**ABINASH N**

**Email ID: abinash@zumisolutions.com**

|  |  |
| --- | --- |
| **At Zumi** | **Embedded Software Engineer**, at Zumi Solutions (P) Ltd., Bangalore. |
| **Current and Previous Experience** | **October 03, 2021**– Till now, Zumi Solutions (P) Ltd., Bangalore.  3 Years 8 Months Experience in:   * Linux CLI. * Firmware design and development. * Programming Language: C, CPP, Embedded C application design and development. * Device drivers, Kernel module. * Middleware development. * Openssl, tls, rsa, aes, hmac. * System daemon. * IOT. * IPC Mechanism-Socket Programming, Message queue, pipe. * State machines. * Thread handling - pThreads for Multi-threading, Mutex. * BLE \_ Wi-Fi initialization,Configuration,communication. * Debuggers: GDB, Valgrind. * Microprocessor - STM32MP157,ESP32,. * Shell scripting for application startup in user space. * Yocto project. |
| **Projects** | **Project 1:** i.MX8MM Board Bring-up and Peripheral Validation  **Platform:** NXP i.MX8M Mini (i.MX8MM) SoC  **Embedded Vertical:** Embedded Systems / Industrial  **Bootloader:** U-Boot  **OS:** Embedded Linux (Yocto Project, Kernel 5.x)  **Language:** C, Shell Scripting  **Description:** Performed complete board bring-up of a custom i.MX8MM-based hardware platform. Validated critical hardware peripherals and ensured stable Linux booting with device tree and driver configurations. Focused on both low-level hardware interface testing and Linux-level peripheral functionality.  **Roles & Expertise:**   * Booted the i.MX8MM custom board using UUU (Universal Update Utility) via USB-OTG for initial flashing and recovery. * Ported and configured U-Boot for custom SOM and baseboard combination. * Built and customized Linux kernel and device tree to enable onboard peripherals. * Verified USB OTG/Host mode functionality with mass storage and HID devices. * Validated GPIO inputs/outputs via sysfs and confirmed pinmux settings. * Tested MAC (Ethernet) by enabling network interface and transferring files using SCP. * Performed GSM modem testing over USB using AT commands via minicom. * Verified SOM-level interfaces like UART, I2C, SPI, and audio codec initialization. * Automated basic tests and logs using shell scripts for factory-level validation. |
| **Project 2:** Qualcomm IPQ5018 WiFi Router Firmware Development  **Platform:** Qualcomm IPQ5018 (AP-MP03.3)  **Embedded Vertical:** Networking / Wireless Routers  **Bootloader:** U-Boot  **OS:** OpenWrt 19.07-SNAPSHOT (LuCI NHSS.QSDK.12.2.r4)  **Kernel Version:** Linux 5.4.213  **Language:** C, Shell scripting  **Description:**Developed custom firmware using Qualcomm SDK (QSDK) for a WiFi router based on IPQ5018. Performed complete board bring-up and customized firmware for enhanced routing, VLAN, and QoS support. Integrated and validated 2.4GHz and 5GHz wireless modules, LAN/WAN ports, and NAND flash-based boot.  **Roles & Expertise:**   * Performed board bring-up with QSDK, ported bootloader and kernel to match board schematic and memory layout. * Configured NAND flash (GD5F1GQ4RE9IG) and verified U-Boot environment storage. * Enabled DDR3L (256MB RAM, MT41K256M16TW) with custom DDR init values. * Integrated QCA8337N switch for 4 LAN ports (1x WAN + 3x LAN), updated DTS and switch config. * Verified WiFi modules: * WiFi0 (2.4GHz, IPQ5018 internal radio) * WiFi1 (5GHz, QCN6122 via PCIe) * Modified /etc/config/network, /etc/config/wireless, and /etc/config/firewall for optimal routing, NAT, VLAN, and security. * Implemented VLAN, QoS, and firewall rules to support multi-tenant use cases. * Debugged firmware boot failures, optimized kernel logs, and resolved driver compatibility issues. * Utilized OpenWrt UCI CLI tools and LuCI interface for quick config changes. * Ran real-time traffic simulation tests to validate throughput, latency, and roaming. |
| **Project 3:** Custom Firmware Update Framework on AM437x Board  **Platform:** TI Sitara AM437x  **Embedded Vertical:** Industrial / Embedded Systems  **Bootloader:** MLO + U-Boot (Custom Modified)  **OS:** Linux (Dual RootFS – Main OS and Tiny OS)  **Language:** C, Shell scripting  **Update Tool:** SWUpdate (customized)  **Protocol Stack:** TCP/UDP/IP (custom communication protocol)  **Description:**Implemented a robust and reliable firmware update system on the AM437x platform using SWUpdate and dual-partition NAND layout. The design ensures minimal downtime and safe recovery through two separate OS environments. Firmware update data is transmitted over the network (TCP/UDP) from a PC and flashed to a custom NAND partition.  **Roles & Expertise:**   * Designed and implemented a dual-rootfs architecture: * Main OS (OS1) – Handles user applications and standard operations * Tiny OS (OS2) – Lightweight OS dedicated to handling firmware updates * Allocated NAND flash memory into 80% for OS1 and 20% for OS2. * Modified MLO and U-Boot scripts to support partition switching and boot control logic. * Integrated and customized SWUpdate to: * Receive firmware image via TCP/UDP * Flash OS1 partition with new rootfs, kernel, and app components * Perform sanity checks and revert in case of failure * Developed firmware-update application in OS2 to handle flashing, verification, and reboot to OS1. * Created Main\_OS\_app in OS1 to initiate update request and trigger reboot to OS2. * Built Final\_client\_pc tool on PC for centralized management: * Communicates with both OS1 and OS2 over TCP/UDP/IP * Sends the update image file * Coordinates switching, flashing, and verification * Ensured seamless system reboot sequence: * Normal boot to OS1 → Update triggered → Reboot to OS2 → Flashing → Reboot to OS1 * Conducted end-to-end testing including NAND bad block handling, flashing integrity, and rollback on error |
