# EDA 大作业二
# 投币式手机充电仪
# 实验报告

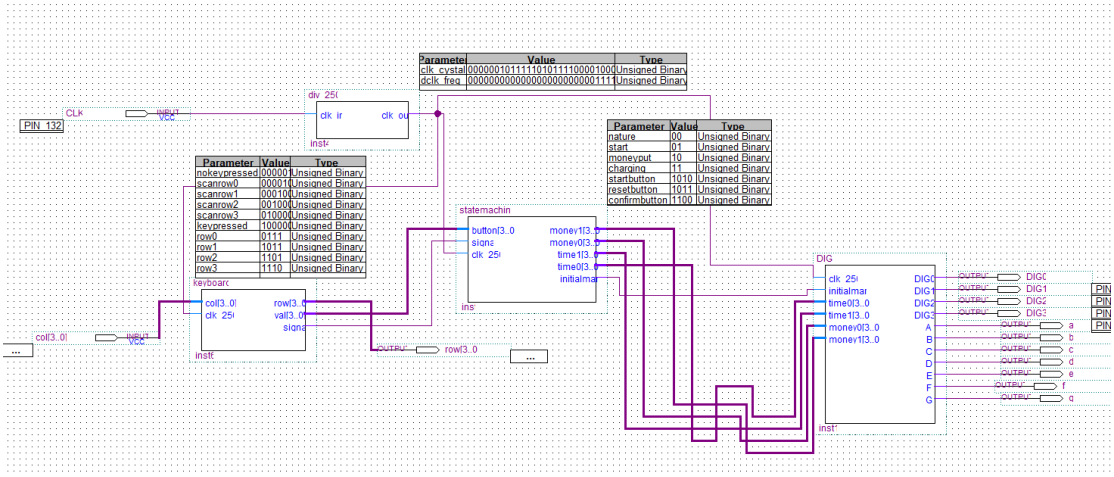
姓名：常成
班级：自 75 班
学号：2017010252

一、 实验目的

1. 学习自顶向下、分模块的数字系统分析、设计与调试方法。
2. 学习编写测试文件，编写 testbench 对设计电路进行仿真验证。
3. 掌握规范使用硬件描述语言描述状态机电路的方法。

