东南大学数字系统设计

课 程 设 计 报 告

**选择题目：** **第四题**

**成员一：** 61522533 查李垣

**成员二：** 61822205祈与同 —

**评定成绩：**  **审阅教师：**

**2024年 8 月 31 日**

**一、选择题目及功能要求**

**1.1 题目描述**

此项目用于模仿地铁售票的自动售票，完成地铁售票的核心控制功能。

① 地铁售票机有两个进币孔，可以输入硬币和纸币，售货机有两个进币孔，一个是输入硬币，一个是输入纸币，硬币的识别范围是1 元的硬币，纸币的识别范围是5 元，10 元，20元。乘客可以连续多次投入钱币。

② 以南京市轨道交通1/2/3/4号线为基准进行设计考虑。站点数较多，需自行编码。

③ 系统可以通过按键设定当前站点为4条线路中任意一站。

④ 乘客买票时可以有两种选择，第一种，乘客已经知道所需费用，直接选择票价，如2元、

3元或4元或更多。第二种，不知道票价，选择出站口，系统以目的地与当前站的站数来

进行计算价格，计算方式参考南京市轨道交通计价标准。请注意，由于换乘站的存在导

致两地之间有可能有多种价格的，以最低价格为准。

⑤ 得到票价单价后，选择所需购买的票数，然后进行投币，投入的钱币达到所需金额时，

售票机自动出票，并一次性找出余额，本次交易结束，等待下一次的交易。在投币期间

乘客可以按取消键取消本次操作，钱币自动一次性退出。

**1.2 完成功能概述以及项目优势**

本项目包含4个状态，依次是“用户确定单价状态”，“用户选择票数状态”，“用户进行付款并找零/退回 状态”“重置参数状态”，用户随时可以返回第一个状态进行重新确定单价，并且在付款时若是金额没有达到应付额则可以进行退款。付款完成并且出票后系统会自动回到“用户确定单价”状态，等待下一轮的购票。

本项目的优势在于：

1. **显示直观，操作简易**：

在用户确定单价时，如果处于“通过起始站和终点站确定价格模式”，用户可通过**左右按键确定要操作的位数，通过上下键确定该位数字的加减**，与此同时价格会**同步显示**在数码管的最右侧，方便用户查看。

如果处于“自行选择票价模式”，考虑到票价不会是两位数，故提供了2元~9元的范围供用户选择，确定单价后，用户按下“确定”键便完成了操作，

下图展示了数据的显示模式，数码管的第一位显示的是起始站的地铁线的序号（1，2，3，4），第二位显示的是始发站的高位数字（始发站序号为个位数时为0），第三位显示的是始发站的低位数字；数码管的第四位显示的是终点站的地铁线的序号（1，2，3，4），第二位显示的是终点站的高位数字（终点站序号为个位数时为0），第三位显示的是终点站的低位数字；例如，下图显示了从1号线的第11站到4号线的第02站的价格：

