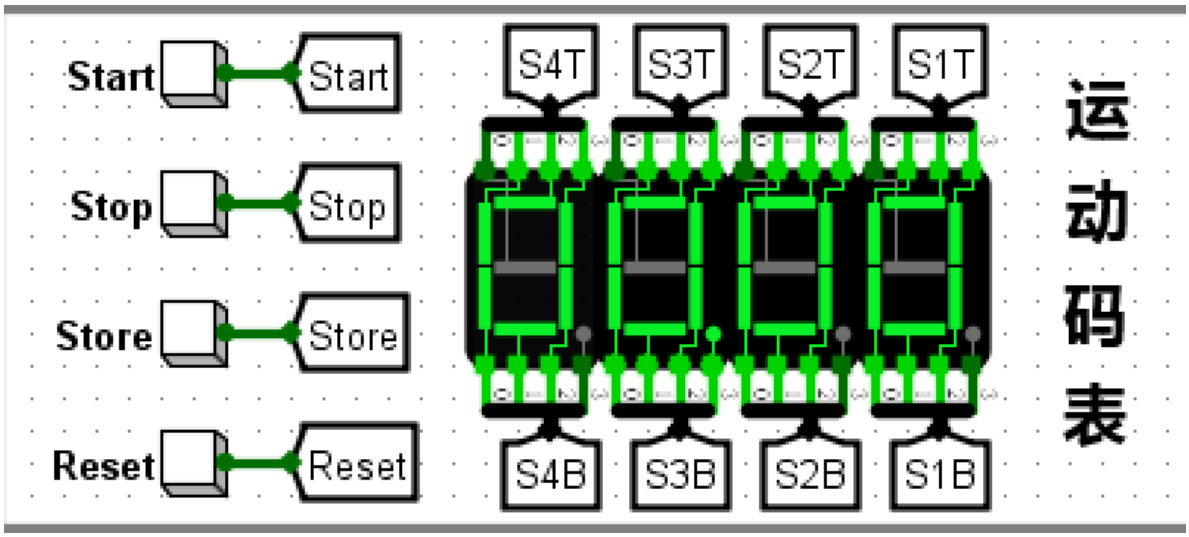


图 2.2-31 码表显示 运动码表观察测试

在 logisim 平台上，按下 start 进行驱动测试，随着时钟脉冲信号的输入，观察码表显示上数字显示变化。

（3）测试分析

每一位均可循环输出“0-9”，功能实现。

2.2.5计数进位

码表计数器。

（1）电路图

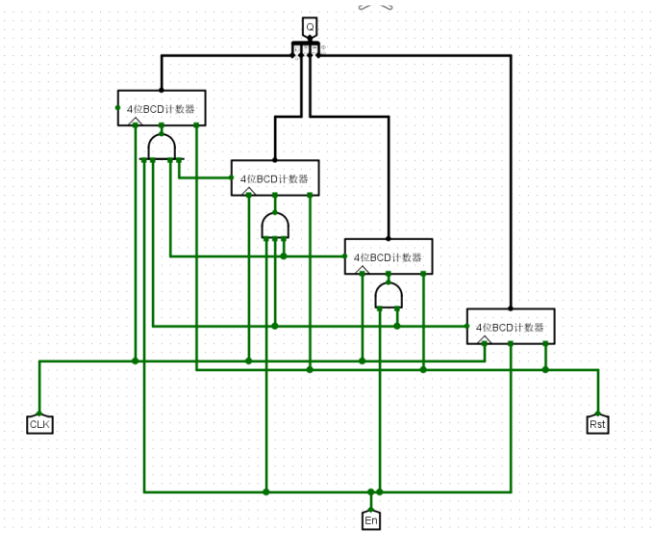


图 2.2-32 码表计数器电路图

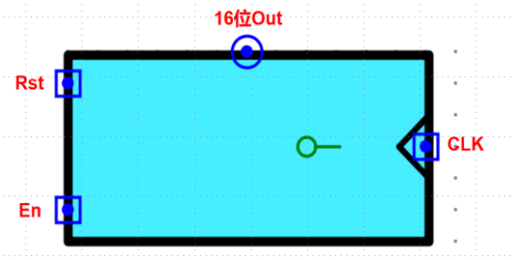


图 2.2-33 码表计数器封装图

（2）测试图

在 logisim 的码表计数器自动测试中，可以直接进行测试。

这里给出一个验证所有情况的状态——从“09.99”到“10.00”。

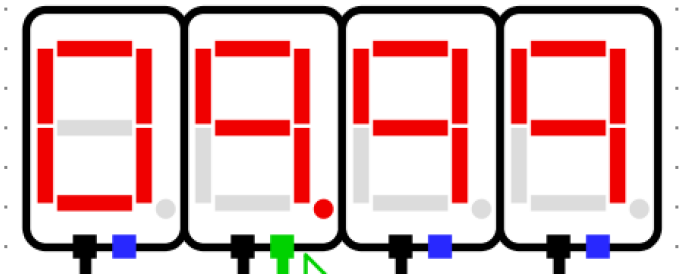


图 2.2-34 “09.99”

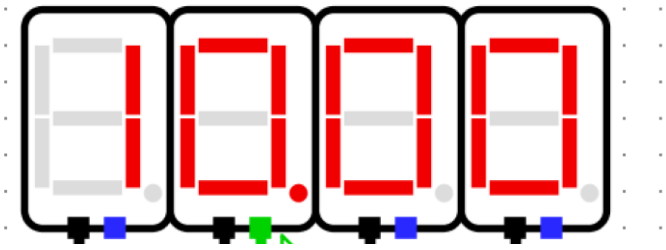


图 2.2-35 “10.00”