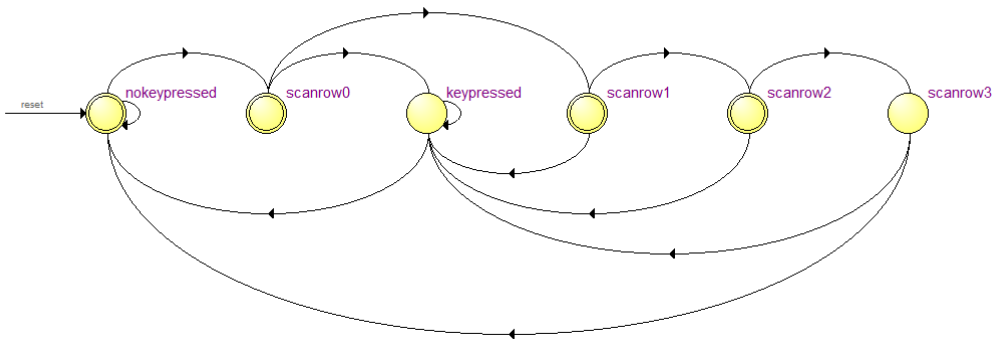
二、 预习任务

1. 阅读并分析任务要求，画出电路的总体框图，注明各功能模块及其引脚。
2. 根据任务要求画出控制电路的状态转换图。

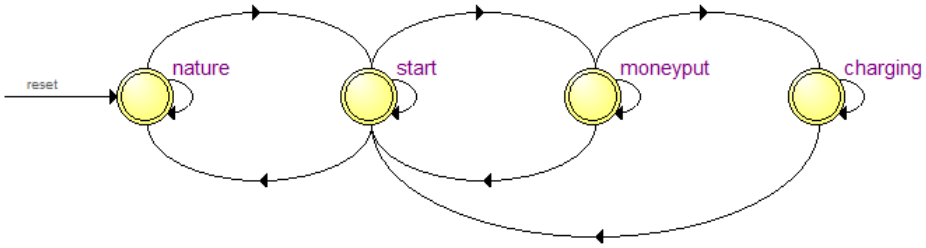



（1） Clk_250：分频器，将晶振产生的 50MHz 的时钟信号分频为 250Hz；

（2） Keyboard：键盘输入模块，采用行扫描方式，定义为无按键输入、扫描第一行、扫描第二行、扫描第三行、扫描第四行、按键输入信号等六个状态，在有按键时进行防抖处理，signal 为非抖动有效按键的输出信号；下图是生成的状态转换图；




（3） Statemachine:状态机模块，定义为 nature（初始状态）、start（开始状态）、moneyput（投币状态）、charging（充电状态），下图是生成的状态转换图：

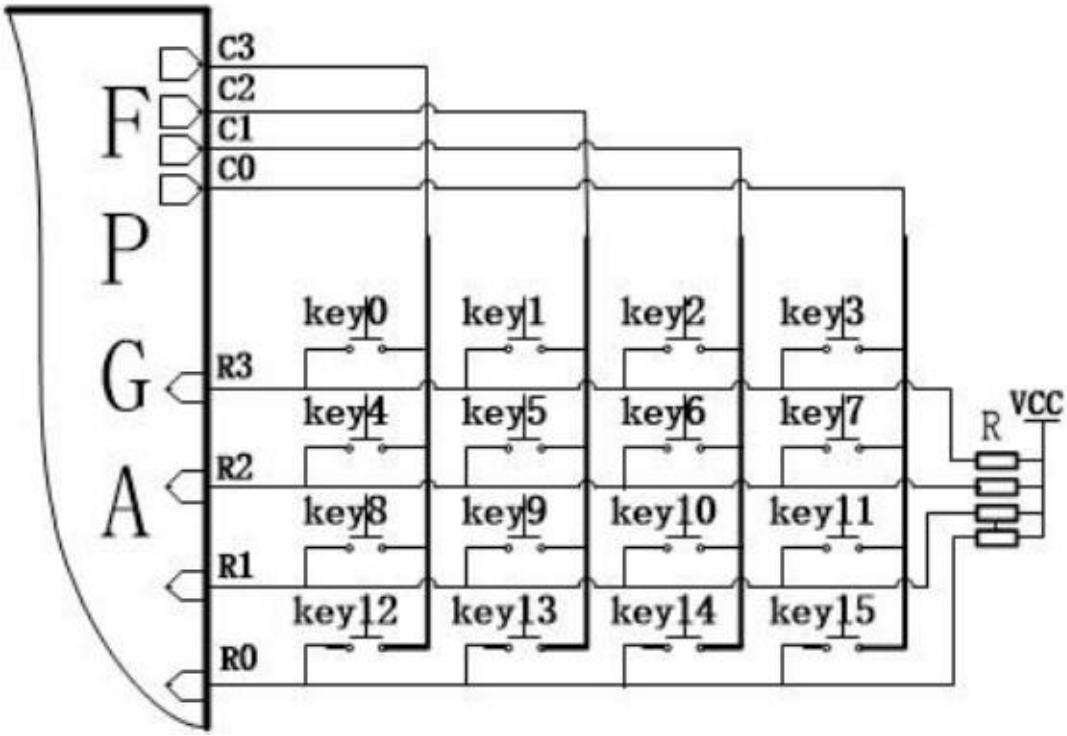



（4） DIG：数码管扫描显示模块，以 250Hz 时钟信号扫描显示数码管，利用人眼的暂留效应，使四个数码管产生同时显示数字的效果，利用获得的 money1、money0、time1、time0，在数码管上显示对应数字；

