数字逻辑与数字系统设计 实验大作业报告 (2020年)

课程名称: 数字逻辑与数字系统设计

任课教师:李琼

作业题目: 电子密码锁设计

完成人: 傅浩东

学号: 1190202105

班级: 1903002

报告日期: 2020年12月19日

报告成绩	
教师评语	

目录

第1节 设计要求	3
第 2 节 工作原理及系统方框图	3
2.1 工作原理	3
2.2 系统方框图	4
第 3 节 各部分模块具体功能及设计思路	5
3.1 初始化	5
3.2 扫描信号分频模块	5
3.3 倒计时模块	6
3.4 比较判断输入是否正确	6
3.5 密码与七段数码管转换模块	6
3.6 输入密码显示模块	6
3.7 时间倒计时和错误次数显示模块	7
3.8 输入密码模块	7
3.9 更改密码模块	8
3.10 主模块	8
第 4 节 调试过程	8
4.1 管脚编号错误	8
4.2 电平出现未知错误	9
4.3 管脚约束文件未知错误	9
4.4 某些情况,多个 always 模块对同一个变量赋值出错	10
4.5 一个 always 模块多个上升下降沿与条件冲突	10
4.6 数据类型错误	10
4.7 未知错误	10
第 5 节 设计结论	11
5.1 设计成果	11
5.2 待改进之处	11
第 6 节 设计心得与总结	12
第 7 节 参考文献	13

附录	14
附录一 总体设计图	14
附录二 各模块仿真截图	15
B.1 比较模块	15
B.2 密码输入模块(修改密码相似)	15
B.3 密码转换 LED 模块	15
B.4 判断正确与否模块	16
B.5 倒计时数据显示模块	16
附录三 工作说明与贡献	16

第1节 设计要求

- 1. 设计一个开锁密码至少为 4 位八进制数字的密码锁。
- 2. 当开锁按扭开关的输入的密码等于所设密码时启动开锁控制电路,并且用 七段数码管 *OPEN* 表示开锁状态。
- 3. 从开始按扭触动后的 10 秒内若未能将锁打开,则电路自动复位并发出报警信号,即用七段数码管 *EROR* 表示关锁状态。
- 4. 记录错误次数, 当错误次数达到三次后, 密码锁全部关闭。
- 5. 五秒倒计时用七段数码管显示。
- 6. 密码锁四位密码可进行修改。
- 7. 功能锁键,可以暂停输入进程。
- 8. 在修改密码过程中用小数点表示正在修改的位数。

第2节 工作原理及系统方框图

2.1 工作原理

电子密码锁的工作由以下几个部分完成:

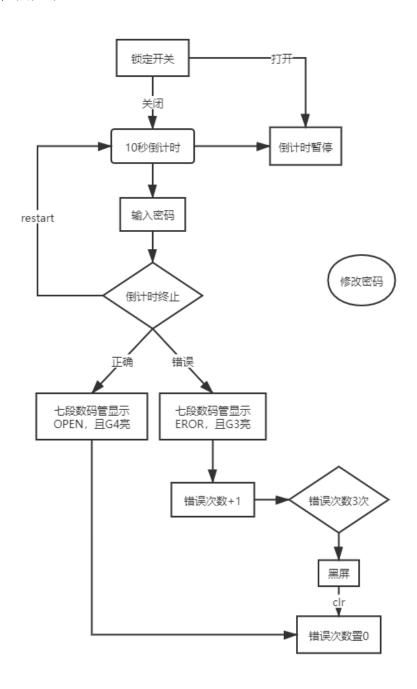
- 1. 初始化: 录入初始状态(包括设定密码锁的内置密码等)。
- 2. 输入密码/修改密码: 从最高位开始,监测八个开关的电平总变化变化, 若是开关总体有脉冲信号,当为上升沿时记录一组电平值;在结束记录 后,位数右移,准备记录下一位的电平值(若在最低位,则输入完成后 回到最高位)。
- 3. 分频: *EGO1* 开发板自带时钟频率为 100MHz,在该模块中对系统自带时钟进行分频,分成四组便于对显示四位密码的四个七段数码管进行动态显示,以及对时钟和错误次数的动态显示。
- 4. 数字显示: 动态扫描实现输入密码、倒计时以及错误次数的七段数码管显示。
- 5. 比较:比较密码和输入密码是否相同,若相同则七段数码管数码管显示 *OPEN*,否则显示 *EROR* 且错误次数加一,直到错误次数为 3,屏幕整体。
- 6. 倒计时功能: ①从第一个按扭触动进行 10 秒倒计时, ②倒计时重置, ③

