**Design Essay: Drone RF Signal Detector**

**Introduction**

The conflict in Ukraine, especially the growing use of drones in combat, inspired this project. Videos have circulated online showing soldiers in high-stress situations trying to avoid FPV drones equipped with explosives. In one such video, a soldier was shown hiding under cover while holding a handheld device with a single antenna. The device was emitting alarms, warning him of an incoming drone. That moment sparked an idea: could I design a similar tool—something simple, effective, and potentially life-saving?

The goal of this project was to develop a rugged, portable, and affordable device that could detect the presence of drones by passively scanning their common control and video transmission frequencies. Many commercial solutions exist, but most are expensive and designed for professional or military use. I wanted to create something more accessible—something that could be built with less than $100 worth of parts and provide early warning in both civilian and tactical environments.

To keep it field-capable, I housed the system in a waterproof Pelican-style case and designed it to use simple user interfaces—an LCD screen and a buzzer—both of which can be muted or dimmed to avoid drawing attention. My goal wasn't to build the final solution to drone detection but to develop a working prototype and learn as much as possible about RF detection, embedded systems, and electronic shielding.

**Final Project Requirements**

* I2C screen
* PWM active buzzer
* SPI serial communication with cc2500
* ADC Logarithmic detection
* External interrupt for testing cc2500

**System Description**

The system passively detects RF signals across four major frequency bands:

* **433 MHz** (often used by DIY drones and telemetry modules)
* **915 MHz** (used by LoRa and some long-range control systems)
* **2.4 GHz** (common for consumer drones like DJI)
* **5.8 GHz** (used by FPV video transmitters)

Each band is monitored by an analog logarithmic RF detector (the AD8317), with a dedicated antenna tuned to that frequency. Originally, I attempted to use digital transceiver modules like the CC2500 for 2.4 GHz and other UART-based RF chips for the lower bands. However, I later found them unreliable or difficult to configure due to poor documentation or manufacturing quality. As a result, I transitioned entirely to using AD8317 detectors across all bands for consistent analog output and simpler interfacing with the microcontroller.

The microcontroller chosen for this project was the STM32L4 Nucleo board, selected for its low power consumption and extensive peripheral support. It reads the analog voltage output from each AD8317 using the onboard ADC and compares it to a calibrated noise floor. If a signal exceeds the threshold, the device triggers an alert: a tone is played on a passive buzzer and a message is displayed on a 16x2 I2C LCD.

**Hardware Overview**

* **Microcontroller**: STM32L4 Nucleo
* **Detectors**: Four AD8317 analog RF detectors (one per band)
* **User Feedback**: QAPASS I2C LCD display and TMB12A05 active buzzer
* **Power Supply**: 3.3V regulator powered by USB or battery pack, 9v batteries
* **Shielding**: Copper-lined case and shielded cables to reduce interference

To prevent EMI issues, I used copper tape to line the inside of the enclosure and created individual Faraday cages around each detector module using 3D-printed shells. All shielding was grounded to the system's common ground.

**Software Summary**

The firmware was developed using STM32CubeIDE and written in C. It uses the ADC to continuously sample voltage from each detector, compares it to a moving average-based noise floor, and triggers alerts when thresholds are exceeded. The software is modular, with separate drivers for the buzzer, LCD, and (initially) the CC2500.

A button on the device (wired to pin PC13) can be used to trigger manual tests or recalibration. I also included an interrupt-based input to support features like “silent mode” or “dark mode” in future revisions.

**Testing Progress**

Testing is organized into four phases:

**Phase 1** has been completed. In this phase, I conducted initial testing in urban environments (such as campus buildings) to establish what typical RF noise looks like across the four bands. Wi-Fi routers, Bluetooth devices, and other transmitters produced measurable signals in the 2.4 GHz and 5.8 GHz ranges. I also verified that shielding and cabling were adequate, and resolved early interference issues caused by the LCD’s unshielded I2C cable.

**Phases 2, 3, and 4** are still pending. The plan for these phases is as follows:

* **Phase 2:** Testing in a low-noise environment, such as a cave or remote area, to confirm that the device registers minimal interference in isolation.
* **Phase 3:** Introducing known interference sources (phones, laptops, drones) in the same low-noise environment to evaluate how well the detector responds to each.
* **Phase 4:** Real-world field tests in city and mountain environments, using a variety of drones. I plan to collaborate with the drone society at my university to gain access to more test platforms.

**Challenges Faced**

Throughout the project, I encountered several technical obstacles:

* The CC2500 module initially used for 2.4 GHz failed after limited use due to build quality.
* Finding reliable datasheets for low-cost RF modules was difficult, which led to delays and dead ends.
* The I2C LCD cable introduced noise into the system until I added copper shielding to it.
* Creating proper shielding and grounding was essential for reliable analog readings.
* Debugging embedded systems without commercial tools was time-consuming but ultimately rewarding.
* Soldering of delicate circuts

**Future Improvements**

This prototype functions as a proof of concept, but there’s significant room for expansion. My next steps include:

* Completing Phases 2–4 of testing
* Fully replacing the CC2500 with an AD8317 for 2.4 GHz detection
* Adding buttons for toggling silent and dark modes
* Powering the unit with a dedicated battery pack
* Refactoring code for cleaner modular design
* Exploring software-defined radio (SDR) for deeper signal classification in future versions

**Career Reflection**

I started this project knowing it would push me beyond my comfort zone. Over the course of several months, I gained hands-on experience in embedded systems design, RF signal processing, circuit shielding, analog interfacing, and C programming. I also learned how to troubleshoot obscure hardware problems, work with multiple communication protocols (SPI, I2C, ADC, PWM), and write modular, maintainable firmware.

At the end of this phase of the project, I have a functional prototype that performs initial detection reliably in high-noise areas. The system is not yet field-tested against a wide range of drones, but I now have the skills and structure needed to pursue that next step. Most importantly, this project has built a strong technical foundation for future work in RF systems and embedded engineering.