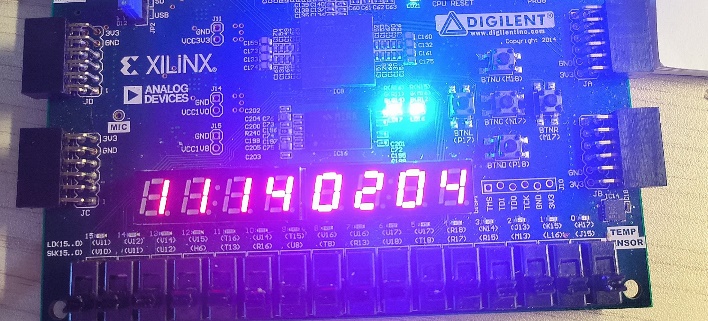


图 1.1 根据起点和终点确定价格

与实际的票价相对比，发现价格是正确的。

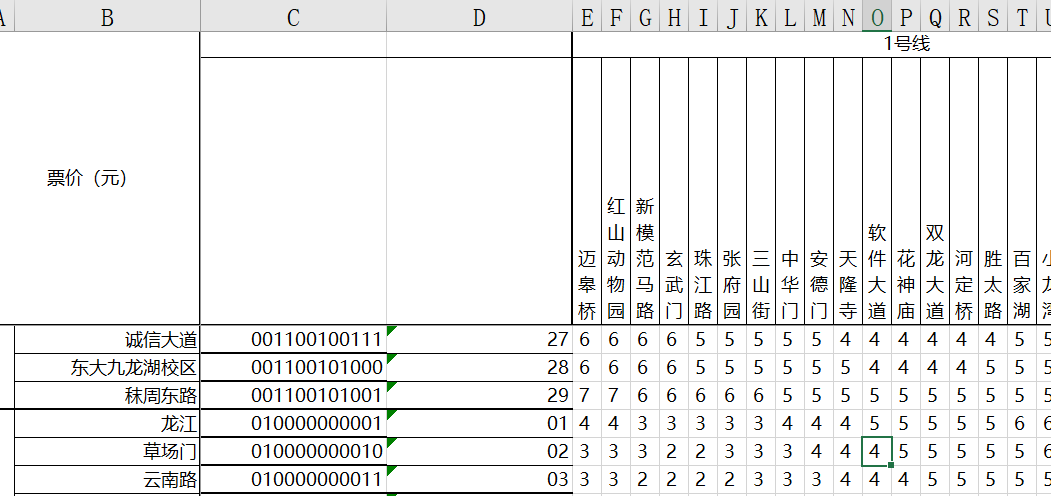
****

图 1.2 根据起点和终点确定价格，价格的验证

同时，在用户自行确定价格时，将不必要的显示去除，方便价格的指定：

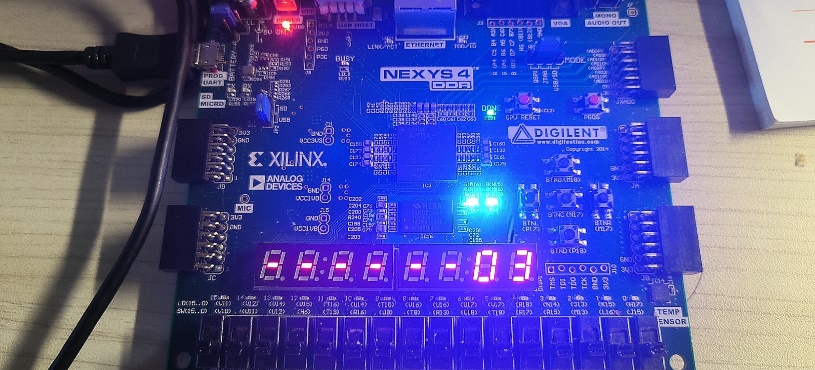


图 1.3 用户自行指定价格的界面

按“确定”键进入“选择票数状态时”，**总价格**会**实时显示在数码管的最右端**，与此同时用户选择的**票数**会实时**显示在数码管最左端**，如下图，是选择票数为9，单价为8元的显示界面，与此同时可以看到指示灯发生变化，说明状态的改变。

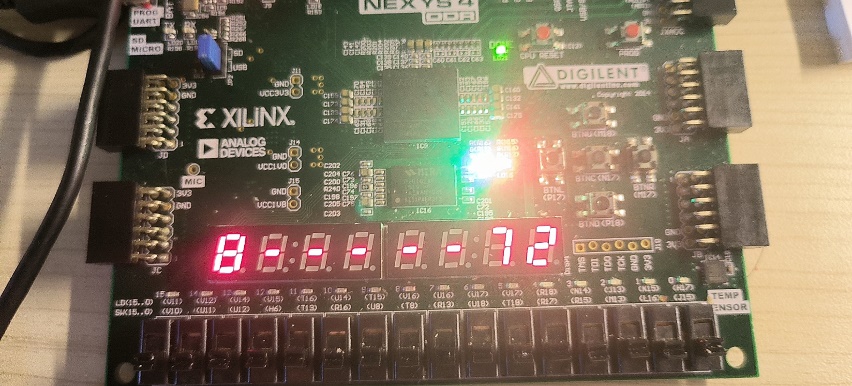


图 1.4 用户选定的总价格

选好总价格以后，进入付款状态，用户通过拨动4个拨码开关来实现“20，10，5，1”的投入，到达所需金额后将会出票，同时多余的钱数会被退还，若是中途用户取消操作，按下按钮中的“down”键即可实现退款重付；若是想要重新选择则直接拨动最左侧的拨码开关返回原始状态，如下图1.5是投入4张20元纸币后的结果，系统将返回8块。