倒计时暂停。

7. 重置:密码错误达到三次黑屏,可以进行重置使得整体恢复初始状态,但密码没有改变。

2.2 系统方框图

系统方框图如下:



第 3 节 各部分模块具体功能及设计思路

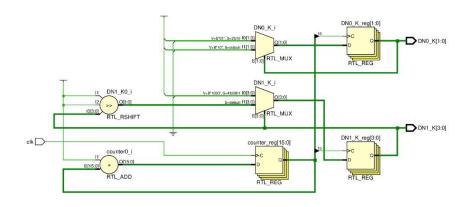
3.1 初始化

初始化密码锁系统中的变量。其中较为重要的变量有:

- 1. Correct_password 三十二位二进制数表示正确密码, 初始化为四个二进制的 八位数,即 4×8'b00000010,表示初始密码为四个六。
- 2. my_password:每八位对应一个人为输入密码,初始化为 0,它们在数码管上用四位来显示。通过密码与七段数码管的对应转换关系,可以转换为变量 led 1~led 4,扫描实现七段数码管的显示。
- 3. ChooseChange/ChangePassword:表示当前输入的位数,初始化为四位密码的第一位,即为 4'b1000。
- 4. IsLock, right, wrong, resetting: 表示密码锁当前状态的变量, 初始化时 给其赋值, 使其表示上锁状态。
- 5. NumWrong: 错误次数初始化为 0。

3.2 扫描信号分频模块

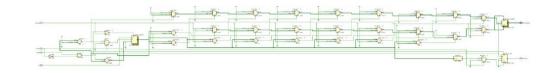
扫描信号分屏是通过一个 16 位的 reg 类型变量 counter 实现分频。每当时钟上升沿时 counter 信号自加一,使其在 0 ~ 16'd1111_1111_1111_1111 之间循环变化。在其最高位数字发生变化的情况下,对应的输出 DN1_K 和 DN0_K 依次向右循环移动一位,可以为 4'b1000、4'b0100、4'b0010、4'b0001 和 2'b10、2'01,正好对应两组六个七段数码管的显示。这样便利用 EGO1 开发板内置的 100MHz 时钟实现了简单的分频功能。



3.3 倒计时模块

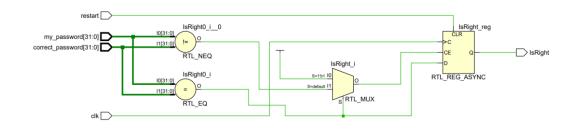
倒计时模块运用了与分频模块类似的原理: 构建 reg [31:0] counter, counter 的变化范围为 32'd0 ~ 32'd900_000_000, 对应着 10 秒的倒计时。

此模块还受 restart、resetting 及 IsLock 变量的状态控制,由此来实现倒计时的重置与暂停功能。



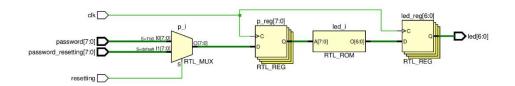
3.4 比较判断输入是否正确

将输入的密码与密码锁的正确密码(均用 32 位寄存器的形式表示)进行比较,根据比对结果 $my_password$ 和 $correct_password$ 的相同性,判断输出 lsRight 为 0 或 1。



