CS207 Digital Logic project：抽油烟机控制电路(Control circuit of the kitchen exhaust hood)

Group Member：Wai Yan Kyaw，Sean Sovann，莫丰源

Group work distribution（分工）:

Wai Yan Kyaw: Mode switching, Extraction Function, advance settings.

Sean Sovann: Power on/off, Self-Cleaning Function, bonus part, light.

莫丰源: Auxiliary Functions (time, reminder, query).

Groupmate Ratio

Wai: 33.3%

Sean: 33.3%

莫：33.3%

Develop plan schedule & implementation status(开发计划日程安排和实施情况）

week 1: discuss about the choice of project and divdided the work. （done）

week 2: finish time counting part and Power on/off part. (done)

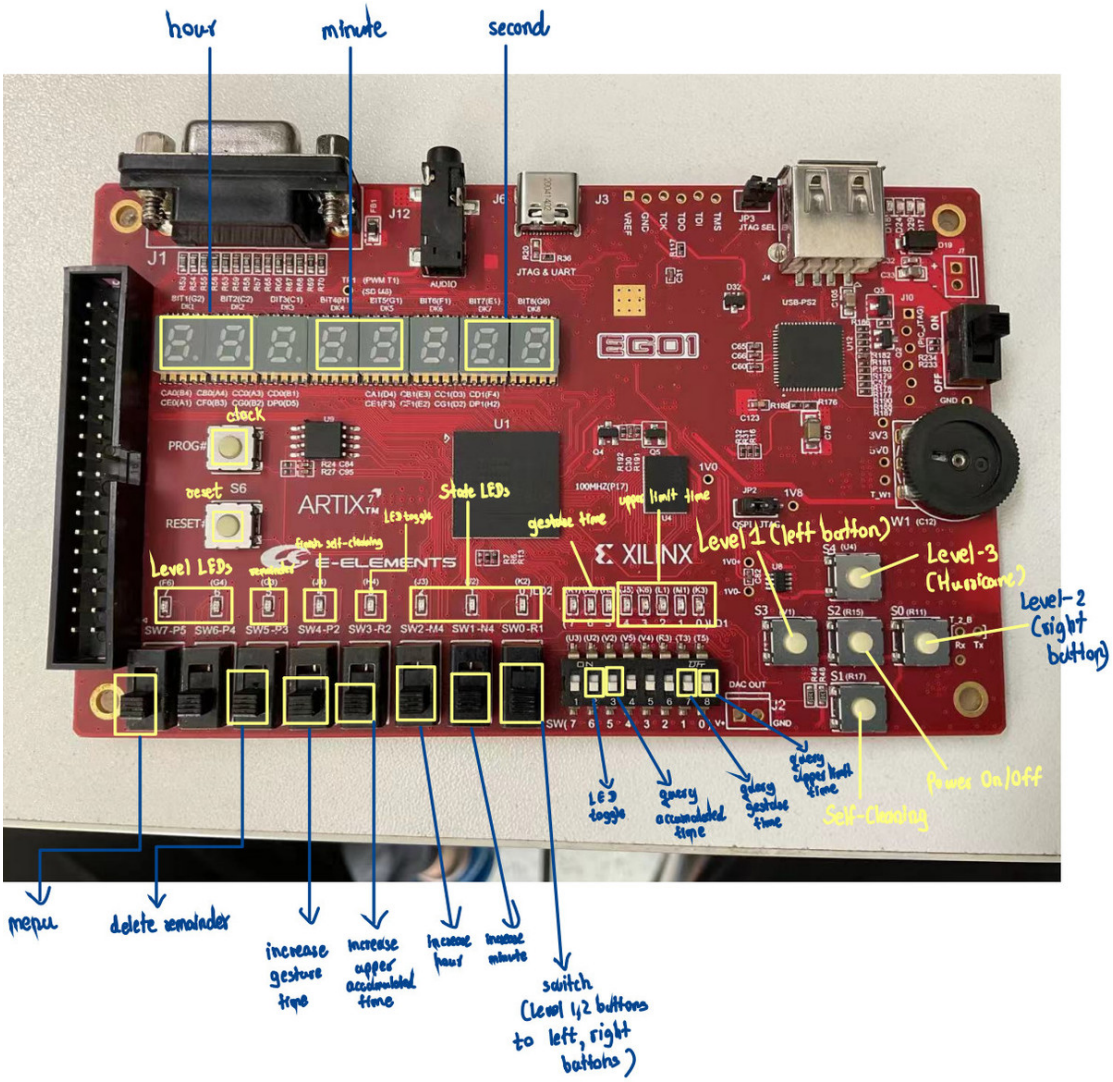
week 3: finish the rest of the part. (done)

week 4: combine together and debug. (done)

