Documentation

Battery-powered electronic relay switching device

Anton Labada

March 8, 2025

Contents

1 Introduction

This project implements a low-power STM32-based system designed for relay control, LED blinking, and battery monitoring, with an efficient deep sleep mode to conserve power (simulating power-off conditions). The system is powered by a 3.7V Li-ion battery, making it ideal for portable or remote applications requiring minimal power consumption.
Project goals:

- Efficient power management to extend battery life.
- User-friendly control with a single button for multiple functions.
- Reliable low-battery indication to prevent unexpected shutdowns.

2 Hardware Description

The system is built around the STM32L031K6 microcontroller, chosen for its low power consumption and built-in ADC for battery monitoring.
Microcontroller unit:

- STM32L031K6: Ultra-low-power 32-bit ARM Cortex-M0+ MCU.
- Operates at 3.3V, making it compatible with the battery power system.
- Provides GPIOs for LED control, button input, and ADC-based battery monitoring.

Power supply:

- 3.7V Li-ion battery (18650) as the main power source.
- TPS63060 buck-boost converter regulates voltage to 3.3V for the MCU.
- Step-up converter (MT3608) converts 3.3V to 5V to power the relay module.
- Voltage divider scales battery voltage for ADC monitoring.

Relay control:

- 5V single-channel relay module (SRD-05VDC-SL-C) for switching high-voltage loads. Operates with a maximum current rating of 10A at 250VAC or 30VDC. The module includes two LEDs: one for power status and another indicating the relay state (open or closed).

3 Firmware Description

STM32CubeIDE was used for its compatibility with STM32 microcontrollers, providing tools for code generation, debugging, and peripheral configuration. The STM32 HAL library was also utilized, as it simplifies peripheral initialization and configuration while ensuring portability across different STM32 microcontroller families.
The C language was chosen due to its low-level control over hardware, which is essential in embedded systems development. It allows direct access to MCU registers, efficient memory usage, and fast execution times, which are critical for real-time applications and low-power operations such as standby and wake-up modes.
The primary file where the firmware logic is implemented is `main.c`. The remaining files configure peripherals and are generated automatically from the `.ioc` file.
Function descriptions:

- `battery_measure()` – Reads the ADC value from the battery and converts it into the actual battery voltage using a voltage divider ratio. Returns the battery voltage for monitoring purposes.
- `enterDeepSleep()` – Puts the STM32L031 into low-power standby mode for minimal power consumption. The function enables a wake-up pin to resume operations.
- `main()` – Handles button presses by monitoring the GPIO pin for short presses (to trigger standby mode) or long presses (to change the relay switching frequency). Generates control signals (GPIO output) for the relay module. The status LED on the module turns on when the signal is high and off when it is low. Implements battery monitoring using the ADC, triggering the built-in green LED as a low-battery indicator when the voltage drops below a certain threshold (3.1V).

The code is thoroughly commented to provide clear insights into its functionality. Each function, variable, and algorithm is documented to explain its purpose.

4 Conclusion

The project was thoroughly tested on a physical STM32F411 development board, and all functions performed as expected. The relay switching, LED control, battery monitoring, and power management features operated correctly, demonstrating the system's reliability and efficiency in real-world conditions.

However, some challenges were encountered with the button functionality. In certain cases, the system fails to detect a long button press, entering standby mode immediately and preventing relay frequency adjustments. This issue is currently under investigation, with potential solutions including software-based debounce improvements or the implementation of separate buttons for different tasks.

Overall, the project successfully met its objectives, providing an efficient and low-power relay switching solution. Future improvements could focus on refining button responsiveness and optimizing firmware for even lower power consumption.