3.5 密码与七段数码管转换模块

将四个 8 位密码 password 转换成对应着七段数码管 A、B、C、D、E、F、G 各部分显示的数组,即 led 显示,由此来实现密码显示。

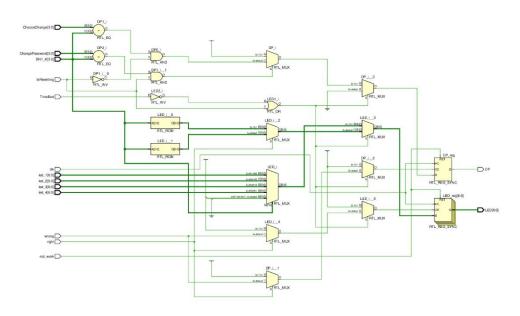


3.6 输入密码显示模块

时钟 clk 上升沿用 case 语句对分频部分得到的 DN1_K 进行一次判断。若错

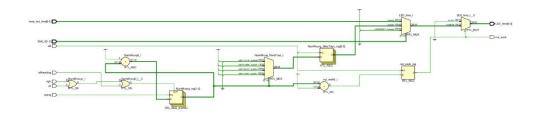
误次数达到三次则不显示,否则根据 $DN1_K$ 的值(某一位为高位,其余为低位) 决定将 $led_1 \sim led_4$ 中的某个信号值通过动态扫描显示到数码管上。当倒计时结束,若输入正确则显示 OPEN,若错误则显示 EROR。

当 ChooseChange 与 DN1_K 相等时,该位七段数码管小数点闪亮一次,由于刷新频率足够快,在人眼看来其保持常亮。



3.7 时间倒计时和错误次数显示模块

同上,通过分配信号扫描显示错误次数与倒计时。

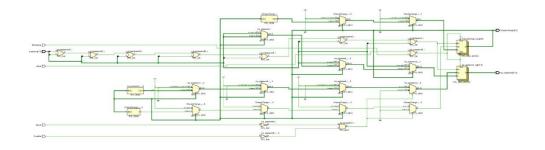


3.8 输入密码模块

输入 [7:0] password 对应 8 个拨动开关的状态,检测到任意一个开关被拨动 (产生上升沿),则记录当前开关状态到表示位数的变量 ChooseChange 所对应的 密码位 my_password[31:24]\[23:16]\[15:8]\[7:0],并且对 ChooseChange 进行一次 循环右移操作,当下次检测到开关波动产生上升沿信号则输入下一位。

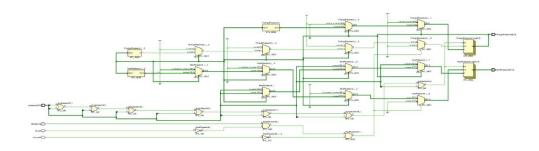
同样,输入操作受 restart、TimeEnd、resetting 等功能的影响。例如,当检

测到倒计时结束后,启动重新开始功能,ChooseChange 重置为 4b'1000 (即第一位),并且通过修改 my_password 的值为初始值 0,使数码管显示回归初始状态。



3.9 更改密码模块

当选择更改密码,resetting 信号置为 1,由此原输入密码显示模块显示更改密码且与输入密码输入模式相同。



3.10 主模块

以上模块并行。

第4节 调试过程

4.1 管脚编号错误

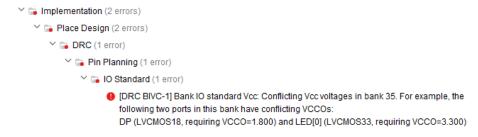
在测试过程中发现问题:拨动按钮输入密码,七段数码管上的显示均无变化。 经检查后发现问题在于将灯的管脚编号设置为了拨动开关的编号。由于综合、实现、生成比特流文件过程均未报错,所以一直认为是密码锁程序的逻辑出现问题, 未往管脚文件的方向思考,在这一实际非常简单的错误上耽误了较多时间。

如下错误代码,需改为拨动开关对应引脚:

set_property PACKAGE_PIN F6 [get_ports password[7]]
set_property IOSTANDARD LVCMOS33 [get_ports password[7]]