返回零钱后用户再次按“确定”键，系统进入“参数清零”状态，0.5s后系统**自动返回**到“确定单价”状态，其他用户进行下一次购票。

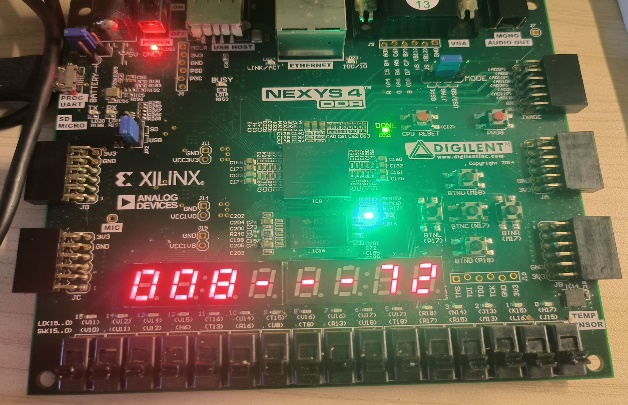


图 1.5 退款找零界面

1. **鲁棒性较强**

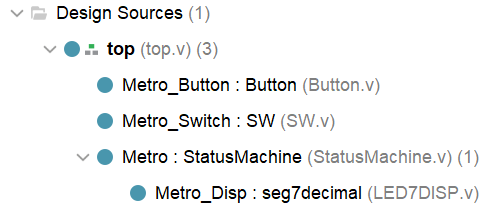
本系统考虑到了多样的输入样例，对于边缘特殊值有着较好的处理，比如输入为9张9元的票时，如果用户选择使用5张20元来付款也能正确处理而不会出现截断情况。

除此之外，**用户对于按键的“胡乱操作”也不会造成错误**，比如当前显示的是地铁4号线，用户再增加就会变成1号线而不是5号线；系统会根据用户输入的地铁线路来判断该线路的站台数而不会出现越界；用户从站数较多的线路更换线路时系统也会作出防越界判断，比如用户将地铁3号线的29站，只修改地铁线路到4号线，那么后两位就会自动变更为4号线的09站而不是4号线的29号站，确保了正确的输入范围。