System Function List (系统功能列表)

Power on/off （include bonus part)
Mode switching
Extraction Function
Self-Cleaning Function
Auxiliary Functions
Query Function

System usage instructions (系统使用说明)



Input:

sw7-p5: menu button

sw5-p3: delete reminder button

sw4-p2: increase gesture time

sw3-r2: increase upper accumulated time

sw2-m4/sw1-n4: increase hour/minute

sw0-r1: switch for level button or the bonus part button

sw6-u2: led toggle

sw5-v2: query accumulated time

sw1-t3: query gesture time

sw0-t5: query upper limt time

S3: level1 & left button for bonus

S0: level2 & right button for bonus

S2: power on/off button

S1: cleaning mode button

S4: hurricane mode button

Output:

F6, G4: level LEDs

G3: reminder

J4: self cleaning

H4: led toggle

J3, J2, K2: state LEDs

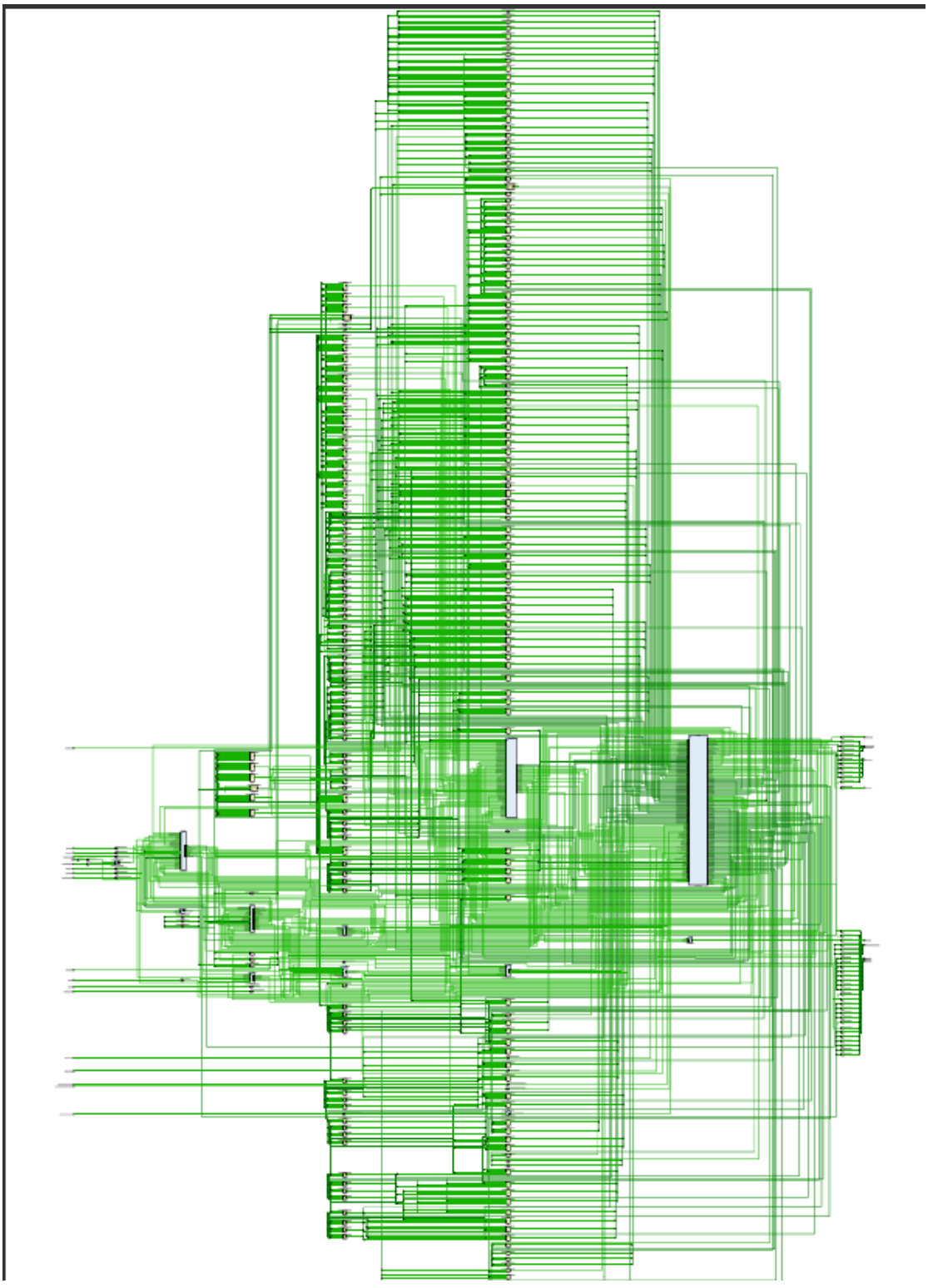
7-segment: time displaying

K1, H6, H5: gesture time

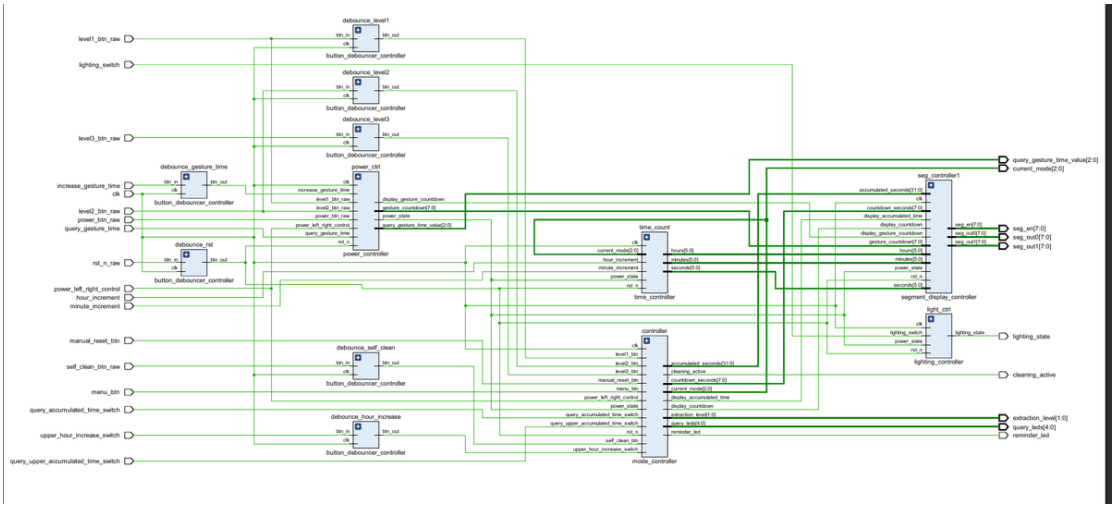
J5, K6, L1, M1, K3: upper limit time

System structure description (系统结构说明)

implementation design



systhesis design



The top_kitchen_hood_controller is the highest of all the module, and under top module, there are lighting_controller, mode_controller, power_controller, segment_display_controller, time_controller. And the button_debouncer_controller is used for debounce of the button.

Sub module function description (子模块功能说明)

Top Module

Input and Output Ports

Input Ports

- **clk**: System clock signal for synchronization.
- **rst_n_raw**: Raw reset signal (active low).
- **power_btn_raw**: Raw input from the power button.
- **menu_btn**: Input for the menu button.
- **level1_btn_raw**: Raw input for level 1 extraction.
- **level2_btn_raw**: Raw input for level 2 extraction.
- **level3_btn_raw**: Raw input for level 3 extraction.
- **self_clean_btn_raw**: Raw input for self-cleaning mode.
- **power_left_right_control**: Control signal for power direction.
- **manual_reset_btn**: Input for a manual reset.
- **hour_increment**: Signal to increment the hour.
- **minute_increment**: Signal to increment the minute.