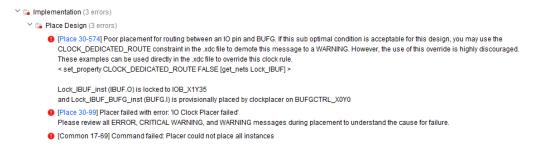
set_property PACKAGE_PIN G4 [get_ports password[6]]
set_property IOSTANDARD LVCMOS33 [get_ports password[6]]
set_property PACKAGE_PIN G3 [get_ports password[5]]
set_property IOSTANDARD LVCMOS33 [get_ports password[5]]
set_property PACKAGE_PIN J4 [get_ports password[4]]
set_property IOSTANDARD LVCMOS33 [get_ports password[4]]
set_property PACKAGE_PIN H4 [get_ports password[3]]
set_property IOSTANDARD LVCMOS33 [get_ports password[3]]
set_property PACKAGE_PIN J3 [get_ports password[2]]
set_property IOSTANDARD LVCMOS33 [get_ports password[2]]
set_property IOSTANDARD LVCMOS33 [get_ports password[1]]
set_property IOSTANDARD LVCMOS33 [get_ports password[1]]
set_property IOSTANDARD LVCMOS33 [get_ports password[0]]
set_property IOSTANDARD LVCMOS33 [get_ports password[0]]

4.2 电平出现未知错误



管脚文件缺失语句 set_property IOSTANDARD LVCMOS33 [get_ports DP], 导致 Run Implementation 步骤失败。修正后 Run Implementation 步骤成功通过。

4.3 管脚约束文件未知错误



当要使用按键的输入上升或下降沿信号时,会有如上报错,在管脚约束文件对应的按键添加 set_property CLOCK_DEDICATED_ROUTE FALSE [gets_nets XXXX], 其中 XXXX 表示某个按钮对应的输入,这样在规避报错的同时还可以消除按钮抖动。

4.4 某些情况, 多个 always 模块对同一个变量赋值出错

在一个以上的 always 块中对同一个变量赋值容易产生竞争冒险,且不能综合。如果一个变量的值必须与两个 always 事件相关联则将它们放在同一个 always 模块里面,或者一个 always 块只对一个变量赋值,最后将其用组合逻辑将其变为一个信号。本人主要采用第一种方法,通过逻辑组合将它们放在一个 always 模块中,但由此也产生了如下 4.5 错误。

4.5 一个 always 模块多个上升下降沿与条件冲突

当一个 always 模块条件中有多个上升下降沿的使用,若采用多个 if 条件语句并行,有时会产生如下报错。更改方法为,将多个 if 语句合并成 if-else 语句

- (Synth 8-91) ambiguous clock in event control [lock.v:112]
- [Synth 8-6156] failed synthesizing module 'lock' [lock.v:3]
- (1) [Common 17-69] Command failed: Synthesis failed please see the console or run log file for details

4.6 数据类型错误

Reg 和 wire 数据类型使用混乱,导致出现如下错误:

✓ □ Synthesis (3 errors)

- (9) [Synth 8-685] variable 'correct_password' should not be used in output port connection [InputPassword.v:79]
- (Synth 8-6156) failed synthesizing module 'Main' [InputPassword.v:22]
- [Common 17-69] Command failed: Synthesis failed please see the console or run log file for details

4.7 未知错误



- DRC NSTD-1] Unspecified I/O Standard: 7 out of 38 logical ports use I/O standard (IOSTANDARD) value 'DEFAULT', instead of a user assigned specific value. This may cause I/O contention or incompatibility with the board power or connectivity affecting performance, signal integrity or in extreme cases cause damage to the device or the components to which it is connected. To correct this violation, specify all I/O standards. This design will fail to generate a bitstream unless all logical ports have a user specified I/O standard value defined. To allow bitstream creation with unspecified I/O standard values (not recommended), use this command: set_property SEVERITY {Warning} [get_drc_checks NSTD-1]. NOTE: When using the Vivado Runs infrastructure (e.g. launch_runs Tcl command), add this command to a .tcl file and add that file as a pre-hook for write_bitstream step for the implementation run. Problem ports: LED_time_wrongtime[6:0].
- ① [DRC UCIO-1] Unconstrained Logical Port: 7 out of 38 logical ports have no user assigned specific location constraint (LOC). This may cause I/O contention or incompatibility with the board power or connectivity affecting performance, signal integrity or in extreme cases cause damage to the device or the components to which it is connected. To correct this violation, specify all pin locations. This design will fail to generate a bitstream unless all logical ports have a user specified site LOC constraint defined. To allow bitstream creation with unspecified pin locations (not recommended), use this command: set_property SEVERITY {Warning} [get_drc_checks UCIO-1]. NOTE: When using the Vivado Runs infrastructure (e.g. launch_runs Tcl command), add this command to a.tcl file and add that file as a pre-hook for write_bitstream step for the implementation run. Problem ports: LED_time_wrongtime(6:0).
- (9) [Vivado 12-1345] Error(s) found during DRC. Bitgen not run.