**二、代码层次及功能的实现细节**

**2.1. 代码层次**

本项目以 为顶层设计， 为按钮消抖电路，为开关消抖电路，为状态机模块，里面调用了数码管显示模块

****

**图2.1 项目代码层次**

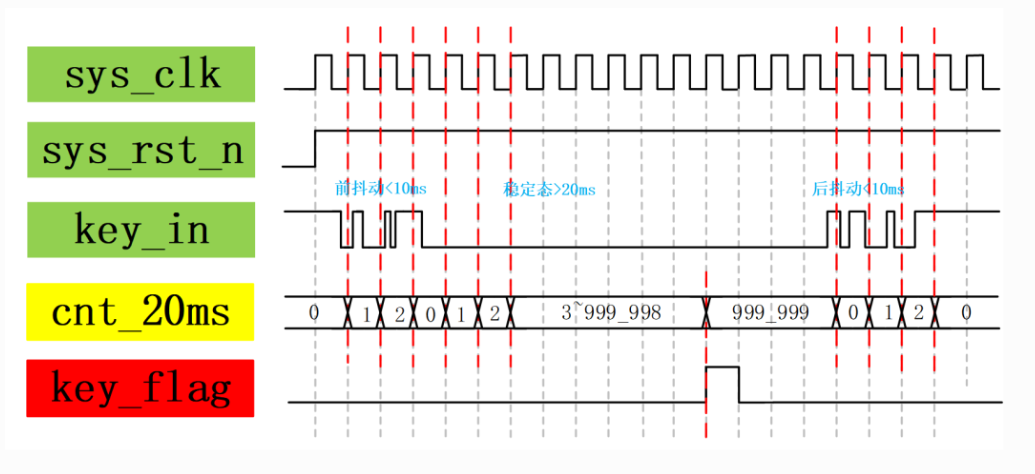
* 1. **实现细节**

**2.2.1 按键与开关消抖模块的思路，仿真及实现**

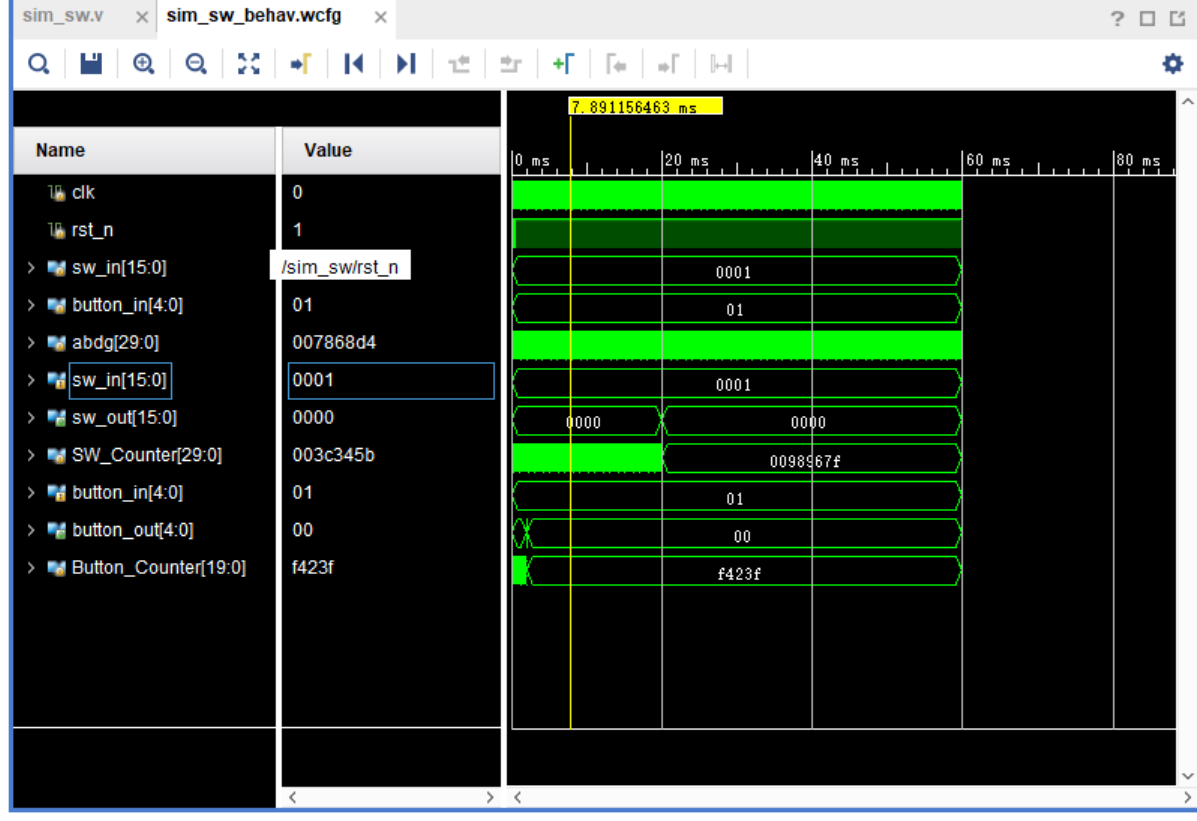
**信号输出：后续信号触发检测时，在敏感信号表中，我们选择电平触发方式，所以在按下按键或者拨动开关时，要求消抖后达到在信号的上升沿检测后输出一个时钟周期的高电平。**

**消抖：采用延时消抖思路，采用计数器计数进行延时，在信号第一次发生变化时，如由0到1或有1到0时开始计数，若计数结束后，变化依旧存在，则输出一个脉冲信号，并且保持计数器值，反之则将计数器重新置0，由于按键和开关的物理属性不同，我们选取不同时长的计数器。**

**理想的波形图：**

****

**仿真图像：**



**计数器保持**

按钮计数器

**产生脉冲信号**

按钮输入

开关输入

开关计数器

开关输出

开关输入

**复位信号（0复位）**

**2.2.2 7段数码管的显示模块仿真及实现**

**显示模块：采用四位编码的形式，并且与限制程序中端口相对应，从而实现编码值为1-9时，显示1-9数字而编码值为10-16时显示横杠。具体地来讲，就是动态驱动数码管方式：使用一定的刷新时间让8个数码管轮流显示，如果频率够高，这样就能让人眼感觉到数码管一直在亮了。点亮相应数码管的时候给其显示相应的值。**

**2.2.3 顶层模块的实现**

**顶层对按键，拨码开关，数码管进行了例化，并且规定了状态机的跳转条件；为了指明系统目前所处的状态，我们把四种状态机的状态进行编码，令不同状态对应不同的LED灯排列顺序。我们将商品的价格，购买的票数，起点站，终点站等信息根据状态的不同显示到数码管上，由于数码管的数量有限，例化数码管显示模块时，其中的显示变量在不同阶段被赋予了不同的涵义。**