- **query_upper_accumulated_time_switch**: Switch for querying upper accumulated time.
- **upper_hour_increase_switch**: Switch for increasing upper hours.
- **increase_gesture_time**: Signal to increase gesture time.
- **query_gesture_time**: Signal to query gesture time.
- **lighting_switch**: Control signal for lighting.
- **query_accumulated_time_switch**: Switch for querying accumulated time.

Output Ports

- **query_leds**: Output for LED indicators related to queries.
- **query_gesture_time_value**: Output for the current gesture time value.
- **current_mode**: Indicates the current operational mode.
- **extraction_level**: Indicates the current extraction level.
- **cleaning_active**: Indicates if the self-cleaning mode is active.
- **reminder_led**: Output for the reminder LED.
- **seg_en**: 7-segment display enable signals.
- **seg_out0**: 7-segment display output for the first display.
- **seg_out1**: 7-segment display output for the second display.
- **lighting_state**: Indicates the current state of the lighting.

Top module function explanantion

The top_kitchen_hood_controller module is a comprehensive control unit that integrates various functionalities needed for a kitchen hood system. By managing user inputs, controlling outputs, and coordinating multiple subsystems, it provides a robust solution for related features efficiently. Its design emphasizes stability and responsiveness through debouncing and structured state management.

The sub module contain 7 parts.

A. lighting_controller

Input and Output

Input Ports

1. clk Description: Clock signal used to drive state updates.
2. rst_n Description: Active low reset signal. The system resets when this signal is low.
3. power_state Description: Power state signal indicating whether the system power is on.
4. lighting_switch Description: Lighting switch input signal used to control the lighting state.

Output Ports

1. lighting_state Description: Output signal for the lighting state, indicating whether the lighting is on or off.

Submodule Function Description

This module implements a simple lighting control logic that manages the light's on/off state through clock and reset signals, and determines the final lighting state based on the power state and switch input.

B. mode_controller

Input and output

Input Ports

1. **clk** Description: Clock signal used to drive state updates.
2. **rst_n** Description: Active low reset signal. The system resets when this signal is low.
3. **power_state** Description: Power state signal indicating whether the system power is on.
4. **menu_btn** Description: Input for the menu button.
5. **level1_btn** Description: Input for the level 1 button.
6. **level2_btn** Description: Input for the level 2 button.
7. **level3_btn** Description: Input for the level 3 button.

8. **self_clean_btn** Description: Input for the self-cleaning button.
9. **manual_reset_btn** Description: Input for the manual reset button.
10. **query_upper_accumulated_time_switch** Description: Switch for querying upper accumulated time.
11. **upper_hour_increase_switch** Description: Switch for increasing hours.
12. **power_left_right_control** Description: Input for left/right power control.
13. **query_accumulated_time_switch** Description: Input for querying accumulated time.

Output Ports

1. **accumulated_seconds** Description: Output for accumulated seconds.
2. **display_accumulated_time** Description: Output for displaying accumulated time.
3. **query_leds** Description: Output for query LEDs.
4. **current_mode** Description: Output for the current mode.
5. **extraction_level** Description: Output for extraction level.
6. **countdown_seconds** Description: Output for countdown seconds.
7. **cleaning_active** Description: Output for cleaning active status.
8. **reminder_led** Description: Output for reminder LED.
9. **display_countdown** Description: Output for displaying countdown.

Submodule Functionality Explanation

This module implements a comprehensive control logic for operating modes, extraction levels, and accumulated time tracking, with additional features for debouncing, manual control, and status display management. It ensures smooth transitions between states based on user inputs and power conditions.

C. power_controller

Input and output

Input Ports

1. **clk** Description: Clock signal used for synchronization.
2. **rst_n** Description: Active low reset signal. The system resets when this signal is low.

3. **power_btn_raw** Description: Raw input from the power button.
4. **power_left_right_control** Description: Control signal for left/right power gestures.
5. **level1_btn_raw** Description: Raw input from the level 1 button.
6. **level2_btn_raw** Description: Raw input from the level 2 button.
7. **increase_gesture_time** Description: Signal to increase gesture duration.
8. **query_gesture_time** Description: Signal to query the current gesture time.