三、 设计思路及各模块功能分析

1. 矩阵键盘：

首先分析矩阵键盘的原理，FPGA 板中矩阵键盘的设计是在行线和列线的每个交叉点上设置一个按键，当某一个按键被按下时，行线上的电平值将改变。

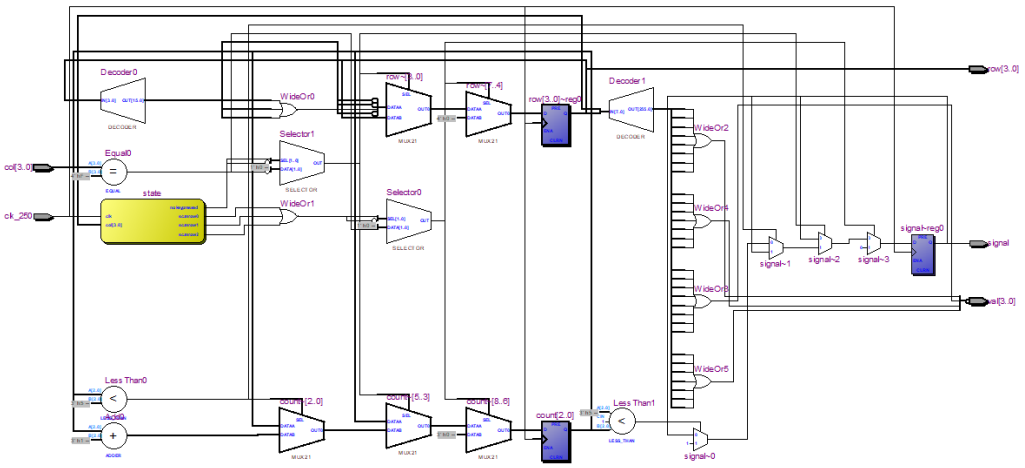



我的扫描思路是对行线 row0~row3 以跑马灯型式编码，循环输出“0111、1011、1101、1110”；同时检测列线的状态，如列线上均为高电平，即值为“1111”，表明没有按键按下。如检测到某列线上出现低电平，则表示键盘中有按键按下，同时开启行线的扫描，当某一行中有按键按下，转化为 keypressed，这时扫描暂停，相当于将此时的行线与列线编码固定住，每一对行列值都对应着不同的数字和功能按键。

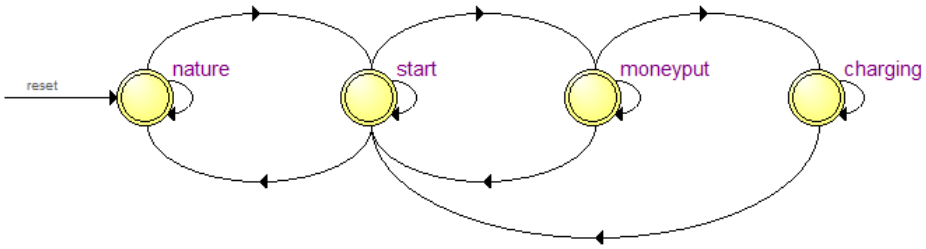
矩阵键盘的防抖功能：在 keypressed 状态采用一个计数变量 count，由于采用的是 250Hz 时钟信号进行扫描，考虑到按键抖动时间不会超过 20ms，因此当

计数变量从 0 计数至 5 时，即可认为是有效非抖动按键，这时将 signal 信号置为 1，后方连接的状态机模块读取按键值，否则视为抖动，signal 变量一直为 0。

矩阵键盘的防长按键功能：由于正常按键本身按键时间已经足够长，所以只需做到在 signal 有效信号输出时，只读取一次变量值即可，因此在之后的状态机模块取 signal 的上升沿作为 always 模块的敏感表信号，当上升沿到来时，读取 button 的值，并且只读取一次，以防止长按键多次输入数字的情况。