（3）测试分析

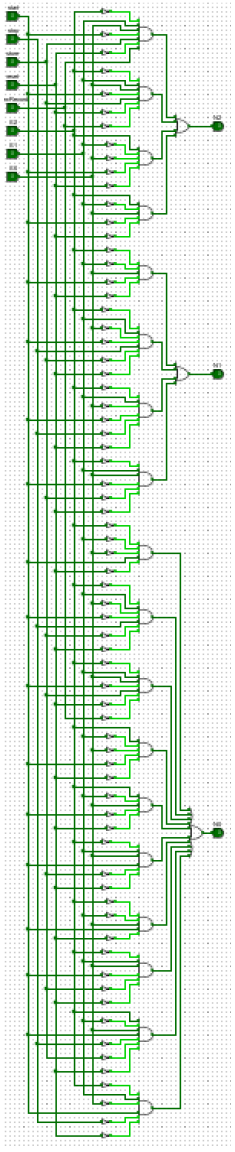
测试中，各位均可正常进位。

2.2.6按钮功能实现

码表控制器

（1）电路图

码表控制器两个函数电路图如下



（此电路图较大，但实际作用不大）

图 2.2-36 码表控制器状态转换电路图

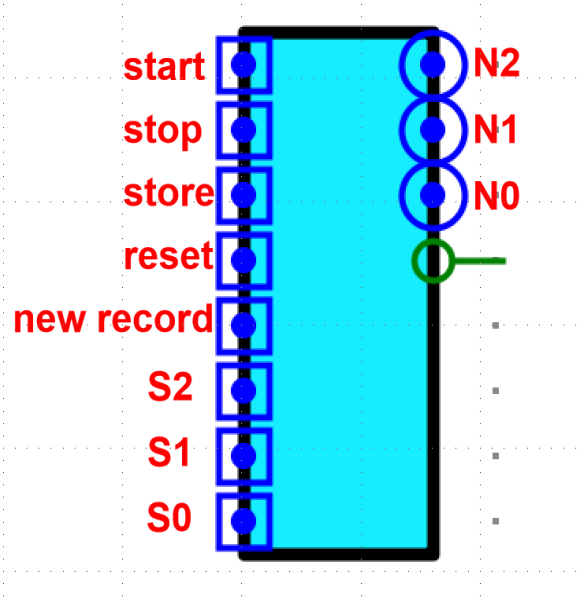


图 2.2-37 码表控制器状态转换封装图

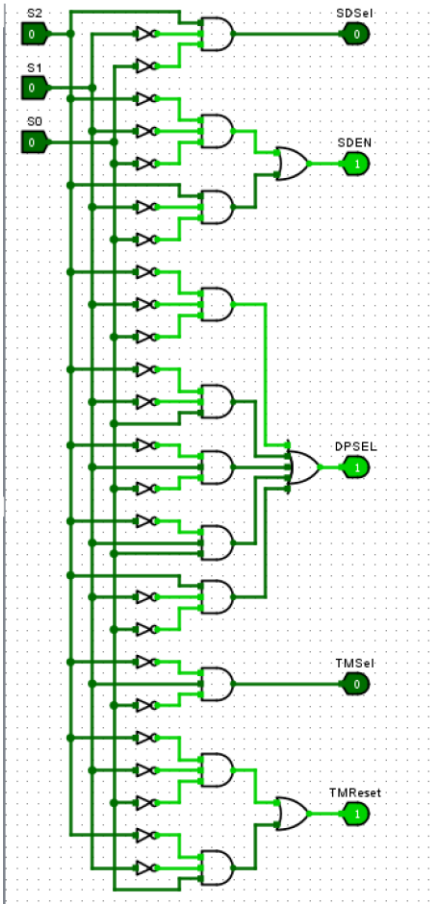


图 2. 2-38 码表控制器输出函数电路图

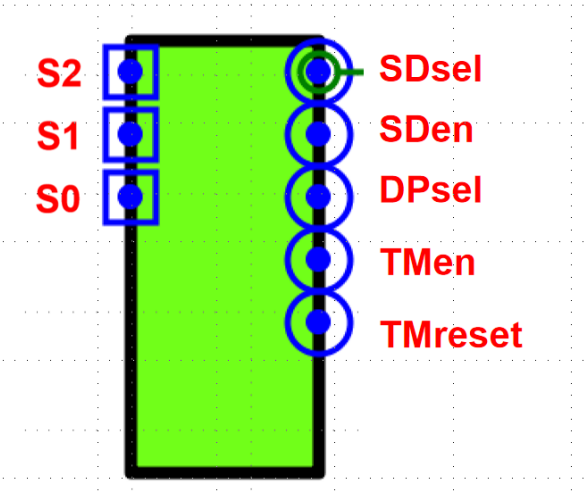


图 2. 2-39 码表控制器输出函数封装图

码表控制器电路如下

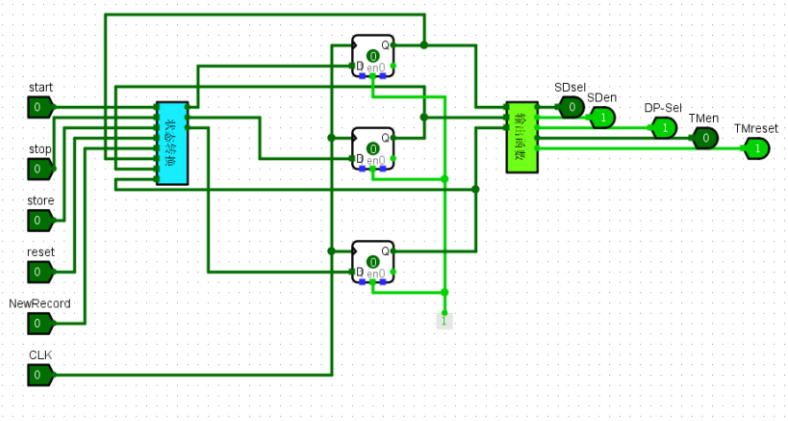


图 2. 2-40 码表控制器电路图

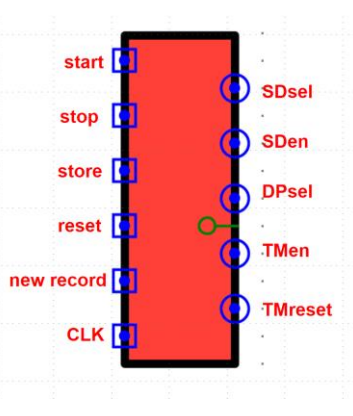


图 2. 2-41 码表控制器封装图

（2）测试图

在 logisim 中的运动码表中进行测试。

