CS207 数字逻辑项目文档

0. 团队分工及开发计划日程安排和实施情况

0.1 团队分工

组长：倒计时模块，显示模块，菜单模块（飓风模式），系统时间模块，工作时间模块，贡献百分比为 37%；

组员 1：开关机模块，手势模拟模块，菜单模块（各档位切换），贡献百分比为 33%；

组员 2：智能提醒模块，高级设置模块，贡献百分比为 30%；

后期模块整合、顶层模块设计均由三人共同完成。

0.2 开发计划日程安排

第十三周：组队成功，经过讨论各功能在按键与拨键的实现位置，初步确定系统使用说明，周末分配任务，确定变量命名和注释等格式；

第十四周：各自写好个人负责的模块，并在周末进行整合；

第十五周：三人共同设计顶层模块，并对整合过的完整代码查找问题与改正。

0.3 实施情况

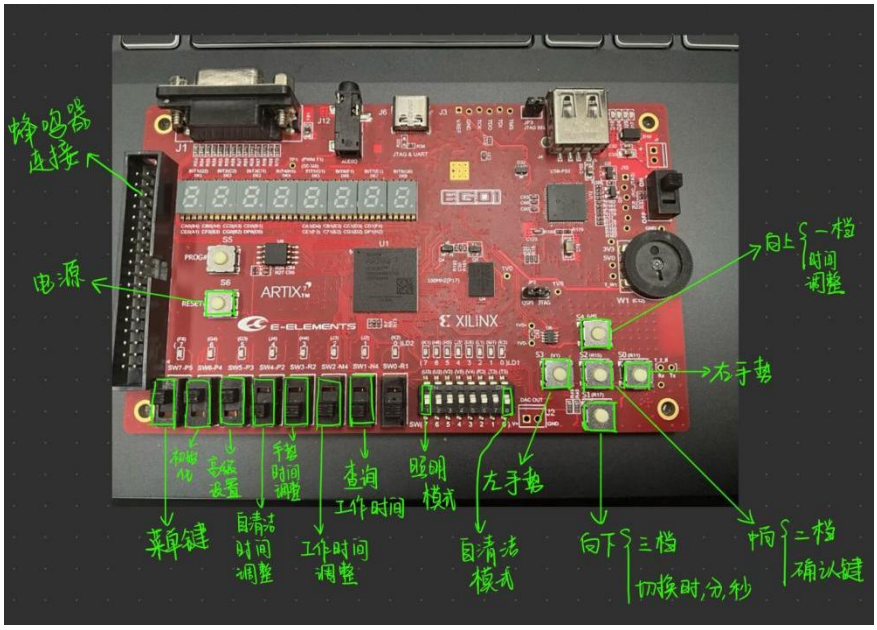
个人分模块写代码的任务延迟了两至三天完成，其余任务正常进行与完成

1. 系统功能列表

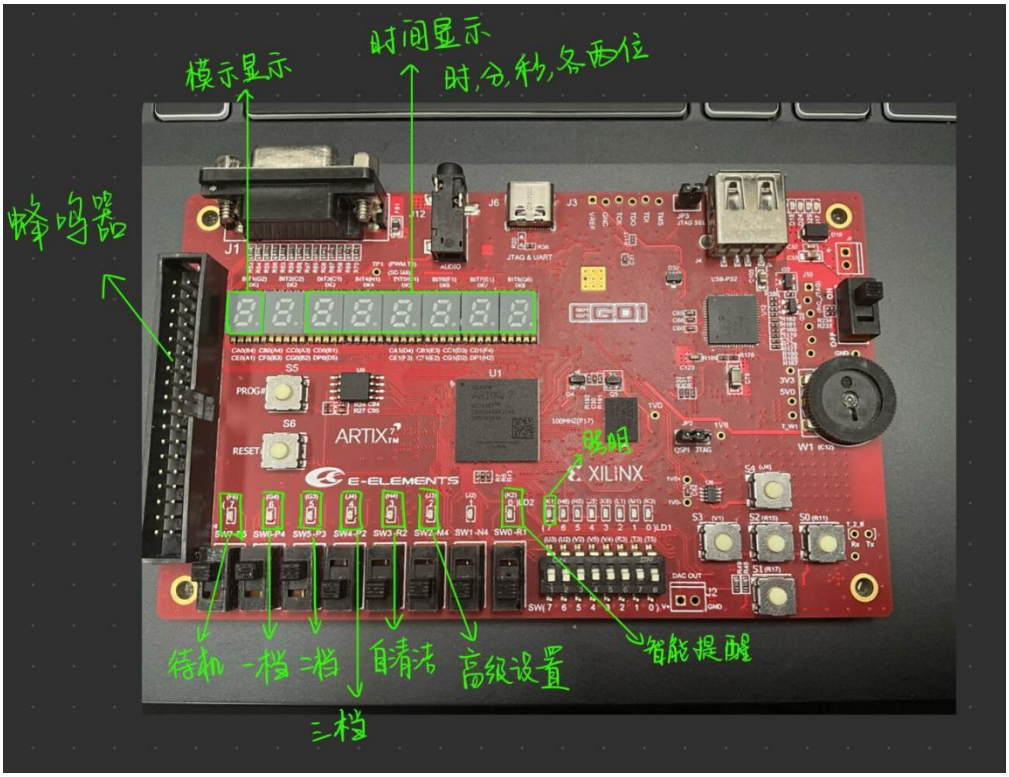
https://bb.sustech.edu.cn/bbcswebdav/pid-498233-dt-content-rid-16910279_1/courses/CS207-30022126-2024FA/digital_logic_2024f_projects_introduction_%E4%B8%AD%E6%96%87.pdf

2. 系统使用说明

输入：



输出：



3. 系统结构说明

最上层为 menu_module 模块，即菜单模块，控制着开关机并通过输入判断下一步进入待机、各工作档位、自清洁、高级设置等状态。在 menu_module 模块里，实例化了 count_down 模块、working_time 模块、system_time 模块、advance_settings 模块、show_time 模块、sound_player 模块。

第二层有 count_down 模块、working_time 模块、system_time 模块、advance_settings 模块、show_time 模块、sound_player 模块。其中 count_down 模块用于飓风模式和自清洁模式的倒计时，working_time 模块用于记录当前工作时间，system_time 模块用于记录当前系统时间，advance_settings 模块用于进入高级设置功能、show_time 模块用于显示当前系统或工作时间，sound_player 模块用于播放开关机时的提示音。

