RFx – An Open Source Radio Frequency Discovery Device with Integrated AI Processing

Lewis Birch
School of Computing & Communications
Lancaster University
Lancaster, UK
l.birch@lancaster.ac.uk

***Abstract*—RFx is an open-source, low-cost toolkit for passive RF monitoring with on-device TinyML. A modular RF front-end and microcontroller run pretrained customer models to detect and classify signals in real time, untethered from cloud or host PCs. Designed for rapid prototyping, RFx lets users explore edge AI, spectrum sensing and RF security projects.**

I. Introduction / Background

Wireless technologies are increasingly prevalent resulting in a crowded and complex radio frequency (RF) environment. With applications ranging from consumer electronics, industrial IoT, national security and privacy auditing depending on the ability to detect, monitor and interpret RF signals. Despite this, passive RF monitoring larger remains inaccessible to developers, students and researchers due to the high cost, complexity or closed source nature of existing tools.

Pre-existing solutions such as spectrum analysers and software defined radios (SDRs) are often bulky with limited mobility, power intensive and require cloud-based processing for advanced analytics. These limitations result in barriers to rapid prototyping and field deployment, especially for real-time use cases. Such as drone detection, anomaly detection, or wireless intrusion monitoring. While embedded machine learning (ML) has seen rapid growth, it has not yet been widely used in edge-based RF in a portable and developer friendly form.

RFx addresses this shortfalls by proposing a lightweight, open-source toolkit for passive RF monitoring with on-device ML. Designed and built around a low-cost RF front-end and single-board computer hardware, RFx supports ML pipelines that can classify or detect events (e.g., Wi-Fi beacons, drone telemetry, unlicensed transmissions) directly on device, without the need for cloud connectivity. It is designed to be simple, extensible and modular. Ideal for prototyping new RF based applications in resource constrained environments.

This proposed project aligns closely with the goals of the Device Prototyping Summer school, enabling hands on exploration of embedded systems, signal processing and hardware/software design. RFx gives users the ability to build, test and iterate on custom RF sensing solution by combining SDR style flexibility with embedded AI capability. The platform fosters learning in areas such as low power on device ML, antenna design, RF fingerprinting and embedded software development, while encouraging open collaboration and experimentation.

II. Related Work

Passive RF signal monitoring and classification has been explored in academic research and open-source communities. SDRs such as GNU Radio, HackRF, and RTL-SDR [1] [2] [3] have enabled flexible real-time experimentation. However, these typically require powerful and expensive hardware and are not optimised for embedded or mobile applications. Previous work in drone detection, wireless intrusion detection and spectrum monitoring has demonstrated the utility of RF classification, but solutions are often tied to expensive hardware and lack modularity for rapid prototyping.

ML approaches have been applied to RF signal classification, often using spectrograms as inputs to convolutional neural networks (CNNs) or recurrent architectures. Projects like Radio Frequency Fingerprinting via Deep Learning: Challenges and Opportunities [4] have shown promise on device finger printing and signal identification tasks. However, these methods typically require cloud-based or use of server-based GPU hardware for inference, limiting real-time on-device applicability.

Advancements in efficient low-compute ML models, such as the development of Tiny ML [5], enables deep learning inference on low power devices and edge hardware with limited resources. Frameworks like Lite RT [6] and Edge Impulse [7] have made it easier to deploy ML models for audio, vision and sensor data processing. Whilst time-series and frequency domain signal are common in these applications, RF data has received comparatively little attention, despite being a strong candidate for low-power, passive ML systems.

Training robust models typically demands large, varied dataset, but real-world RF captures are hard to gather without access to pre-existing hardware (hence this project). Synthetic RF datasets generating procedurally generated waveforms or simulated RF scenes have proven effective at augmenting scare labelled datasets [8]. Making synthetic RF datasets an ideal companion to RFx: which can then bootstrap model development until the device ultimately closes the loop by collecting real signals in the field.