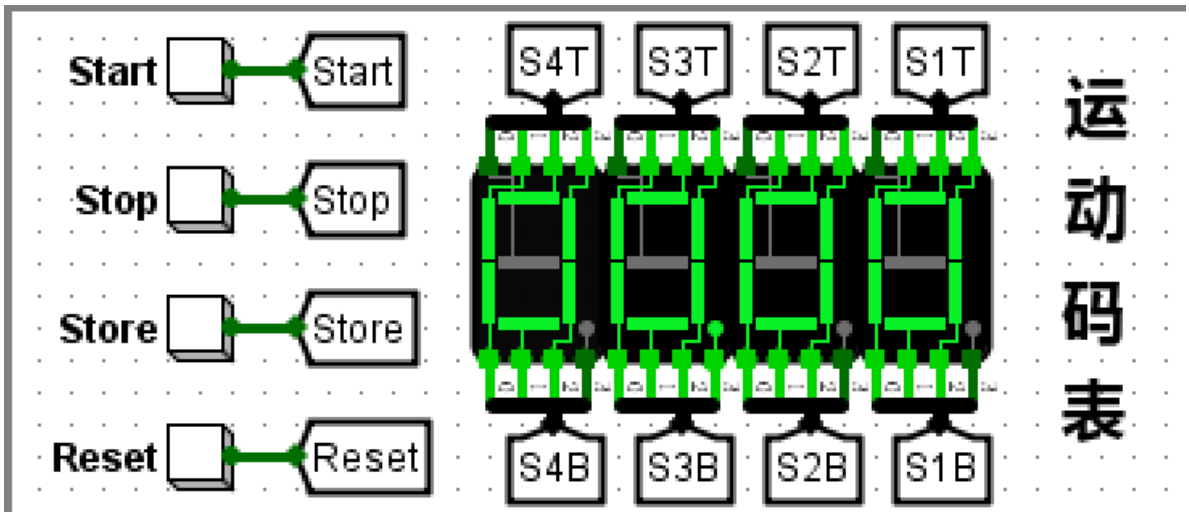


图 2. 2-42 按钮功能测试图

可以按照下面给出的方案，再结合测试者自身的情况进行测试。（记得打开时钟驱动）

测试 reset 功能	在任何一个状态下按下 reset
测试 start 功能	在按下 start 后，再次按下 start
测试 stop 功能	按下 start 后，再按下 stop
测试 store 功能	按下 start，按下 stop，再按下 store，储存一次时间。 再按下 start，一段时间后，按下 stop，再按下 store。
测试按键的鲁棒性	处于某个状态时，按下不会导致状态转移的按键

（3）测试分析

由设计的状态转移可以确定，在任何状态按下 reset，计时会清零，存储器复位为 99.99.

在任何状态下按下 start，计时会清零且立即开始计时。

在 start 之后，按下 stop，计时停止，会显示当前时间。

在 start 和 stop 之后，第一次按下 store，存储器记下当前时间 T0，计时器显示当前时间。再次按下 start，在一段时间 T1 后按下 stop，在按下 store，