在 count_down 模块中实例化了第三层 count_down_light_7seg 模块，用于控制倒计时出现各个数字的数码管，count_down_light_7seg 模块又实例化了第四层 show_7seg_ego1 模块，用于控制每个数码管的亮灯情况；working_time 模块中实例化了第三层 working_time_light_7seg 模块，其中又实例化了 show_7seg_ego1 模块，作用与上一模块同理，均为控制显示的时间；system_time 模块中实例化了第三层 light_7seg_ego1 模块，其中又实例化了 show_7seg_ego1 模块，show_time 模块中实例化了第三层 show_time_light_7seg 模块，其中又实例化了 show_7seg_ego1 模块，此两个模块中子模块的用途也同理；advance_settings 模块中实例化了第三层 time_setting_controller 模块，用于指明调整系统、智能提醒、手势时间中的哪几个，time_setting_controller 模块中又实例化了第四层 time_setting 模块，用于通过位置按键和加法按键两个输入变量来设置新时间；

sound_player 模块则不存在子模块。

4. 子模块功能说明

4.1 menu_module 模块的说明

输入变量 power_on 表示开关机操作，按下菜单键（即输入变量 menu_btn）进入切换模式的状态，输入变量 up_toward 表示上按键，开始一档工作；mid_toward 表示中间按键，开始二档工作；down_toward 表示下按键，开始三档工作；clean_btn 表示拨码开关 T5 进入自清洁模式；advanced_clean 表示拨码开关 P2 进入高级设置，输出变量 mode 代表了切换哪一种状态，其中 s0 代表待机，s1 代表一档工作，s2 代表二档工作，s3 代表三档工作，s4 代表自清洁，s5 代表高级设置。