**2.2.4 状态机的仿真波形及实现**

**状态机基本框架：**

**State：00：选票：确定初始站/终点站，以及购票数，并显示最终价格**

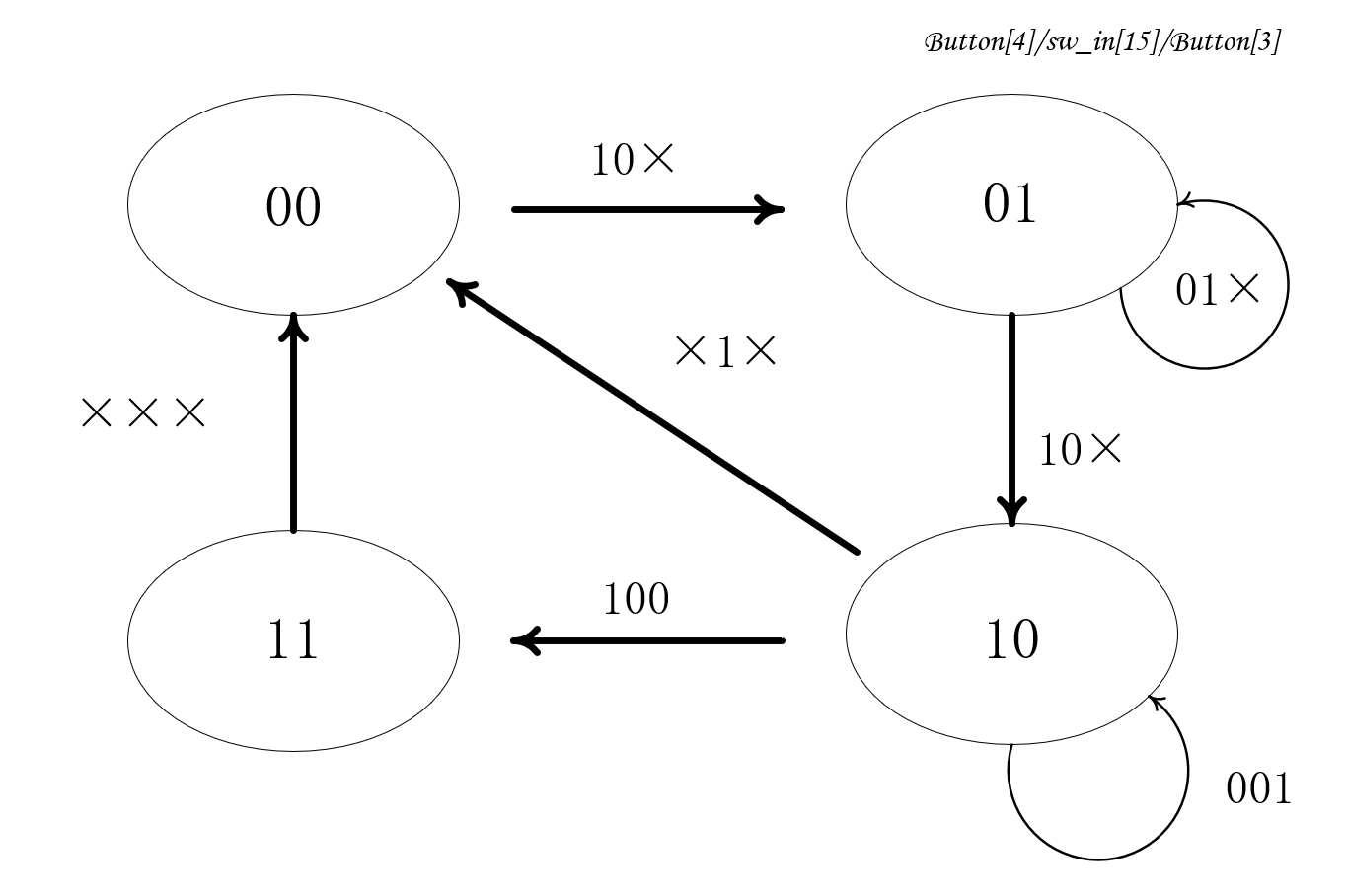
**选定价格：选择对应价格的车票**

**01：付款（付选票/选价格的钱）**

**10：退款（退款由大到小依次退款）**

**11：中间状态，自动跳转为00**

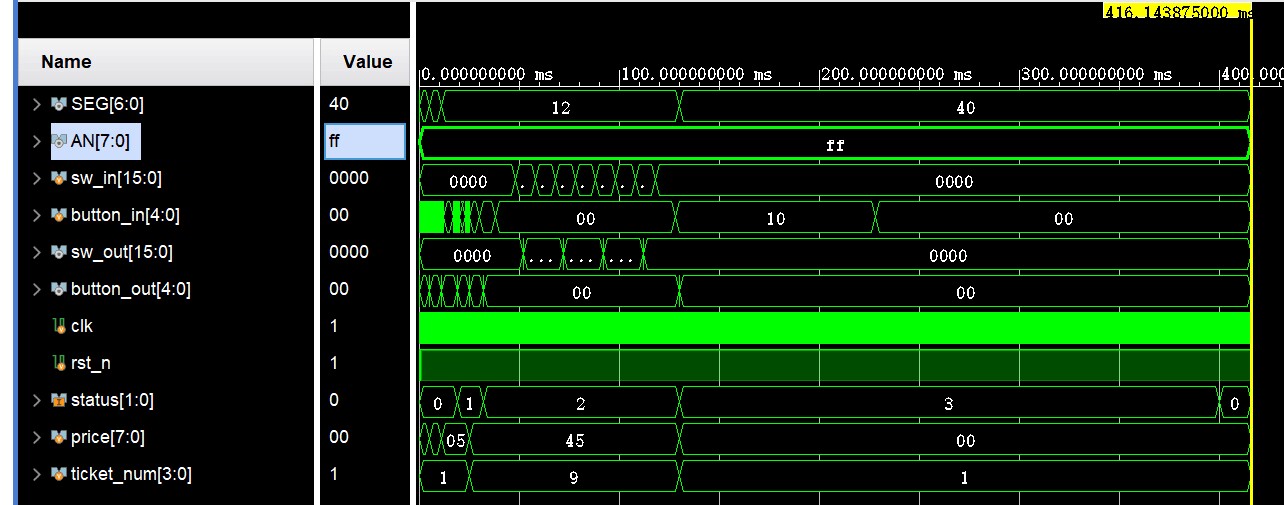
**转态转移图：**

****

**上图中，相当于“确定”键，主要用于进入下一个状态；相当于“清零”键，主要在付款时用于退款重新付款，而主要用于返回一开始的选定单价状态，相当于系统的原状态。**

**下面展示了先选定单价为5元，再选定票数为9张的波形，可以看到计算出的价格为45元，在顾客付款后（体现在sw\_in）并找零后，用户按下确定键后，一段时间后系统自动返回初始状态，这里的3阶段看起来持续时间长，是因为实际上板时为了体现3阶段的存在而故意设置长的，对于仿真而言是不存在问题的。**

**仿真波形：**



**状态**

**三、Bug的出现情况以及解决方法**

**1、状态机仿真时，信号显示没有问题，但状态机却始终无法跳转。**

**问题：仿真文件例化错误，又由于两个变量同名导致显示时误将仿真文件中的信号，当做状态机中的信号。**

**解决方法：修改例化。**

**2、敏感信号列表冲突问题。**

**这是初期时最令人困惑的问题了。状态机是由时钟的上升沿进行驱动的，而我们又想让进行消抖后的按钮输出电平的上升沿来驱动变量的赋值、状态的跳转等。