若 T0>T1，存储器存储更新为 T1，计时器显示 T1；

若 T0≤T1，存储器存储不变，计时器显示 T0。

2.2.7运动码表汇总实验

运动码表。

（1）电路图

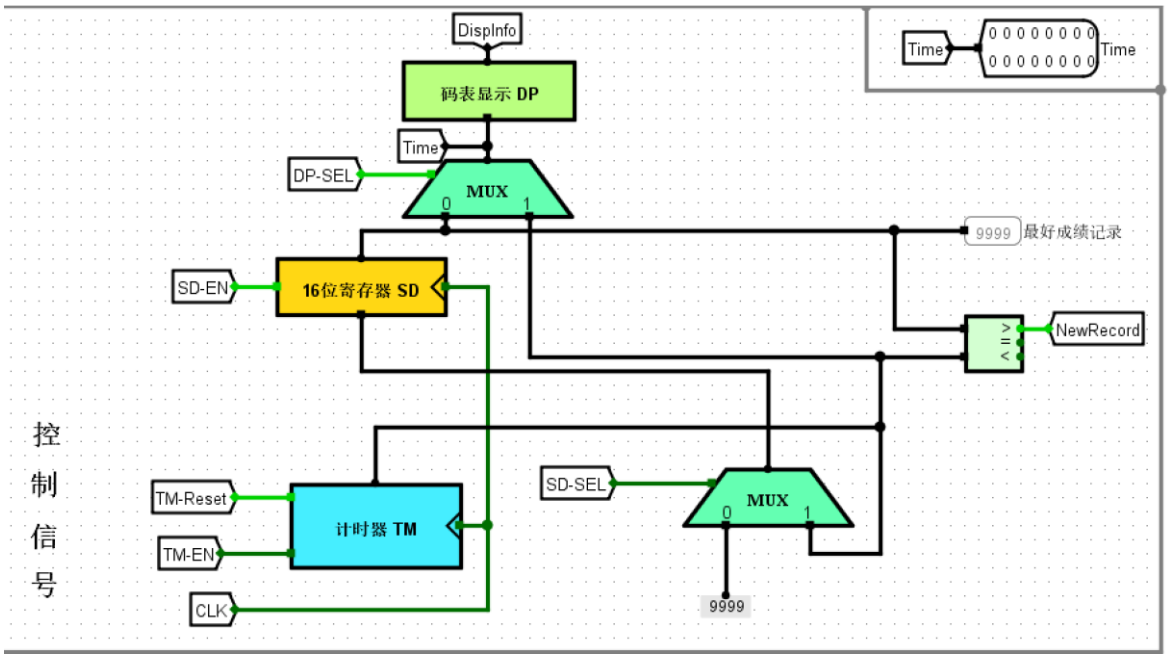


图 2.2-43 运动码表电路图

由于在本实验中运动码表未作为器件参与其他电路的构造，故没有给出封装图。

（2）测试分析。

实际上，在“按钮功能测试”前面的一系列步骤中，早已对运动码表最后的汇总做了测试，这里不再赘述。详细可参见 logisim 上的运动码表进行测试。

3　设计总结与心得

3.1 实验总结

本次实验，对理论学习中的各类触发器（尤其是 D 触发器）的功能有了更进一步的了解，再结合表达式、真值表、流程图的使用，以及合理设计电路逻辑，成功的完成了 BCD 计数器、并行加载寄存器等较为复杂的器件。从最后结果来看，本次实验比较成功；而从实验过程来看，也收获颇多。

3.1.1 遇到的问题及处理

实验中最先遇到的问题是测试平台（头歌平台）与实验要求不一致：

比如实验中的“6”和“9”在数码管中的显示相当于是“8”去掉一“竖”；但是测试平台上是去掉一“横”一“竖”。

比如码表控制器的状态转换，测试平台和实验要求的状态功能也不全相同。

在开始测试出错后，随后我意识到了这些问题并进行了更正。

实验中考虑最多的问题是码表控制器的状态转换函数。这里说明一下，测试平台与实验上的 start 功能不同的是，实验在按下 start 后再次按下 start，计时器会清零且立刻重新开始计时，但是测试平台是按下 start 后再次按下 statrt，计时器清零，不会立刻重新开始计时，需要再按一次 start。

这里问题的根源在于测试平台上，start 和 reset 共用了一个清零的状态。但实际上，这两个清零状态并不相同，应当区分开来（主要原因是清零后的步骤不同）。实验中，我将两个清零状态分开后成功实现了功能。