Output Ports

1. **power_state** Description: Output indicating the current power state (on/off).
2. **gesture_countdown** Description: Output for the countdown timer for gestures.
3. **display_gesture_countdown** Description: Output for displaying the gesture countdown status.
4. **query_gesture_time_value** Description: Output for the queried gesture time value.

Submodule Functionality Explanation

This module implements a comprehensive power and gesture control system, allowing users to interact with the device through button presses and gestures. It ensures stable operation through debouncing, manages power states effectively, and provides a responsive gesture countdown feature.

D. segment_display_controller

Input and output

Input Ports

1. **clk**: Clock signal for synchronization.
2. **rst_n**: Active low reset signal. The module resets when this signal is low.
3. **hours**: Input representing hours (0-23).
4. **minutes**: Input representing minutes (0-59).
5. **seconds**: Input representing seconds (0-59).
6. **display_countdown**: Control signal to indicate if countdown should be displayed.
7. **countdown_seconds**: Input for countdown seconds.
8. **gesture_countdown**: Input for gesture countdown.

9. **display_gesture_countdown**: Control signal to indicate if gesture countdown should be displayed.
10. **accumulated_seconds**: Input for accumulated seconds.
11. **display_accumulated_time**: Control signal to indicate if accumulated time should be displayed.
12. **power_state**: Signal indicating if the power is on (1) or off (0).

Output Ports

1. **seg_en**: Segment enable signals for controlling which segments are active.
2. **seg_out0**: Segment output for the first display.
3. **seg_out1**: Segment output for the second display.

Module Functionality Explanation

The segment_display_controller is designed to handle multiple display modes, providing flexibility in showing the current time, countdowns, and accumulated time based on user inputs and control signals. It effectively manages the synchronization of display updates through a refresh mechanism, ensuring a smooth user experience.

E.time_controller

Input and output

Input Ports

1.clk: Clock input used for timing and synchronization.

2.rst_n: Active-low reset signal; when low, the module resets its internal states.

3.power_state: Indicates whether the device is powered on (1) or off (0).

4.hour_increment: Signal to increment the hour.minute_increment: Signal to increment the minute**.**

5.current_mode: Represents the current operational mode (e.g., standby mode).

Output Ports

1.hours: Output for hours (0-23).

2.minutes: Output for minutes (0-59).

3.seconds: Output for seconds (0-59).

Module Functionality Explanation

The time_controller module effectively manages the current time and allows for user-controlled increments of hours and minutes. It ensures stable button presses through debouncing, resets on power off, and maintains accurate time through a well-structured counting mechanism. This makes it suitable for use in devices requiring timekeeping functionality.

F. button_debouncer_controller

Input and Output

Input Ports

1. **clk**: Clock input used for synchronization.
2. **btn_in**: Input signal from the button that needs to be debounced.

Output Ports

1. **btn_out**: Debounced output signal that reflects the stable state of the button.

Module Functionality Explanation

The button_debouncer_controller effectively ensures that a button press is accurately registered without the effects of bouncing. By using a counter to track the stability of the button state, it provides a reliable debounced output signal that can be used in various applications where button input is required. This is crucial for preventing erroneous multiple detections of a single button press.

Implementation instructions for bonus (bonus 的实现说明)

This code segment effectively manages gesture-based control in a system, allowing users to increase gesture time and control power based on button presses. The use of states helps to ensure that the system responds appropriately to user inputs, with debounce handling preventing erroneous state changes. The countdown timer adds a time-sensitive element, allowing the system to wait for gestures while providing visual

feedback through a countdown display.