**

**解决方法：在状态机内只能用语句，所以我们让按键的信号只存在一个时钟周期就可以达到相同的效果。**

**3、信号的驱动问题**

**在不同的块中不能对同一个信号进行赋值，所以我们尽量在同一个块中对信号赋值，而在其它块中利用变量的值进行判断等操作。**

**4、其余的逻辑bug**

**这里主要是数值的越界与否问题，与选定的线路站数等有关，除此之外，加减法、乘法的处理与显示也值得关注，这涉及到BCD码的转换等问题。**

**注：在提交的源码中，我们编写了大量的英文注释，记录了bug的产生与解决方法。包括一些需要注意的点均有提及。**

**四、项目分工及总结**

**项目分工：查李垣：工作量 60%。负责状态机的部分编写以及数码管的显示部分**

**祁与同： 工作量 40% 负责状态机的部分编写，以及开关，按钮部分编写。**

**总结：在编写过程中，经常遇到各式各样的问题。经常会出现仿真跑成功，但是上板却出现问题。又或者简单的功能直接上板了，但仿真却没有跑成功。同时，我们对这个软件以及代码语言缺乏熟练度，在一些方面使用的相当保守，不敢进行一些大胆的尝试。并且在编写过程中经常出现各式各样的问题。例如，在一开始编写top时，我甚至意识不到，要做到输出端接与接收端要与板上的限制程序的端口要一一对应。然后再生成bit流文件时，经常出现一个端口被复用两次的各式各样的问题。同时，由于代码书写欠规范，两人合作编写状态机时，经常会出现端口不对，导致需要大范围修改的现象。虽然在过程中遇到了各式各样的问题，但最终我们都将其克服。总而言之，完成的还是相对顺利。**