2. 状态机模块：




状态转换：

（1） 初始状态：四位数码管全灭状态，按下开始键后转化为开始状态；

（2） 开始状态：10s 无动作转化为初始状态，当输入数字键后转化为投币状态；

（3） 投币状态：按下清零键转化为开始状态，按下确认键转化为充电状态；

（4） 充电状态：充电结束后，返回到开始状态，同时将钱数清零；

功能实现：

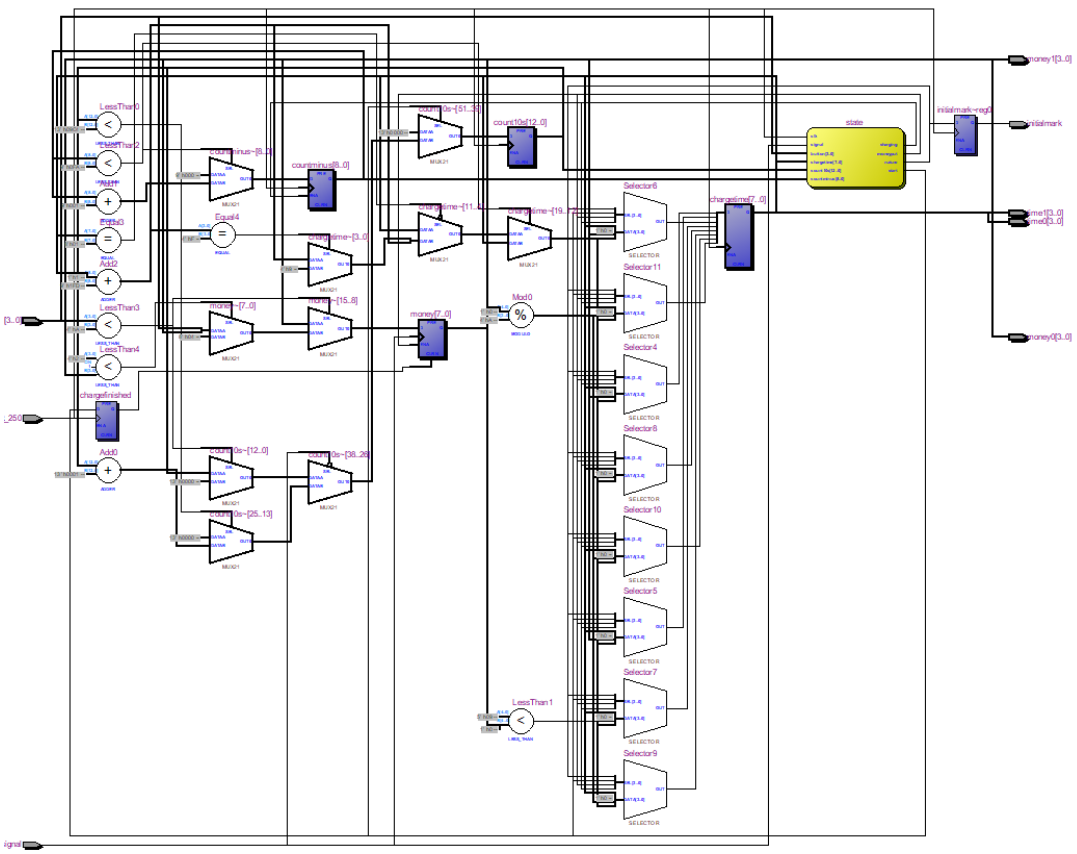
（1） 10s 无动作灭灯功能：同样，设定一个计数变量 count10s,由于采用 250Hz 时钟信号扫描，因此当计数变量计数至 2500，即计数 10s 的时间后，将状态转化为初始状态，同时将 initialmark 信号置为 1，输出后即数码管灭灯；

（2） 钱数输入功能：将 money 和 time 均置为 8 位二进制变量，这里 money 和 time 的二进制值并不是其真实值，而是用前四位表示第一个数字，后四位表示第二个数字，这样处理会在之后将各位的值传递给 DIG 输出模块时带来方便；取时钟信号的上升沿，读取输入的 button 值，因作业要求不允

许有高于 20 的钱数输入，若此时的 money[3:0]即后四位小于 2，将前四位值置为后四位，同时后四位的值置为 button；若此时的后四位大于 2，即将前四位置为 2，后四位置为 0；

