LCD Module Technical Documentation

Core Implementation Details

1. LCD Hardware Interface

The LCD module uses I2C communication through the board library and Adafruit's character LCD library. The display is initialized as a 16x2 character LCD with RGB backlight support. The hardware interface requires:board.I2C() for I2C communication

Character_LCD_RGB_I2C class for display control

16x2 character display configuration

Built-in button support (up, down, left, right, select)

Default contrast setting of 40 (range 0-255)

2. State Management

The module maintains two critical global states:manual_override dictionary:

Tracks override states for light, fan, and watering

Prevents automatic control conflicts

Resets on system restart

watering_active flag:

Critical for preventing concurrent watering operations

Thread-safe state tracking

Used for safety interlocks

3. Critical Functions

Button Debouncing

The debounce function implements hardware button debouncing with a 100ms debounce time. It takes a button state function as input and returns the stable button state. This is crucial for preventing false triggers and ensuring reliable button operation.

Menu Display System

The display_menu function handles:Text display with scrolling for long options

Current selection visibility

Display refresh timing

Two-line display management

4. Threading Implementation

The module uses three main threaded operations:1. Watering Control Thread:Non-blocking water control

Sets global watering_active flag

Implements safety timeouts

Handles state cleanup

Fan Control Thread:

Non-blocking fan operation

Manages fan duration

Handles state cleanup

Camera Control Thread:

Non-blocking picture capture

Status display updates

Error handling

5. Hardware Control Integration

Light Control

The control_light function:Manages grow light state

Updates manual override status

Handles errors with visual feedback

Implements status color coding

Returns to normal state after operation

Watering Control

The control_watering function:Manages watering system safety

Updates watering_active flag

Handles manual override states

Implements safety timeouts

Provides visual feedback

Integration Requirements

1. Required Backend Functions

backend must provide:Configuration Functions:get_plant_settings(): Returns dictionary of plant settings

update_config(section, key, value): Updates configuration file

Hardware Control Functions:growlighton(): Returns boolean success

growlightoff(): Returns boolean success

fanon(duration): Returns boolean success

fanoff(): Returns boolean success

autorain(volume): Returns integer (1 for success)

stopwater(): Returns boolean success

picam_capture(): Returns boolean success

2. Error Handling Integration

Expected error types from backend:Hardware access errors

Configuration errors

Timing/scheduling errors

3. Configuration Integration

The module interacts with these configuration parameters:PLANTCFG section:maxTemp

maxHumid

dryValue

waterVol

sunrise

sunset

checkTime

PICAMERA section:CameraSet

Maintenance Guidelines

1. Critical Areas to Monitor

Thread Management:Check for thread leaks in watering_active state

Monitor manual_override states

Verify thread cleanup

Implement timeout mechanisms

Button Debouncing:Monitor for false triggers

Adjust debounce time if needed (currently 100ms)

Verify button hardware functionality

Display Updates:Menu refresh rate (0.5s)

Time display update (1s)

Scroll speed (0.3s per character)

Contrast settings

2. Common Issues and Solutions

Display Freezing:Clear display

Reset color to green

Reset contrast to 40

Verify I2C connection

Button Malfunction:Test raw button states

Verify debounce function

Check button hardware

Monitor for stuck states

Threading Issues:Check active threads

Reset watering state

Clear manual overrides

Implement timeouts