III. Imagined or existing prototype sketches/drawings/photos

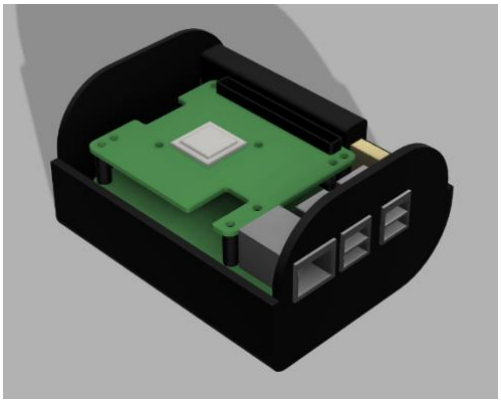

Figure 1 - RFx with Top Half Shell Removed

My experience with integrated electronic design and manufacturing is limited (something I'm hoping to start to remedy through this summer school). Therefore, what I've proposed below is just the form of the device that I can currently understand from parts I've worked with before. I expect with the use of different embedded components, etc. that are optimised for this use case can considerably reduce the size and/or cost of this device.

A. Compute & ML Subsystem

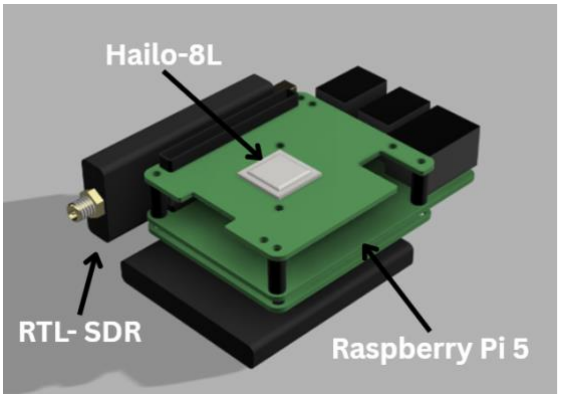

Figure 2 - RFx Compute and RF Capture Components

A Raspberry Pi 5 serves as the main host. It's 2.4GHz quad-core CPU and abundant I/O will give RFx enough headroom to capture, process and buffer RF streams while still talking to external sensors and user-interface peripherals.

Real-time inference will be off-loaded to a Hailo-8L (~ 13 TOPS @ 3W) mounted on the Raspberry Pi M.2 AIHAT+, which interfaces into the Pi 5's native PCIe slot. The Pi's OS automatically configures and uses the Hailo device to easily run ML workloads without the user having to deal with driver code.

To ensure that the Hailo-8L can run without thermal throttling when under load, the Raspberry Pi active cooler will be required. This plugs directly into the Pi's dedicated fan connection.

B. RF Front-end

To convert RF into a digital signal suitable for real-time analysis an RTL-SDR Blog V4 USB dongle will be attached to one of the Pi's USB ports. The V3 covers ~0.5MHz to 1.7GHz, giving RFx access to the most common Global Navigation Satellite System (GNSS), ISM radio bands (Bluetooth, NFC, Wi-Fi, etc.) and drone-control bands. It features a SMA female jack that allows the user choice of band-specific antennas depending on what range of RF signal they wish to capture.

Optionally, for use cases with weak-signals or crowded-band scenarios a low-noise amplifier and a saw band-pass filter could be added without any extra power overhead.

This setup maintains a standard SDR interface, allowing users to swap in higher-performing receivers if required without changing the hardware stack.

C. Power & Charging Subsystem

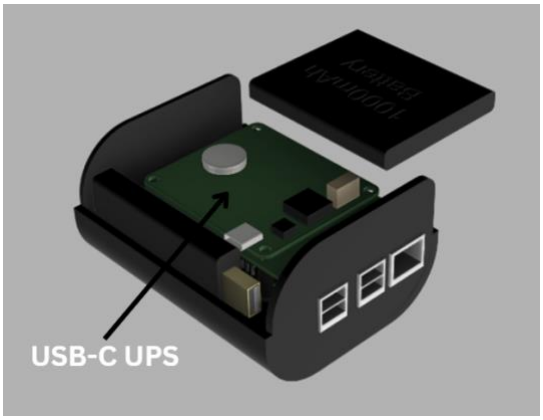

Figure 3 - RFx Power & Charging Components

The RFx device needs to be able to run for several hours on battery only, be easily chargeable and survive abrupt power loss without data loss. This would come in the form of a PiSugar 3 PRO UPS.

This would provide continuous power to the Pi and ensures that the Pi doesn't lose power whilst switching between mains and battery power. It has a USB-C port that practically any modern laptop charger can use or if in the field could be plugged into a larger portable power bank.

The UPS should provide 5,000mAh, which would provide ~ 1.5 hours of continuous RF signal capture + inference (assuming ~5 W Pi5 + ~3W Hailo-8L + 1W RTL-SDR). Which could then be boosted with any off the shelf power bank to at least double the devices usage time (assuming 10,00mAh portable power bank).