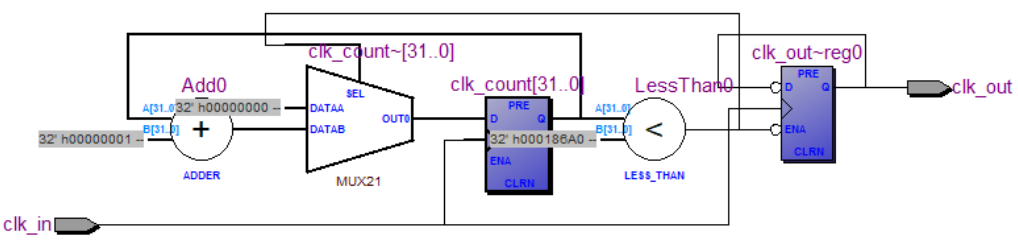
（3） 充电时间的计算：若此时 money[3:0]的值的 2 倍高于 9，chargetime[7:4]的值需要接受进位，因此等于 money 前四位的二倍加一；否则，chargetime[7:4]的值等于 money 的前四位的二倍即可；chargetime 的后四位等于 money 的后四位值的二倍除以 10 的余数；每当 money、chargetime 的值改变时，money1、money2、time1、time2 的值实时改变，实现了同步显示；

（4） 倒计时功能：同样引入一个计数变量 countminus，当计数变量达到 250 时，chargetime 的值减一，同时将 countminus 置零，如此循环直至 chargetime 的值减至 0，状态转化为开始状态；需要注意的是,因为 chargetime 是八位取值，也就是说，chargetime 在减数字的过程中，后四位会出现 1111 的情况，这时需要强制将其转化为 1001,这样就实现了时间真实值的减法运算。



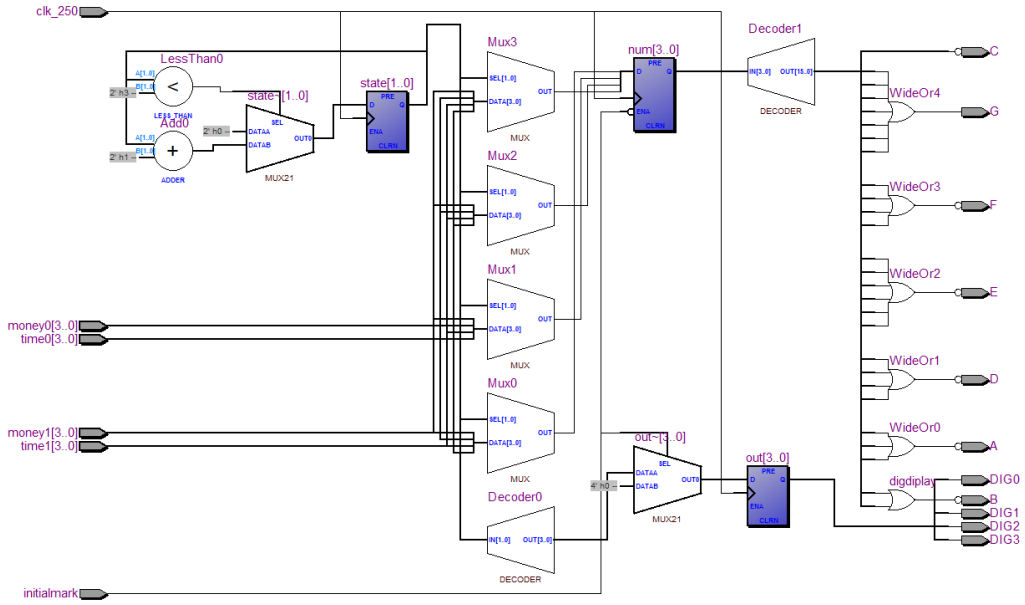
3. 分频器模块：

引入计数变量 clk_count，由晶振产生 50MHz 的信号，当计数变量计至 50_000_000/2*250=100_000 时，时钟信号取反，如此实现分频功能；



4. DIG 显示模块：

显示模块也相当于小型状态机，将其设置为四个状态，分别为扫描四个数码管，同时在其上显示按键对应的数字，将数字对应的段显示赋值给 a,b,c,d,e,f,g，实现显示功能；