4.2 count_down、working_time、system_time、show_time 模块的说明

输入变量 sec_input、min_input、hour_input 分别表示当前时间的秒、分时，输出变量的 seg_en、seg_out0、seg_out1 则代表数码管的显示。

4.3 advance_settings 模块的说明

输入变量 advance_settings 表示进入高级设置；time_btn 表示时间调整按键；mode_clean_btn 表示智能提醒时间按键；mode_gesture_btn 表示手势时间按键；mode_system_time_btn 表示系统时间按键；confirm_btn 表示设置完成后的确认按键。输出变量的 seg_en、seg_out0、seg_out1 依然表示数码管的显示；mode_1_time_hour 至 mode_3_time_sec 分别表示智能提醒时间、手势时间、系统时间设置的时、分、秒。

5. bonus 的实现说明

sound_player 模块实现了一个简单的声音播放功能，通过时钟分频得到合适的低频控制时钟，基于此生成音符频率相关数据，进而利用计数器等逻辑生成最终的声音控制信号，以驱动外部设备产生具有一定频率变化规律的声音，模拟出类似音乐播放的效果。

6. 项目总结

经过本次数字逻辑项目的合作，从原来的不适应到后来相互磨合，我们体会到了合作完成项目的重要性。在合作中，我们充分交流了实验课所学内容，在互相教导中弥补了各自存在的短板，从而互相进步。运用实验课的知识，我们起初在编写各自模块的代码时比较顺利，按时完成了任务，但在模块整合中发现了许多问题，并花了几天的时间发现并改正存在的错误，因此进度也稍有落后。通过此次修复各种漏洞的经历，我们也明白了项目并非只是各个简单模块的加和，其任务量远比各个模块的总和要大，也因此掌握了各个模块整合、测试与开发的高效方法，如统一变量名称、统一编码习惯等。相信经过此次数字逻辑项目的开发与学习，我们可以将学习的知识应用于实践，并为未来合作完成大的工程打下坚实的基础。

7. 想法和建议

设计案例：交通信号控制系统

交通信号灯的智能化控制可以大大提高交通效率。利用 Xilinx 开发板可以设计一个基于传感器与智能算法的交通信号控制系统。

首先，红黄绿灯的倒计时可以轻易通过数码管实现，其中最左侧一位数码管可用 R、Y、G 字母来表示红、黄、绿灯。同时，当红灯与黄灯倒计时不足 3 秒时，可改变数码管的显示频率来实现数字闪动的效果，提醒车辆时间不足，保障安全。

不仅如此，此交通信号控制系统可由车流量的变化而改变灯的时间。当一方车流量较大时，可以人为改变拨码开关，使其进入调整时间模式（即该文档中的 advanced_settings 模块），并通过开发板上的按键，适当延长绿灯时间，缩短红灯时间。进一步地，可以设置一个传感器，若有一方长时间没有车辆要经过时，可以长时间的设置为红灯，此时倒计时虽然结束，但不会切换状态，且剩余时间一直为 0，直到传感器显示有车辆来时再切换到绿灯状态。

再进一步，考虑到路口的复杂性，可以适当加入左转、直行、右转等方向，这些方向同样可通过一位数码管实现。在加入方向后，便要优先显示绿灯的剩余时间，但允许人为通过改变拨码开关实现查询功能，此时每个方向的时间都应该能显示。如此，开发板便能得到更高效、多功能的使用。