最后是一个小问题。在 logisim 软件中，我设计了一个 1 位无符号比较器（实验时以为不可以使用 logisim 上的器件）。后面在测试 logisim 上的 1 位无符号比较器时，发现会得出与正确结果相反的结论。询问老师后，才意识到这是平台工具使用不熟练的问题——Logisim 默认一位比较器是有符号的。这期间我反复测试，与同学交流，前前后后耽误了半个小时。

这也告诉我们，工欲善其事必先利其器。

3.1.2 设计方案存在的不足

与实验要求的功能相比较，本次实验很好的满足了所有功能。但在以下几个方面有待改进：

部分实验电路的连线有待改进。如“2 路选择器（16 位）”的电路连线，其上下的整体分布不太符合平时逻辑，略有不足。还有其他几处线路有待改进。

与实际情况相比，本次实验设计的运动码表，没有处理同时按下多个按钮的情况，一旦出现，其是否产生竞争或者报错有待考察。在实际操作中，显然是有可能出现这种现象的。

3.2 实验心得

广泛查阅资料，强化自身学习能力，多交流多问，自己也多尝试，对知识的掌握与运用有很大帮助。

3.3 意见与建议

对于本次实验，整体感觉挺好的，适合初学者的上手实验，后续进一步学习可以加大难度，并且实现更复杂的电路与功能。

原创性声明

本人郑重声明本报告内容，是由作者本人独立完成的。有关观点、方法、数据和文献等的引用已在文中指出。除文中已注明引用的内容外，本报告不包含任何其他个人或集体已经公开发表的作品成果，不存在剽窃、抄袭行为。

已阅读并同意以下内容。

判定为不合格的一些情形：

（1） 请人代做或冒名顶替者；

（2） 替人做且不听劝告者；

（3） 实验报告内容抄袭或雷同者；

（4） 实验报告内容与实际实验内容不一致者；

（5） 实验电路抄袭者。

作者签名：