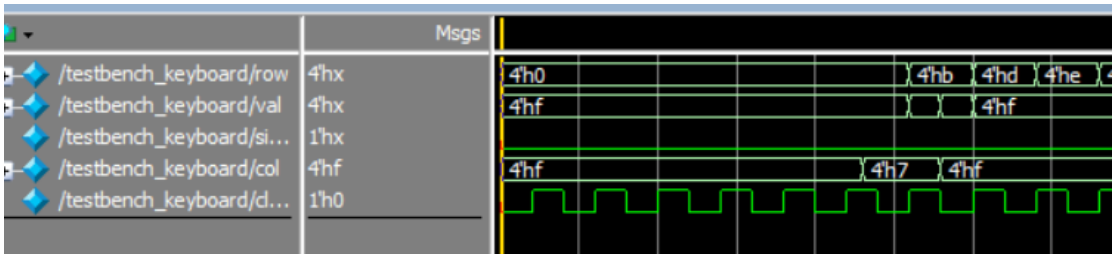

各模块引脚含义

| 模块 | 引脚名（I/O） | 功能 |
|---|---|---|
| 分频器模块 | Clk_in(I) | 晶振输入 50MHz 信号 |
| | Clk_out(O) | 分频输出 250Hz 信号 |
| DIG 显示模块 | DIG0\DIG1\DIG2\DIG3(O) | 四位数码管 |
| | a,b,c,d,e,f,g(O) | 七段数码管显示 |
| | Clk_250(I) | 250Hz 时钟信号 |
| | Money1[3:0](I) | 钱数十位 |
| | Money0[3:0](I) | 钱数个位 |
| | Time1[3:0](I) | 时间十位 |
| | Time0[3:0](I) | 时间个位 |
| | Initialmark(I) | 灭零信号 |
| 键盘输入模块 | Col[3:0](I) | 列线判断是否有键按下 |
| | Clk_250(I) | 250Hz 时钟信号 |
| | Row[3:0](O) | 行线判断是否有键按下 |
| | Initial(O) | 灭零信号 |
| | Signal(O) | 有效按键判断信号 |
| | Val[3:0](O) | 按键对应值 |
| 状态处理模块 | Button[3:0](I) | 按键对应值 |
| | Clk_250(I) | 250Hz 时钟信号 |
| | Signal(I) | 有效按键判断信号 |
| | Initialmark(O) | 灭零信号 |
| | Money1[3:0](O) | 钱数十位 |

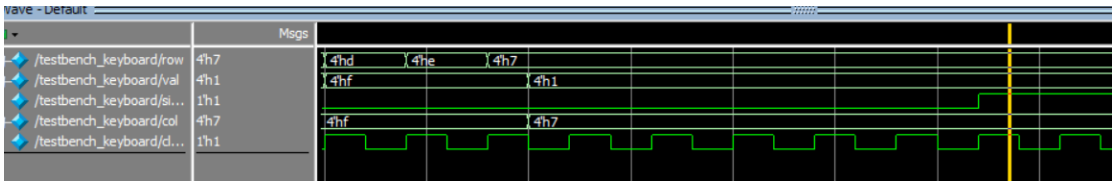
| | | |
|---|---|---|
| | Money0[3:0](O) | 钱数个位 |
| | Time1[3:0](O) | 时间十位 |
| | Time0[3:0](O) | 时间个位 |

