SIDRAM

Linux Device Driver Engineer — Embedded Engineer

Bangalore, India

+91-9148475086 — sidramhiremath6@gmail.com

LinkedIn

Professional Summary

Embedded Systems Engineer with 1.8 years of industry experience in Linux Device Driver development, Embedded Linux, and Linux System Programming. Hands-on experience developing Linux character and platform drivers, GPIO drivers, Device Tree integration, kernel synchronization mechanisms, and sysfs interfaces for ARM-based embedded platforms. Proficient in C programming, POSIX system programming, socket programming, and low-level protocol implementation for industrial communication systems. Experienced in BSP bring-up, Linux kernel configuration, U-Boot customization, OpenWrt, Buildroot, cross-compilation, and embedded system integration. Strong understanding of Linux kernel internals, memory management, synchronization primitives, and hardware-software interaction with a passion for developing reliable, high-performance Linux kernel solutions.

Technical Skills

- **Programming Languages:** C, C++, Embedded C, Python, Shell Scripting
- **Linux Kernel Development:** Linux Kernel Modules, Character Drivers, Platform Drivers, GPIO Descriptor API (gpiod), Sysfs, Device Tree (DTS/DTBO), Module Lifecycle, Kernel APIs, Kernel Build System
- **Kernel Internals:** Memory Management, devm_kzalloc(), krealloc(), Mutexes, Spinlocks, Atomic Operations, Interrupt Handling, Circular Buffers, Kernel Synchronization, Kernel/User Space Interface
- **Linux System Programming:** POSIX APIs, System Calls, Socket Programming, TCP/IP, UDP, select(), poll(), fcntl(), IPC, Signals, File I/O, Process Management, Multithreading, Network Programming
- **Embedded Linux:** Buildroot, OpenWrt, U-Boot, Cross Compilation, BusyBox, ARM-based Systems, BSP Development, Root Filesystem, Linux Kernel Configuration
- **Industrial Communication:** Modbus TCP, EtherNet/IP (CIP), Industrial Ethernet, Protocol Packet Parsing, Network Diagnostics
- **Debugging & Performance Analysis:** GDB, KGDB, strace, ftrace, dmesg, Linux Logs
- **Multimedia & Audio:** ALSA, GStreamer, I2S, USB Camera Integration, INMP441 Microphone
- **Development Tools:** Git, Make, CMake, GCC, Linux CLI, VS Code
- **Hardware Platforms:** Raspberry Pi 4B, BeagleBone Black, STM32F4, u-blox LARA-L6

Professional Experience

Embedded Systems DeveloperMarch 2025 – Present Pelorus Technologies Pvt. Ltd., Bangalore

- Developed Linux character and platform drivers for ARM-based embedded systems, enabling kernel-space interaction with GPIO peripherals and custom hardware interfaces.
- Designed and implemented kernel-space communication mechanisms using circular buffers, managed memory allocation, and synchronization primitives to support reliable data exchange between user space and kernel space.
- Integrated GPIO peripherals using the Linux GPIO Descriptor (gpiod) framework and Device Tree, enabling dynamic hardware resource discovery and platform-independent driver development.
- Developed sysfs interfaces to expose kernel driver attributes, allowing user-space applications to configure and monitor embedded peripherals without modifying kernel code.
- Integrated an INMP441 I2S microphone through the ALSA framework and optimized embedded multimedia pipelines using GStreamer for real-time audio capture and streaming.

- Built and customized Embedded Linux platforms using OpenWrt, Buildroot, and U-Boot, including kernel configuration, root filesystem integration, and cross-compilation for ARM targets.
- Developed Linux system programming utilities in C for industrial communication using POSIX socket APIs, implementing TCP/UDP networking, protocol packet parsing, timeout handling, and concurrent device discovery.
- Performed hardware bring-up, BSP customization, firmware analysis, and embedded Linux debugging using GDB, dmesg, strace, and kernel log analysis.
- Collaborated with embedded software development activities including driver integration, hardware validation, protocol debugging, and system-level testing on Raspberry Pi and BeagleBone platforms.