生成比特流文件过程中,会有如上报错,查阅资料,添加 tcl 文件,含如下三句

set_property SEVERITY {Warning} [get_drc_checks NSTD-1]
set_property SEVERITY {Warning} [get_drc_checks RTSTAT-1]
set_property SEVERITY {Warning} [get_drc_checks UCIO-1]

第5节 设计结论

5.1 设计成果

根据大作业题目要求,完成了带有6个附加功能的四位八进制密码锁。其主要功能为:

- 1. 内置四位开锁密码,通过八个开关实现密码输入,八个拨动开关每次只有一个能为输入状态,所以共产生八个输入组合,对应着 0~7 八个数字。
- 2. 从拨动第一个开关后有 10 秒输入时间, 10 秒倒计时结束后进行密码比对, 若密码输入正确则密码正确指示灯亮, 密码锁进入开锁状态, 数码管显示复位。
- 3. 从拨动第一个开关后有 10 秒输入时间, 10 秒倒计时结束后进行密码比对, 若密码错误未能将锁打开,则电路自动复位,密码错误指示灯亮,密码锁进入关锁状态。

密码锁的 6 个附加功能分别为:

- 1. 错误密码数码管显示 EROR, 正确密码显示 OPEN。
- 2. 时间倒计时显示。
- 3. 修改密码。
- 4. 小数点表示修改位数。
- 5. 错误次数统计,在错误三次后息屏,且可重置。
- 6. 八进制密码。

5.2 待改进之处

本次设计的密码锁的功能离能够投入实际使用的阶段还有一定距离,以下是所设想的一些待改进之处:

1. 实现更多位数(如六位,八位)密码的密码锁。

- 实现自行选择输入的位数,目前只能实现逐次按位输入,一旦一位输错则需要重新输入一轮密码,较为麻烦,打算设计为可左右移动的密码输入方式。
- 3. 通过用户按键进行密码输入正误的判断,目前只能在倒计时结束后进行一次输入正误的判断。
- 4. 在修改密码后会出现部分显示混乱,原因是没有即使对变量进行更改, 由于改动过大就不再继续。

第6节 设计心得与总结

通过这次大作业,我有以下心得与总结:

- 1. 深切感受到了理论设计与工程实现之间仍存在巨大的差距。由于实际元器件的参数性能影响等,vivado激励仿真通过并不意味着实际上板的结果也毫无错误。在大作业的过程中,我不止一次体会到了理论与实践之间的巨大差距,而这类 bug 往往是最隐蔽,最耗费精力才能找出最终解决方案的。
- 2. 对 FPGA 数字电路设计有了更深刻的了解。由于是第一次设计这种规模的数字电路系统,在使用 vivado 软件进行功能实现、管脚设定、程序仿真时遇见了很多之前没有遇到过的 bug。在解决过程中耗费了很多时间与精力。但对我来说,这也有好的一面:若是再进行同等规模的系统设计,由于熟悉了整个流程,自己的效率应该会有所提高。
- 3. 由于之前不能熟练了解 Verilog 语言特性、语法以及对数字逻辑系统设计流程的认识不足,我在完成这次大作业的过程中走了许多弯路。比如在在思考实现密码锁功能时想到哪一块就直接开始动手写代码,没有一个总体规划的阶段,这导致了很多实现主要功能的程序段没有封装在模块中,妨碍了仿真调试,也影响了 RTL 综合结果的美观性。
- 4. 让我对自己完成大作业的心态有所反思。通过这次大作业,我认识到了要根据对自己能力的预期合理拓展密码锁程序。刚开始编写程序时构想了不少高级的附加功能(如通过按键实现输入位数的转换等),但初步写