It has been divided into two parts

Part 1 Gesture Function Part

```
if (power_left_right_control) begin
                    // Update countdown timer
                    if (countdown_counter >= ONE_SECOND) begin
                        countdown_counter <= 0;              // Reset countdown counter
                        if (gesture_countdown > 0) begin
                            gesture_countdown <= gesture_countdown - 1; // Decrement
gesture countdown
                        end
                    end
                    else begin
                        countdown_counter <= countdown_counter + 1; // Increment
countdown counter
                    end

                    case (gesture_state)
                        IDLE: begin
                            if (!power_state && level1_btn_debounced && !left_btn_prev)
begin
                                gesture_state <= WAIT_FOR_RIGHT; // Transition to wait
for right gesture
                                gesture_countdown <= current_gesture_time; // Set
countdown
                                display_gesture_countdown <= 1; // Show countdown
                                countdown_counter <= 0; // Reset countdown
                            end
                            else if (power_state && level2_btn_debounced &&
 !right_btn_prev) begin
                                gesture_state <= WAIT_FOR_LEFT; // Transition to wait
for left gesture
                                gesture_countdown <= current_gesture_time; // Set
countdown
                                display_gesture_countdown <= 1; // Show countdown
                                countdown_counter <= 0; // Reset countdown
                            end
                        end
```

```
                    WAIT_FOR_RIGHT: begin
                        if (gesture_countdown == 0) begin
                            gesture_state <= IDLE;        // Back to idle if countdown
is zero
                            display_gesture_countdown <= 0; // Turn off countdown
display
                        end
                        else if (level2_btn_debounced && !right_btn_prev) begin
                            power_state <= 1;             // Turn on power for right
gesture
                            gesture_state <= IDLE;        // Back to idle
                            display_gesture_countdown <= 0; // Turn off countdown
display
                        end
                    end

                    WAIT_FOR_LEFT: begin
                        if (gesture_countdown == 0) begin
                            gesture_state <= IDLE;        // Back to idle if countdown
is zero
                            display_gesture_countdown <= 0; // Turn off countdown
display
                        end
                        else if (level1_btn_debounced && !left_btn_prev) begin
                            power_state <= 0;             // Turn off power for left
gesture
                            gesture_state <= IDLE;        // Back to idle
                            display_gesture_countdown <= 0; // Turn off countdown
display
                        end
                    end

                    default: gesture_state <= IDLE;           // Default to idle state
                endcase
            end
            else begin
                gesture_state <= IDLE;                        // Reset gesture state to idle
```

```
                display_gesture_countdown <= 0;          // Turn off countdown display
            end
        end
    end

endmodule
```

Part 2 Gesture timing part

```
    always @(posedge clk or negedge rst_n) begin
        if (!rst_n) begin
            current_gesture_time <= 3'd5;        // Default gesture time to 5 seconds
            prev_time_increase_stable <= 0;      // Reset previous stable flag
            query_gesture_time_value <= 3'd0;    // Reset query value
        end
        else begin
            prev_time_increase_stable <= time_increase_stable;

            // Handle gesture time increase - only in standby mode
            if (in_standby && time_increase_stable && !prev_time_increase_stable) begin
                if (current_gesture_time >= 3'd7) begin
                    current_gesture_time <= 3'd0; // Wrap around gesture time
                end
                else begin
                    current_gesture_time <= current_gesture_time + 1; // Increment
gesture time
                end
            end

            // Handle query display - only in standby mode
            if (query_gesture_time && in_standby) begin
                query_gesture_time_value <= current_gesture_time; // Set query value
            end
            else begin
                query_gesture_time_value <= 3'd0; // Reset query value
            end
        end
    end
```

Project summary（项目总结）

Effective teamwork and thorough testing were crucial in our project to design a modern kitchen exhaust hood. Our multidisciplinary team of designers, engineers, and developers collaborated closely, holding regular meetings to share insights and align on goals.

During development, we implemented features such as standby, extraction, and self-cleaning modes, ensuring a user-friendly interface and integrating gesture control. Testing validated performance through rigorous assessments and user feedback, allowing us to refine usability and address any issues.

This collaborative effort resulted in a reliable, well-designed kitchen exhaust hood that meets the needs of modern users, highlighting the importance of teamwork in product development.