四、仿真测试及验证

1. 键盘仿真：



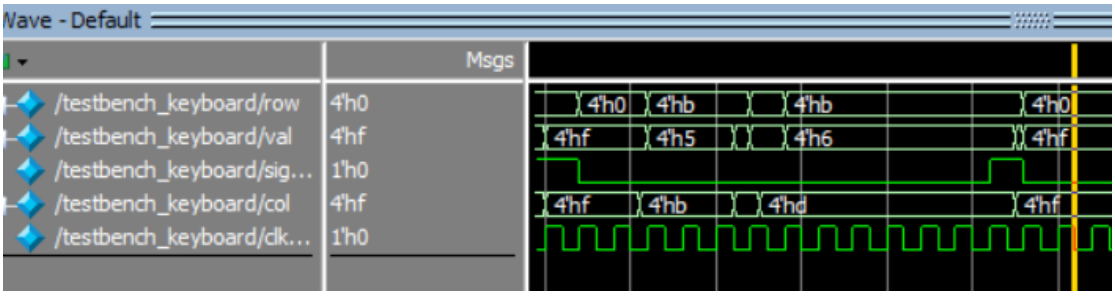

图一




图二

从仿真波形图一中可以见到，当有输入（对应列 col 为 4’h7 时）直到输入结束，signal 并不会变化，原因是持续时间太短，在波形图中只有 3ms，因此视为抖动，系统并不会认为这是一个有效按键；

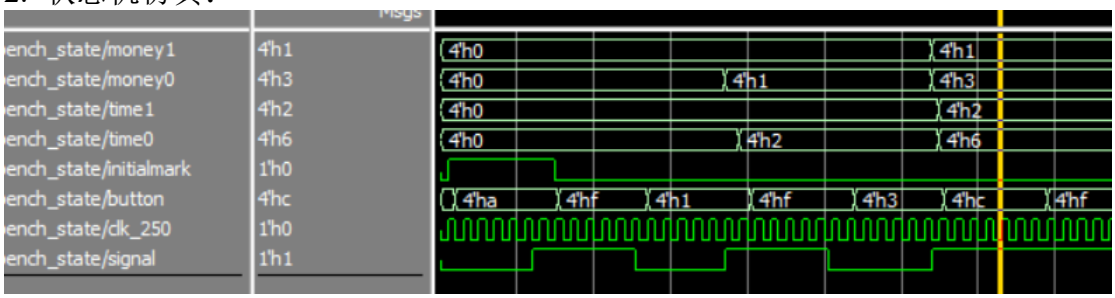
在仿真波形图二中，可以看到当 col 输入时间足够长（4’h7）,持续超过 20ms，系统认为这是一个有效按键，在经过大约 5 个时钟周期后 signal 变为 1，将触发信号传递给状态机；验证了防抖功能的正确。




图三

之后的波形同样验证了这一点，如图三所示，当 col 值变为 4’hb 后，虽然同时在行列处于不同状态时，val 值也相应地被同步赋为相应的值，但是因为这一按键持续时间太短（约 10ms），signal 信号并不会提供输出，之后 col 变为 4’hd，并且持续 30ms，认为是正常的按键，signal 在 20ms 判定时间结束之后变为 1；同时上述不同取值也可以验证键盘不同数字对应关系正确。

2. 状态机仿真：

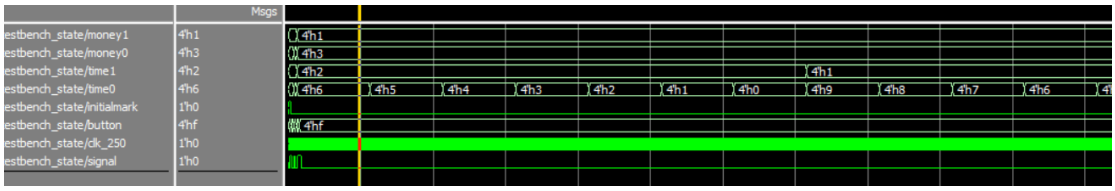


图四

初始 initialmark 变量为 1，系统处于灭灯状态，当按下开始键（对应 4'ha）后，经过 20ms，signal 变为 1，经过一个时钟周期后，initialmark 变为 0，系统处于开始状态；