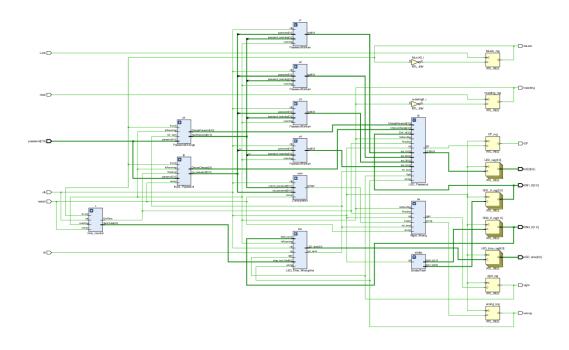
- 好这些功能后,测试发现存在着很多莫名其妙的错误,在最后不得已将它们中的大部分删除。
- 5. 考虑运用 design-block 模块来进行连接,但由于路线之间的连接繁琐就放弃了,完成密码锁之后又认为这个方法应该可以一试。

第7节 参考文献

- [1] CSDN.verilog 入门教程_申缘-CSDN 博客_verilog 语言入门教程[EB/OL]. 2014[2020-12-5]. https://blog.csdn.net/caojunjun12345/article/details/31357785.
- [2] 博客园.Vivado 常见问题集锦 NingHeChuan 博客园[EB/OL]. 2017 [2020-12-6]. https://www.cnblogs.com/ninghechuan/p/7247441.html.
- [3] 博客园.按键消抖方案的 verilog 描述 自由的青 博客园[EB/OL]. 2017 [2020-12-8].https://www.cnblogs.com/qingkai/p/7596126.html.
- [4] CSDN.VerilogHDL 常用的仿真知识_tianpu2320959696 的博客-CSDN 博客 [EB/OL].2018[2020-12-9].https://blog.csdn.net/tianpu2320959696/article/details/81063681.
- [5] verilog 时钟分频设计 https://blog.csdn.net/moon9999/article/details/75020355

附录

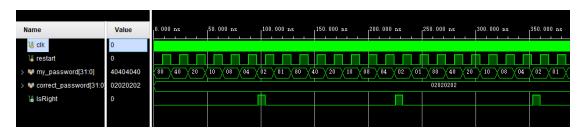
附录一 总体设计图



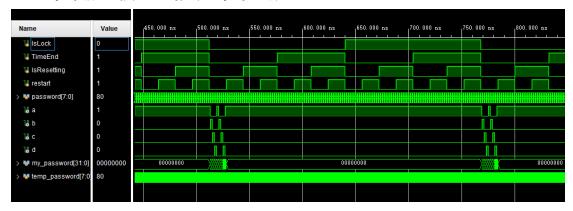
附录二 各模块仿真截图

由于部分模块仿真耗时长,结果无太大意义,此处只仿真主要几个模块。

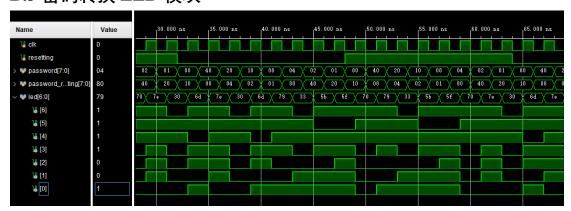
B.1 比较模块



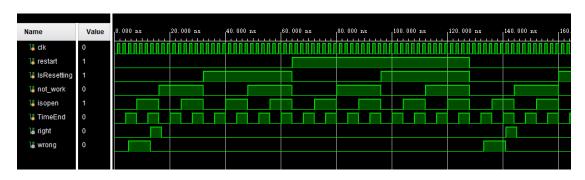
B.2 密码输入模块(修改密码相似)



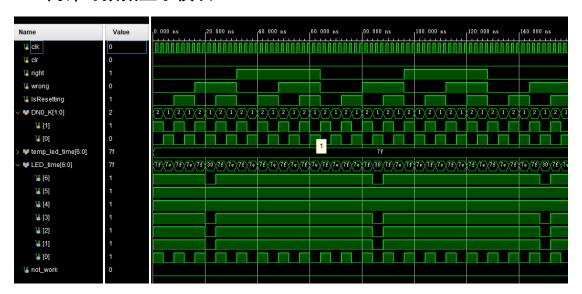
B.3 密码转换 LED 模块



B.4 判断正确与否模块



B.5 倒计时数据显示模块



附录三 工作说明与贡献

本人单独成组,独立完成各项工作。