Projects

Project 1: Linux Character Platform Driver for Embedded Surveillance System

- Designed and developed a Linux character platform driver for an ARM-based embedded Linux platform, providing a kernel-space interface for real-time audio data exchange between user-space applications and hardware peripherals.
- Implemented an in-kernel circular buffer using dynamic memory allocation (`devm_kzalloc()`, `krealloc()`) and mutex synchronization to support reliable, low-latency buffering of PCM audio streams.
- Developed platform drivers using the Linux GPIO Descriptor (`gpiod`) framework with Device Tree integration to dynamically discover and configure board-specific GPIO resources during driver initialization.
- Exposed configurable driver parameters through Sysfs attribute groups, enabling user-space applications to monitor and control hardware resources without requiring custom ioctl interfaces.
- Implemented robust driver initialization and cleanup routines using managed kernel resource APIs, ensuring safe module loading/unloading and preventing resource leaks.
- Integrated an I2S digital microphone with the ALSA subsystem and collaborated with user-space multimedia components to enable real-time audio acquisition for embedded surveillance applications.
- Participated in BSP customization, Linux kernel configuration, and embedded Linux platform integration, including bootloader configuration, root filesystem integration, and cross-compilation for ARM targets.
- Performed kernel debugging using `dmesg`, kernel logs, and driver instrumentation to validate driver functionality, hardware initialization, and runtime stability.

Project 2: Industrial Network Discovery and Protocol Analysis Framework

- Developed a native Linux system programming application in C for automated discovery and identification of industrial Ethernet devices using POSIX socket APIs.
- Designed a scalable network discovery framework utilizing TCP/IP and UDP communication to enumerate embedded controllers and identify active services across industrial networks.
- Implemented non-blocking socket communication using `fcntl()`, `select()`, socket timeouts, and asynchronous connection handling to improve network scan efficiency and responsiveness.
- Developed protocol serialization and deserialization routines for Modbus TCP requests and responses, implementing Report Slave ID and Read Device Identification services.
- Implemented low-level packet parsing to decode binary protocol payloads and extract device metadata including vendor information, firmware revision, product identifiers, and serial numbers.
- Developed protocol handling modules for industrial Ethernet communication, including request generation, response validation, exception handling, and communication retry mechanisms.
- Performed byte-level packet analysis and protocol validation using Wireshark to verify packet structures, communication sequences, and protocol compliance during development and debugging.
- Built reusable networking modules using Linux socket APIs including `socket()`, `connect()`, `send()`, `recv()`, `select()`, and `fcntl()`, emphasizing modular design and maintainability.
- Applied Linux system programming concepts including binary data manipulation, network protocol implementation, timeout management, file descriptor handling, and robust error recovery.

Project 3: Embedded Linux BSP Development and Platform Integration

- Built and customized an Embedded Linux Board Support Package (BSP) for an ARM-based platform using OpenWrt, including Linux kernel configuration, root filesystem generation, package integration, and cross-compilation.
- Customized U-Boot bootloader configuration and Device Tree to initialize platform peripherals, networking interfaces, GPIO resources, and board-specific hardware during system boot.
- Configured and integrated Embedded Linux software components including BusyBox, kernel modules, startup services, and platform initialization scripts for production-ready system deployment.
- Performed firmware image analysis to understand bootloader stages, kernel boot flow, root filesystem organization, and embedded Linux startup sequence.
- Configured Linux networking components including routing tables, firewall policies, and secure communication paths to support embedded edge deployments.
- Integrated platform-specific device drivers and validated peripheral initialization during kernel boot and runtime operation.
- Performed board bring-up, platform validation, kernel debugging, and system-level troubleshooting throughout the complete boot sequence from bootloader to user space.
- Utilized Buildroot/OpenWrt build systems, cross-compilation toolchains, and Linux development utilities to automate embedded firmware generation and deployment.

Education

Post Graduate Diploma in Embedded Systems 2024 – 2025
Indian Institute of Embedded Systems (IIES), Bangalore

Relevant Coursework: Linux Device Drivers, Linux Kernel Internals, Embedded Linux, Advanced C Programming, RTOS, Microcontroller Architectures

Bachelor of Technology (B.Tech.) in Electronics and Communication Engineering 2020 – 2024
Bangalore Institute of Technology, Bangalore