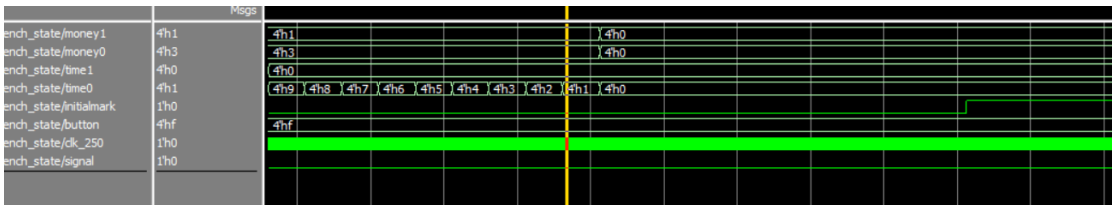
当按下按键 1 时（对应 4'h1），经过 20ms，signal 置为 1，同时 money0 和 time0 被赋以相应的值；

当再按下按键 3 时（对应 4'h3），经过 20ms，确认为有效输入，同时 money1,money0,time1,time0 被赋以相应的值；




图五

在图五中，为启动倒计时，可见钱数一直稳定显示，时间每隔 1s 的时间减少 1；




图六

在图六中，为倒计时结束时，可见在持续 10s 没有动作后，initialmark 信号置为 1，系统进入初始状态同时灭灯；

这样就验证了所有的情况。

五、设计和调试中遇到的问题及解决方法

1. 这次大作业真的是遇到了好多玄学 bug，首先是状态转换图始终无法生成，后来才知道是 state 变量作为输出的原因（因为一直在用 state 变量作为输出看仿真是否正确），为了状态转换图我修改了 n 次状态机代码，也在这里耽误了很多时间；

2. 键盘输入值的时候，之前总是要按两下才可以，后来意识到因为最初设计时我是取 signal 上升沿读值，而有效按键对应的行列值也是在上升沿才被赋值，并且在行列值确定之后才会确认按键值，因此该上升沿读取按键可能存在延迟或无效；解决办法是将 row，col 一直动态赋值，在 signal 上升沿到达时直接读取；

3. 10s 无动作，开始状态转化为初始状态，之前我的 count10s 变量只在 start 状态中赋值变化，这样导致的问题是：当 start 状态时有输入导致状态变化时，count10s 的值并不会置 0；当再次回到开始状态后，时间小于 10s 数码管就会灭掉，解决办法是在其他三个状态中都将 count10s 置零；

4. 充电时间倒计时时，当时间倒计时为 0 时，钱数并不会马上置零，后来发现是计数变量控制 chargetime 减少的过程中多进行了一次循环，因此延迟 1s，修改循环判定条件后问题得以解决。

六、 作业总结

1. 这次作业需要用到 verilog 语言，在之前的几次实验和大作业 1 的时候，我就

初步接触了语言的学习，因此在大作业 2 发布后，语言的学习没有成为很大的障碍，但是 debug 的过程依然艰辛；
2. “仿真”，顾名思义，是要仿出实际情况，我在本次作业也是格外注意了这一点，对于键盘模块，在按键抖动和长按键时间的处理上进行了妥善安排；对于状态机模块，按键出现赋值之后，signal 信号只有在 20ms 之后才会出现高电平，确认为有效按键，因此在仿真时，signal 信号的时间设置要晚于 button 值 20ms，不宜出现其他时间；同时，仿真尽量要覆盖全面，将可能出现的各种情况尽量都显示在波形图上，逐项检查，这样下载到 FPGA 板上时也不会出现很多问题；
3. verilog 语言的束缚确实很少，但是也是真的容易出错，VHDL 看上去比较复杂，有很多模块化的看上去冗余的语言，这也是我抵触和没有学习 VHDL 的原因，直观上感觉 verilog 和 VHDL 更像是 C 语言和 C++的关系，之后有时间也是想体验下 VHDL 语言的应用；
4. 这次作业比较贴近生活实际，同时设计想法具有很多优点。首先是客户友好性，比如在未确认充电前均可清零，防止因为输错按键造成损失；还有资源节约性，在开始状态 10s 后无动作系统自动灭灯，减少能源损耗等等。这也提醒我们在设计一款产品时，要综合考虑多方面的因素，而不应该仅仅停留在“能用就行”的层面。

最后感谢老师和助教在大作业完成和验收的过程中给予我的帮助！