Vaibhav Kshirsagar

+91-74982 36616 | maulikshirsagarvk18@gmail.com | github.com/Vaibhav0216 | Pune, India

Professional Summary

Embedded Systems Engineer with **2.5+ years** of production experience spanning **Linux BSP development**, bare-metal and RTOS-based firmware on **STM32MP1 (dual Cortex-A7/M4)**, **ESP32**, and **STM32F4** platforms. Hands-on expertise in **Linux device driver development**, **Device Tree configuration**, **DRM/KMS display** and **touchscreen integration**, and hardware interface bring-up (UART, SPI, I2C, RS-485, GPIO). Proficient in **C programming**, **FreeRTOS multithreading & IPC**, cross-compilation toolchains, and the **Yocto build system**. Architected and deployed a production IoT gateway platform — from Linux kernel-level drivers to AWS cloud backend — currently serving **70+ field devices** across 5 industrial customers.

EDUCATION

Government College of Engineering, Karad **2020 – 2024**
*Btech - Electronics and Telecommunication - **CGPA** - **7.52*** *Karad, India*

R.P. College **2020**
*HSC - General Science - **Percentage** - **78.77%*** *Dharashiv, India*

Technical Skills

Languages	C (primary), C++, Python, Bash/Shell Scripting, JavaScript, SQL
SOC / Embedded	STM32MP157 (Cortex-A7/M4), STM32F4 (Cortex-M4), ESP32, Raspberry Pi, FreeRTOS, ESP-IDF
Linux BSP	Device Tree (DTS/DTSI), Device Driver development, DRM/KMS framebuffer, Linux GPIO chardev, Yocto build system, U-Boot, cross-compilation (arm-linux-gnueabihf), systemd
Hardware Interfaces	UART, SPI, I2C, GPIO, RS-485, Modbus RTU/TCP, ADC, USB, Ethernet, NVIC/EXTI
Display / Touch	MIPI-DSI, DRM/KMS, Linux framebuffer (/dev/fb0), capacitive touchscreen (Linux input subsystem), TFT LCD
Protocols	MQTT (TLS 1.2), TCP/IP, Modbus RTU/TCP, HTTP/HTTPS, UART, SPI, I2C, RS-485, WebSocket
Tools & DevOps	GDB, JTAG, logic analyser, Git, GitLab CI/CD, Docker, CMake, Make, Jenkins
Cloud / IoT	AWS IoT Core, Lambda (Python), API Gateway, DynamoDB, RDS (PostgreSQL), S3
Security	Secure Boot V2, Flash Encryption, TLS 1.2, JWT Authentication

Experience

AMSET Technologies Pvt. Ltd. **July 2024 – Present**
Embedded Systems Engineer *Pune, India*

Project 1: Industrial Modbus-to-MQTT IoT Gateway *ESP32 • FreeRTOS • Modbus RTU • MQTT/TLS • C • AWS IoT Core*

- Designed complete gateway firmware architecture on ESP32 using **FreeRTOS** — structured as independent tasks communicating via queues and semaphores for Modbus polling, MQTT publishing, SD logging, and Wi-Fi management.
- Wrote a **custom Modbus RTU master driver** in C over UART, handling RS-485 framing, CRC validation, exception codes, and configurable polling intervals; validated against physical field devices.
- Implemented **MQTT client stack with TLS 1.2**, structured JSON payloads, and automatic reconnection logic for stable cloud publishing to AWS IoT Core.
- Engineered **SD-card offline data logging** (16 GB, FAT32) with RTC timestamps as a circular buffer; stress-tested with 20,000+ log files for durability under power-cycle conditions.
- Hardened firmware security using **ESP32 Secure Boot V2 + Flash Encryption**, preventing firmware extraction and enabling authenticated OTA updates.
- Added **edge-level automation**: configurable rule engine and scheduling that executes locally without cloud dependency.
- Built a **lightweight HTTP web server** on the device for Wi-Fi provisioning and remote diagnostics.
- Debugged hardware/software issues using **JTAG, ESP-IDF GDB monitor**, and serial trace analysis; resolved race conditions in FreeRTOS task synchronization.
- **Deployed to 70+ industrial field devices** for multiple customers — firmware is in active production.

Project 2: Multi-Tenant IoT Cloud Platform *AWS IoT Core • DynamoDB • PostgreSQL • Lambda (Python) • Next.js • WebSocket*

- Architected full cloud backend from scratch after evaluating ThingsBoard limitations; designed for multi-tenant isolation so each customer accesses only their own devices, dashboards, and reports.
- Built **high-speed telemetry ingestion pipeline**: AWS IoT Core Rules → Lambda → DynamoDB with GSIs optimized for time-series queries.