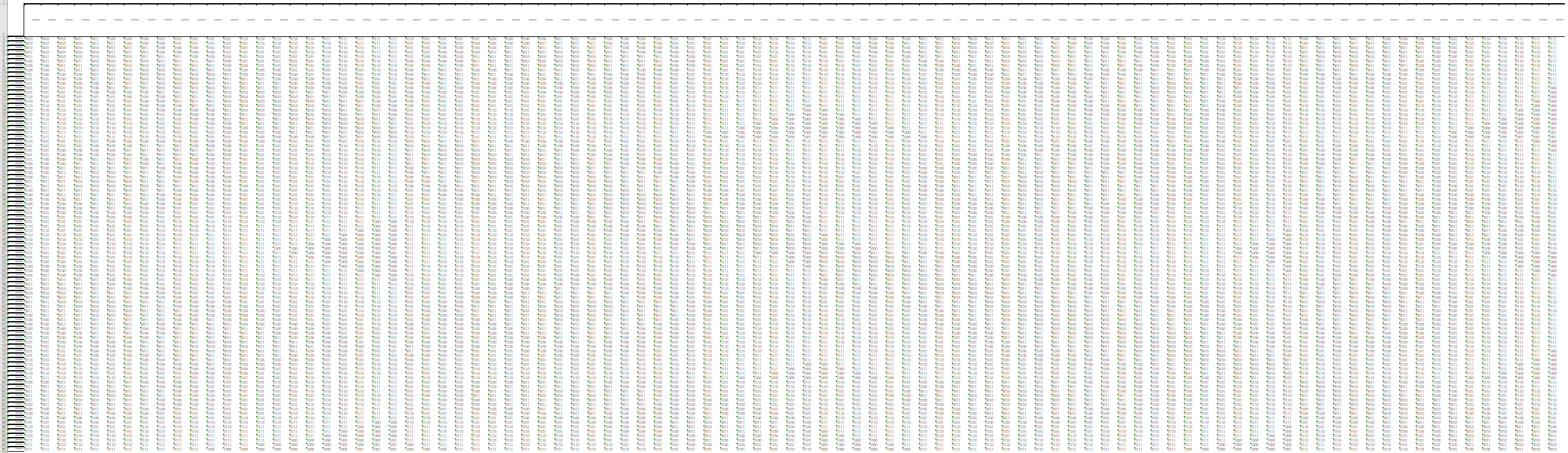
**附录**

为实现准确计算站与站之间乘坐地铁的最低票价，本项目选择在开发板外提前计算出各票价，存储在中，

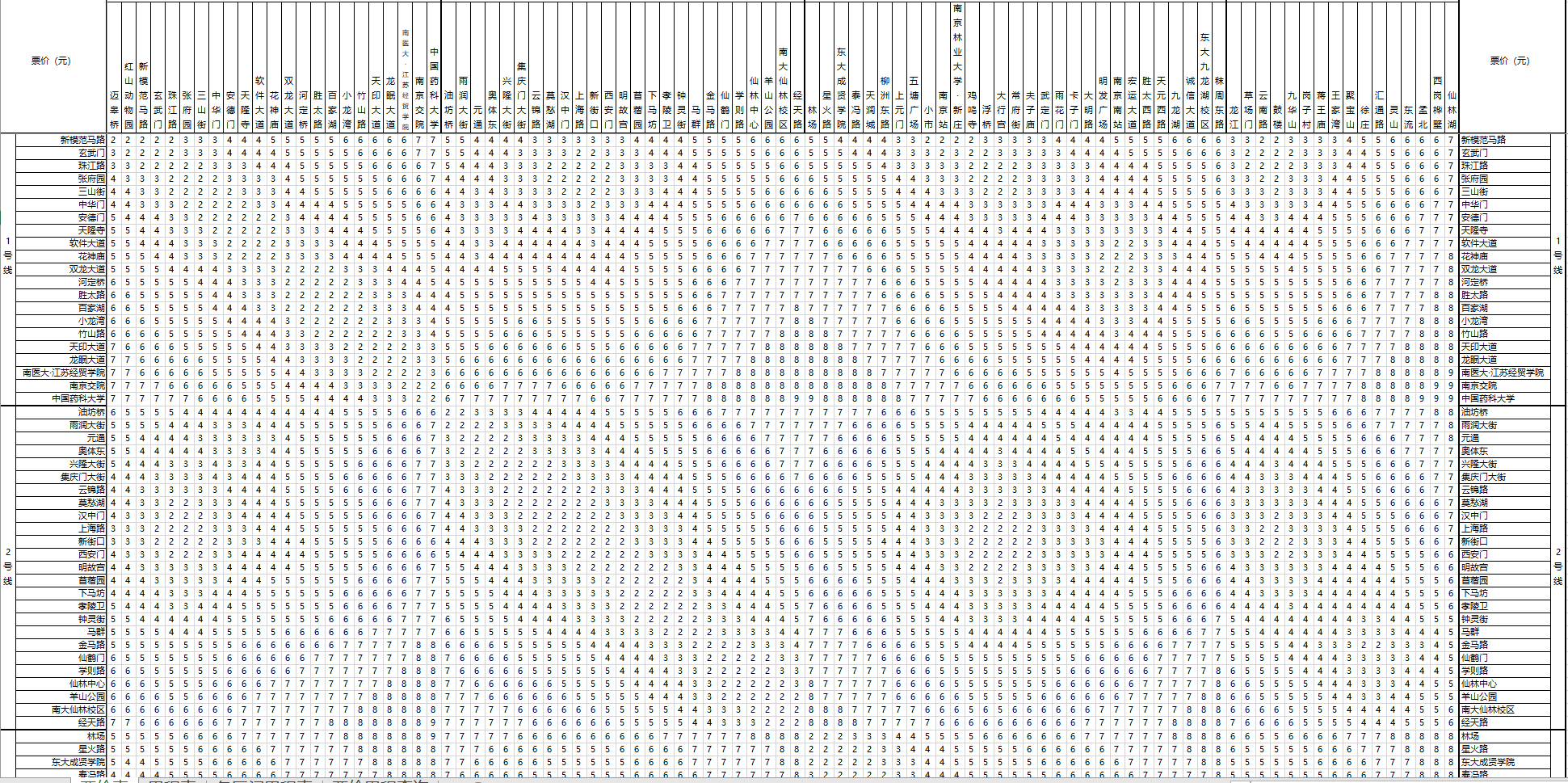