D. Enclosure

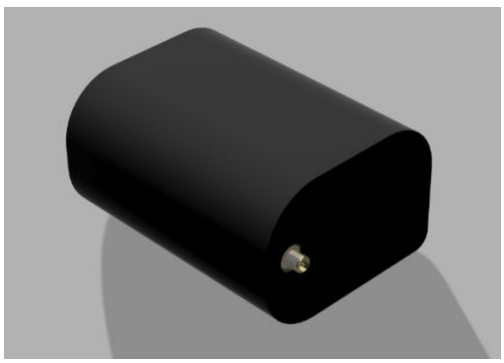

Figure 4 – RFx with Full Enclosure and External SMA Port

The housing is a two-piece 3D print (either PETG or ASA) chosen for its impact resistance, UV resistance and low moisture content. Combined with a neoprene gasket is

compressed between lid and base to provide a reasonable level of water proofing. All threaded joins will use brass heat-set inserts melted into boss pillars, allowing all components to be mounted directly to standoffs.

IV. Responsible Innovation

RFx is grounded in responsible innovation; the entire hardware stack, firmware, and ML models are open source. The device is receive-only, providing no transmit function that could be used for nefarious purposes.

The build prioritises low-power silicon (< 10 W), recyclable casing, and modular, screw-secured boards so worn parts can be swapped rather than discarded, keeping both energy use and e-waste low while delivering clear social value through improved access to RF analysis and safety.

V. Author Bio(s) / Experiences

I was drawn to RF-based drone detection out of both curiosity and concern over the growing use of drones in sensitive environments. While exploring passive RF detection, I found a lack of open-source, affordable tools and this then led to the idea of RFx.

My background includes a MSci in Computer Science and a BTEC extended diploma in mechanical engineering, with hands-on skills in CAD/CAM, and 3D printing. I've also worked on synthetic dataset generation for sensitive applications, and contributed to the training, fine-tuning, and optimization of AI models [9]. I've also published multiple pieces of work in the AI security space during my PhD [10] [11], as well as being involved in a startup span out of Lancaster University.

VI. References

[1] "GNU Radio - The Free & Open Source Radio Ecosystem · GNU Radio," *GNU Radio*. https://www.gnuradio.org/

[2] "HackRF One - Great Scott Gadgets," *greatscottgadgets.com*. https://greatscottgadgets.com/hackrf/one/

[3] admin, *Rtl-sdr.com*, Nov. 19, 2019. https://www.rtl-sdr.com/

[4] S. Al-Hazbi, A. Hussain, S. Sciancalepore, G. Oligeri, and P. Papadimitratos, "Radio Frequency Fingerprinting via Deep Learning: Challenges and Opportunities," *arXiv.org*, 2023. https://arxiv.org/abs/2310.16406 (accessed May 14, 2025).

[5] J. Lin, L. Zhu, W.-M. Chen, Wei Chen Wang, and S. Han, "Tiny Machine Learning: Progress and Futures [Feature]," *IEEE Circuits and Systems Magazine*, vol. 23, no. 3, pp. 8–34, Jan. 2023, doi: https://doi.org/10.1109/mcas.2023.3302182.

[6] Google, "LiteRT overview," *Google AI for Developers*, 2024. https://ai.google.dev/edge/litert

[7] "Edge Impulse," *edgeimpulse.com*. https://edgeimpulse.com/

[8] Y. Kaasaragadda and S. Kokalj-Filipovic, "ReFormer: Generating Radio Fakes for Data Augmentation," *arXiv.org*, 2024. https://arxiv.org/abs/2501.00282

[9] S. Trawicki, W. Hackett, L. Birch, N. Suri, and P. Garraghan, "Compilation as a Defense: Enhancing DL Model Attack Robustness via Tensor Optimization," *arXiv.org*, 2023. https://arxiv.org/abs/2309.16577 (accessed May 14, 2025).

[10] L. Birch, W. Hackett, S. Trawicki, N. Suri, and P. Garraghan, "Model Leeching: An Extraction Attack Targeting LLMs," *arXiv.org*, 2023. https://arxiv.org/abs/2309.10544 (accessed May 14, 2025).

[11] W. Hackett, L. Birch, S. Trawicki, N. Suri, and P. Garraghan, "Bypassing Prompt Injection and Jailbreak Detection in LLM Guardrails," *arXiv.org*, 2025. https://arxiv.org/abs/2504.11168 (accessed Apr. 22, 2025).