- Developed **real-time alert system** using WebSocket API Gateway + DynamoDB Streams with sub-1-second alert delivery to dashboards.
- Implemented **remote firmware logging**: device logs pushed to S3 on alert trigger for post-mortem analysis.
- Modeled relational data (customers, users, devices) in **PostgreSQL (RDS)** with JWT-secured API authentication.
- Platform **currently serving 5 industrial customers** with hourly, daily, monthly, and custom-range reporting.

Project 3: Firmware Code-Generation Tool (Internal Tooling) *Python • C Code Generation • Excel Automation*

- Built a **Python CLI tool** that reads customer-provided Modbus register maps (Excel) and auto-generates complete, production-ready C source files for polling, parsing, scaling, and MQTT payload formation.
- Eliminated repetitive per-device firmware changes; **reduced new-device onboarding time from days to under an hour**.
- Centralized all configuration (baud rate, function codes, polling intervals, JSON keys) in one place, cutting firmware defect rate across all deployed gateways.

Project 4: STM32MP157 Industrial IoT Gateway — Linux BSP & Application *C • STM32MP1 • Linux • Device Tree • DRM/KMS • Yocto • Modbus • MQTT*

- Brought up a **dual-core STM32MP157F-DK2** (Cortex-A7 Linux + Cortex-M4 FreeRTOS) gateway; configured **Device Tree (DTS/DTSI)** for UART7 (RS-485), GPIO, and peripheral pin-muxing.
- Developed **Linux GPIO chardev driver interface** (/dev/gpiochip) for RS-485 direction control; implemented UART7 enablement via Device Tree overlays for Modbus RTU communication.
- Integrated a **MIPI-DSI TFT touchscreen display** using the **DRM/KMS** subsystem; configured DSI panel driver and Device Tree bindings for the display pipeline, and built interactive UI with touch-based settings using the Linux input event subsystem.
- Implemented **multi-threaded gateway application** in C with pthreads — Modbus polling, MQTT publish (TLS), offline SD storage with replay, connectivity monitoring, and display rendering running as concurrent threads.
- Built **offline-first data pipeline**: sensor readings stored as JSON files on SD card when network is unavailable; replay thread publishes queued data upon reconnection with zero data loss.
- Configured **Yocto build system** for custom Linux image with libmodbus, mosquitto, sqlite3, cJSON, OpenSSL, and libcurl; set up **cross-compilation toolchain** (arm-linux-gnueabihf) for iterative development.
- Managed **Cortex-A7 Cortex-M4 communication** using OpenAMP/RPMsg for dual-core coordination; M4 handles real-time Modbus tasks with hardware watchdog (IWDG).
- Deployed with **systemd service management**; integrated syslog/journald for diagnostics and remote log upload to Google Sheets via HTTPS webhooks.

Syrma SGS Technology **January 2024 – May 2024**

Software Developer Intern *Pune, India*

CI/CD Pipeline Analytics & BOM Fault Detection Tools *Python • Bash • GitLab API • MERN • Nexar API*

- Automated CI/CD pipeline reporting via **Bash scripting + GitLab API**, generating per-contributor Excel summaries of success rates, failure reasons, and stage bottlenecks — **saved 2+ hours/day** of manual effort.
- Improved pipeline success rate from **80% to 90%** by identifying recurring failure stages and enabling proactive resolution.
- Built a **BOM fault detection tool** using Nexar API to automatically validate PCB component availability; reduced 3+ hour manual BOM review to minutes for 100-item lists.

Projects

GPIO & EXTI Driver — STM32F4 (Bare-Metal) *C • ARM Cortex-M4 • STM32CubeIDE • No HAL* **2023**

- Developed a **complete GPIO peripheral driver from scratch** by directly programming AHB1 clock, GPIO registers (MODER, OTYPER, OSPEEDR, PUPDR) without any HAL/LL abstraction layers.
- Implemented **NVIC + EXTI interrupt handling** with dynamic line-to-pin mapping; demonstrated deep understanding of ARM Cortex-M4 exception model and memory-mapped I/O.
- Established proficiency in **bitwise manipulation, peripheral register access,** and hardware-level debugging.

Embedded Linux IPC Gateway (Unix Domain Sockets) *C • Linux • AF_UNIX • I2C • Syslog* **2025**

- Designed a **multi-process IPC system** using Unix Domain Sockets for decoupled sensor acquisition and data processing services; interfaced hardware via **Linux I2C driver** (/dev/i2c-*) with CRC validation.
- Implemented custom protocol with structured headers, connection retry logic, and **syslog/journald** integration for production-level diagnostics.

Leadership

Alpha Geeks, GCEK — Content Team Head **April 2021 – January 2023**

- Led CP/DSA workshops and organized coding contests; grew contest participation by **50%** during tenure.

Certifications

- Cloud Computing — NPTEL (87%)
- Python Programming Language — NPTEL (96/100)