为了节省精力，我们使用了调用了包进行重复代码的撰写，具体见下图。



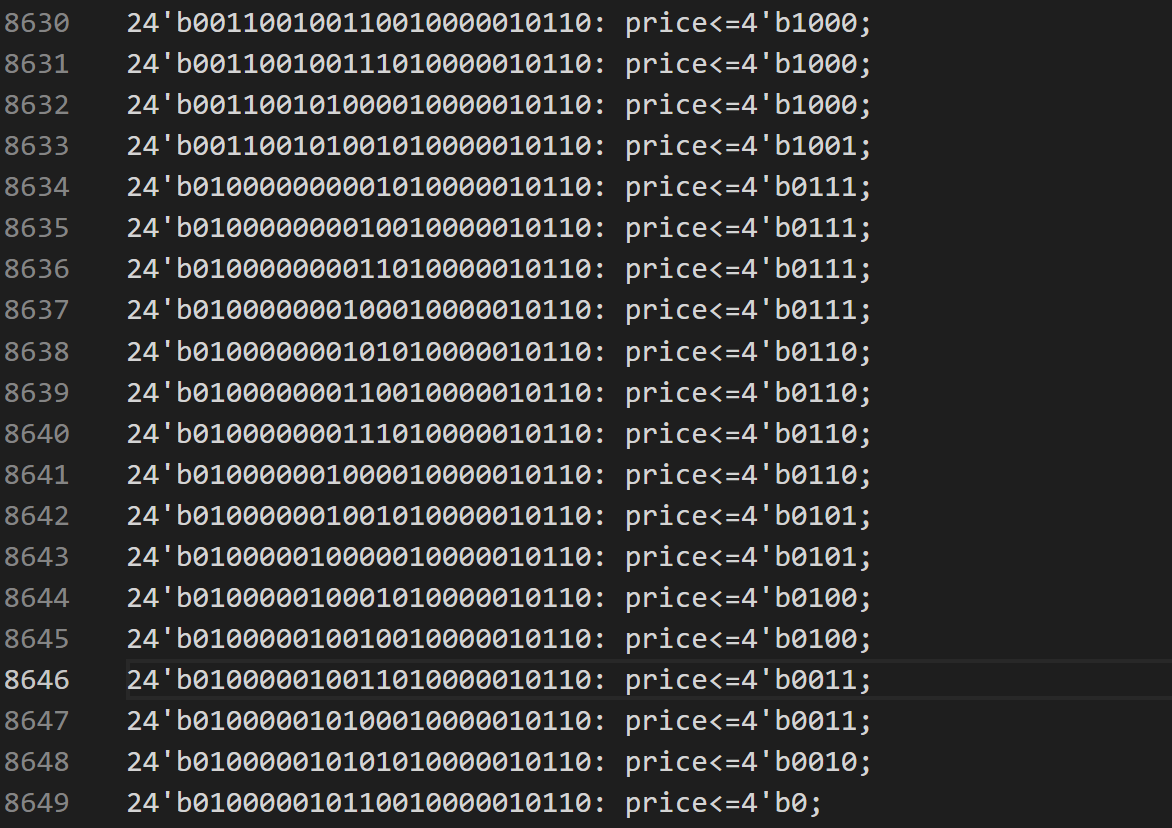
附1 脚本代码



附2 进行编码后的价格表



附3 编码前的价格表



附4 脚本输出的txt文件，共93×93行