Svitlana Orlova

Embedded | Linux Developer | IRL Stamp 4

Contact Information

- Mobile: +353(85)127-69-81
- Email: svitlana.orlova@outlook.com
- Linkedin: www.linkedin.com/in/svitlana-orlova
- More: svitlana-orlova.github.io

Summary

Experienced Embedded Systems Engineer with 20+ years in both research and commercial environments across startups and large enterprises. Adept at designing and debugging low-level software for microcontrollers and embedded platforms, with a strong emphasis on safety, performance, and cross-platform reliability.

- Expert in C/C++ for microcontroller development (ESP32, STM32..)
- Deep knowledge of Linux systems and kernel-level programming
- Precision-driven in microcontroller initialization and memory-safe coding
- Passionate about reverse engineering and protocol analysis
- Hands-on experience with electronics design and troubleshooting

Technical & Collaboration Skills

- Documentation & Auditing:
  Strong emphasis on traceable, auditable processes; comfortable with Markdown, Doxygen, and internal wikis
- Requirements Engineering:
  Skilled at translating hardware specs and system constraints into clear, testable software requirements
- Version Control:
  Proficient in Git (CLI and GUI), branching strategies, and collaborative workflows (GitHub, GitLab)
- Scripting & Automation:
  Experienced with shell scripting, JavaScript, and Python to build automation tools, streamline workflows, and support system integration
- Language Proficiency:
  Native Ukrainian speaker with strong professional English communication skills

Experience

- 2024-2025 Embedded Software Engineer
  Developed firmware for STM32F4 microcontrollers. Built desktop applications using Qt for embedded system configuration and diagnostics
- 2022-2024 Linux Developer
  Engineered Linux kernel drivers for video cameras, primarily for NVIDIA Jetson platform. Designed and deployed cross-platform video streaming software (Linux, Windows, macOS)
  Programmed microcontrollers including STM32F4 and ESP32, integrating them with host systems
- 2020 Embedded Software Engineer
  Created modular firmware components for ESP32, optimizing for memory footprint and reliability
- 2010-2012 Firmware Developer
  Developed firmware for AVR microcontrollers, including low-level register manipulation and timing-critical routines

- 2006-2008 C Developer
  Built software for data parsing, database interaction, and web crawlers, using C and shell scripting
  Focused on performance tuning and robust error handling in resource-constrained environments
- 2003-2005 Software Developer
  Maintained and extended software for hardware systems communicating via serial ports and Ethernet
  Managed Unix systems while developing custom tools for device control and monitoring
  This role marked the beginning of a long-term focus on embedded systems and low-level programming

Selected Projects

- 2021-2022 Cross-Platform UI Toolkit
  Designed and implemented a custom UI toolkit supporting RPi, Android, Linux, Windows, and WebAssembly, featuring a built-in markup language, network stack, storage layer, touchscreen input, and 3D rendering
  Originally conceived as a lightweight C++ game development framework, evolved into a full-featured cross-platform UI engine
- 2023 WiFi-UART Protocol Firmware for ESP32
  Developed production-grade firmware for ESP32, implementing a custom WiFi-UART protocol for device communication.
  Built a complete front-end UI using a self-designed framework with rich user interaction features
  Technologies: C99, JavaScript, custom UI engine
- 2022 EV Charger Controller (ESP32)
  Engineered firmware and UI for an ESP32-based EV charger, integrating safety mechanisms, scheduling, user authorization, and usage accounting
  Developed both backend logic and frontend interface using C99 and a custom JavaScript framework
  Focused on reliability, user experience, and secure access control
- 2021 Linux MSR Manipulation Module
  Created a Linux kernel module to manipulate Model-Specific Registers (MSR) on Intel x86_64 systems
  Exposed a character device with a minimal command-line interface for safe user-space interaction
  Technologies: C90, Linux kernel APIs, low-level register access