Through this project, I have gained the experience of coding on hardware, which can help my further development on Signals and Systems. (莫丰源）

After this project, I found fine use of YouTube videos and github can help me develop further in computer science. (Wai)

Working with teamates and overcame a project is a great experience. (Sovann)

Project Proposal （project 出题）

Reference

【FPGA】 Xlinx EGO1 开发板实验设计——简易数字密码锁演示_哔哩哔哩_bilibili

简易电子密码锁（eletronic password lock)

中文版

7 位数码管：显示密码

SW0-SW7：1-8 共 8 个数字，开关是 1 时，对应 1-8。开关是 0 时，对应是 0。

led：开关显示灯，密码错误灯，报警提示灯，重置密码状态灯，密码正确解锁灯，忘记密码状态灯。

按钮: 确认密码按钮，输入密码按钮，重设密码按钮，开关机按钮，忘记密码按钮，恢复出厂设置按钮。

整体实现功能：

用户在打开这个密码锁开关后，可以通过输入密码来解锁。

初始密码设置为 12345678。

当用户开机后，需要按输入密码按钮进入输入状态，然后通过拨弄开关来输入密码，同时用户的尝试会动态显示在 7 位数码管上。当密码输入完以后，用户按下确认密码按钮来核验密码是否正确。如果密码正确，则密码正确解锁灯亮起。如果密码错误，则会亮起密码错误灯，并当第三次及其以上密码输入错误，则报警提示灯会开始闪烁，直到第一次正确输入密码，则密码错误灯和报警提示灯熄灭，而正确解锁灯亮起。

当第一次正确输入密码后，可以进入重设密码的状态。当按下重设密码按钮时，重置密码状态灯亮起。进入重设密码状态后，用户可以手动拨动开关进行密码设置，密码设置完成后，按下确认密码按钮退出重设密码状态并修改初始密码。

当连续三次错误输入密码后，可以进入忘记密码状态。当按下忘记密码按钮时，错误提示灯和报警提示灯会熄灭，然后忘记密码状态灯亮起，进入忘记密码状态后，七段数码管会显示当前的密码，然后再次点击忘记密码按钮来退出忘记密码状态，此时忘记密码状态灯熄灭。

当恢复出厂设置按钮被按下，恢复初始密码。

值得注意的是，在关机状态时，一切操作无效。

加分项设计：

①可以外接蜂鸣器，当报警提示灯闪烁时可以用蜂鸣器进行报警。

②利用外接设备（电脑端）输入输出密码

③可以添加输入密码的条件 ,比如第三次及以上后每次输入错误密码后会进入锁定状态，7 位数码管会显示 30s 的倒计时，在此期间无法对开发板进行任何有效操作，倒计时结束后退出锁定状态。

English Version

7-Segment Display: Show Password

SW0-SW7: 8 switches corresponding to numbers 1-8; when a switch is set to 1, it corresponds to 1-8. When a switch is set to 0, it corresponds to 0.

LEDs: Switch display light, password error light, alarm indicator light, reset password status light, password correct unlock light, forget password status light.

Buttons: Confirm password button, input password button, reset password button,

power on/off button, forget password button, restore factory settings button.

Overall Functionality:

When the user turns on the password lock, they can unlock it by entering a password. The initial password is set to 12345678.

After powering on, the user must press the input password button to enter input mode, then toggle the switches to enter the password. The user's attempts will be dynamically displayed on the 7-segment display. After entering the password, the user presses the confirm password button to verify if the password is correct. If the password is correct, the password correct unlock light will illuminate. If the password is incorrect, the password error light will turn on. After three or more incorrect password attempts, the alarm indicator light will start flashing until the first correct password is entered, at which point the password error light and alarm indicator light will go out, and the correct unlock light will illuminate.

After the first correct password entry, the user can enter the reset password state. When the reset password button is pressed, the reset password status light will turn on. In the reset password state, the user can manually toggle the switches to set a new password. After setting the password, pressing the confirm password button will exit the reset password state and update the initial password.

After three consecutive incorrect password entries, the user can enter the forget password state. When the forget password button is pressed, the error indicator light and alarm indicator light will go out, and the forget password status light will illuminate. In this state, the 7-segment display will show the current password. Pressing the forget password button again will exit the forget password state, turning off the forget password status light.

When the restore factory settings button is pressed, the initial password will be restored.

It is important to note that in the powered-off state, all operations are invalid.

Bonus Part:

1. An external buzzer can be connected to sound an alarm when the alarm indicator light is flashing.
2. Allow password input/output via an external device (computer).
3. Add conditions for password input, such as entering a lock state after three incorrect password attempts, during which the 7-segment display will show a

30-second countdown. During this time, no effective operations can be performed on the development board, and the lock state will exit after the countdown ends.