Technical Module Description: backlighting.py

Version: 1.0 Date: January 17, 2025

Authors: Henk Stevens, Olaf Mastenbroek, Onno Janssen

Organization: Stichting Oradio

License: GNU GPL

# Module Purpose

This module provides an automated backlight control system for the Oradio device, designed to operate as a persistent background service. It uses a TSL2591 digital light sensor to measure ambient light levels and dynamically adjusts an LED backlight via an MCP4725 Digital-to-Analog Converter (DAC). The backlight level is smoothed over time and minimal brightness settings persist across reboots using EEPROM.

# Main Features

* Light Sensing: Reads visible light data from the TSL2591 sensor via I²C.
* Brightness Adjustment: Converts light levels into DAC values for backlight dimming.
* Smooth Transitions: Gradual adjustment using 30-step interpolation.
* Startup Configuration: Initializes with backlight off (DAC set to 4095).
* Persistent Storage: Writes DAC state to EEPROM for reboot persistence.
* Service-Ready: Designed to run as a systemd or background service.

# Key Components and Flow

## Initialization (\_\_init\_\_)

Sets addresses for I²C devices (TSL2591 at 0x29, MCP4725 at 0x60). Defines brightness ranges and sensor configuration. Writes maximum DAC value (4095) to EEPROM to ensure LEDs are off on startup.

## Sensor Setup (initialize\_sensor)

Enables TSL2591 with ALS and power-on flags. Sets gain and integration time to medium/300ms.

## Measurement & Lux Conversion

read\_visible\_light() fetches raw 16-bit light data from sensor. calculate\_lux() estimates ambient lux using fixed gain/integration scaling.

## Brightness Control

interpolate\_backlight(lux) maps lux levels to a DAC value using interpolation: Below threshold → backlight off (4095); Mid range → scaled dimming; Bright → brighter DAC setting. The DAC controls the 5 independent current sources. So all the leds are controlled at one level, but still independent to control via the GPIO’s to switch on and off. And off is than to the common level of the backlighting level.

## Main Loop (auto\_adjust)

Continuously reads light levels every 0.5 seconds. If a significant change is detected (>30% threshold), it recalculates a new DAC target and gradually transitions brightness using 30 steps.

## Manual Overrides

off() stops the loop and switches off the backlight. maximum() stops the loop and sets backlight to maximum brightness level.

## EEPROM Control

write\_dac\_to\_eeprom() writes a DAC value to non-volatile memory for persistence after reboots.

# Usage Example

To launch as a standalone service:

python3 backlighting.py

# Testing Script: test\_backlighting.py

The script `test\_backlighting.py` is designed to test and manually operate the backlighting system defined in `backlighting.py`. It provides a command-line interface to control and inspect the backlight behavior, allowing developers to verify the system’s response to light levels and user commands during development or diagnostics.

## Main Capabilities:

* Manual control of the auto\_adjust() loop via interactive menu.
* Start and stop the dynamic backlight adjustment in a separate thread.
* Direct control options to turn the backlight off or set it to maximum.
* Sensor test mode prints raw light readings, calculated lux, and DAC output every 2 seconds.

python3 test\_backlighting.py