| **Project 4:** BABY SMILE – IoT-based Baby Monitoring Device  **Platform:** ESP32 (Xtensa Dual-Core 32-bit LX6 MCU)  **Embedded Vertical:** Consumer IoT / Healthcare  **OS:** Bare-metal / FreeRTOS (Optional)  **Language:** C, C++  **SDK/Tools:** ESP-IDF, Bosch BSEC Library, PlatformIO, WhatsApp API (via cloud integration)  **Description:**Developed a smart baby monitoring device using the ESP32 platform, integrating biometric and environmental sensors for continuous health and safety tracking. The device uses WiFi to send live status and alert messages to WhatsApp via a cloud API, providing real-time feedback to parents. Advanced air quality monitoring is implemented using Bosch BME680 with BSEC software for IAQ, CO₂, and bVOC estimation.  **Roles & Expertise:**   * Designed and developed firmware for multi-sensor integration using I2C/SPI/UART on ESP32. * Interfaced and processed data from: * BME680 – Temperature, Humidity, Pressure, VOC (Air Quality) * Oxygen Sensor – Monitors O₂ concentration * Weight Sensor (HX711) – Baby weight and growth tracking * Microphone – Captures baby cry sounds * Speaker – Delivers alerts and communication tones * Fan Control – Maintains ambient comfort using GPIO-based relay or PWM * Integrated Bosch BSEC Library for intelligent IAQ and gas analysis. * Implemented real-time alerts for oxygen level drops, high VOCs, weight loss, or temperature spikes. * Enabled WiFi-based WhatsApp messaging via secure HTTPS cloud API (e.g., Twilio, CallMeBot) to send live status or emergency alerts to parents’ mobile phones. * Developed audio streaming pipeline with I2S + DMA for live monitoring or alert playback. * Designed low-power modes for sleep cycles and sensor wake scheduling. * Built modular and scalable firmware architecture for feature extension (e.g., BLE, camera). * Conducted thorough environmental validation testing under different indoor scenarios. |
| **Project 5:** Custom AM437x Board Bring-Up and Peripheral Enablement  **Platform:** TI Sitara AM437x (Custom Hardware)  **Embedded Vertical:** Industrial Automation / IoT  **Bootloader:** U-Boot (Custom Configuration)  **OS:** Embedded Linux 5.10 (RT-enabled)  **Build System:** Yocto Project (Dunfell / Hardknott)  **Language:** C, Device Tree, Shell  **Description**:Performed complete board bring-up for a custom AM437x-based hardware platform. Ported Linux 5.10 kernel and enabled required peripherals using Yocto. Built a minimal root filesystem and customized board-specific configurations, including device trees and kernel patches.  **Roles & Expertise:**   * Configured Yocto-based Linux build with custom layer for AM437x board: * Used meta-ti, meta-openembedded, and custom meta layers * Customized local.conf, bblayers.conf, and machine configuration * Ported and verified Linux kernel 5.10 on the custom board: * Configured am437x.dts and board-specific overlays * Applied required patches for pinmux, clocks, and peripheral support * Brought up essential peripherals: * Ethernet – Configured CPSW driver and MAC address handling * CAN (DCAN1/DCAN0) – Enabled CAN controller, verified with can-utils * EEPROM (SPI/I2C) – Read/write using /sys and test applications * GPIO / LEDs – Configured via device tree, tested with sysfs/export interface * SD Card Interface – Enabled MMC interface, verified boot and file access * Verified U-Boot SPL/MLO and full boot path: * Built and flashed MLO + u-boot.img * Configured bootargs and kernel image loading from SD/NAND * Performed device bring-up validation using serial console, logic analyzer, and test software. * Created scripts for flashing rootfs and kernel to NAND with bad block handling. |
| **Project 6:** CVE Vulnerability Analysis and Mitigation on Custom Linux **Platform :** Linux  **OS Components**:Kernel: linux-rt-5.10\_new.100+gitAUTOINC+204ec708dc-g204ec708dc  **Root File System:** Custom Yocto-based (Tiny + TISDK base image)  **Toolchain:** cve-bin-tool, Yocto, Patch management  **Security Focus:** Common Vulnerabilities and Exposures (CVE)  Description:Performed complete CVE scanning and vulnerability remediation for a customized Linux kernel and root filesystem. Used the cve-bin-tool to detect known CVEs in userland binaries and kernel components, and systematically applied upstream/backported patches to mitigate them.  **Roles & Expertise:**   * Scanned kernel and rootfs images using cve-bin-tool to identify outdated or vulnerable packages. * Generated CVE reports for: * linux-rt-5.10\_new.100+gitAUTOINC+204ec708dc * Root filesystem binaries (e.g., busybox, openssl, libxml2, etc.) * Reviewed CVE database entries and security advisories to identify: * CVSS severity * Affected versions * Recommended patches or updated package versions * Applied kernel patches to fix critical CVEs related to: * Network stack * Filesystem bugs * Real-time scheduling issues * Rebuilt the Linux kernel and verified stability with real-time application tests. * Updated vulnerable userland packages in the Yocto rootfs via SRCREV, PREFERRED\_VERSION, or custom patching. * Validated that patched components were correctly integrated using a second pass with cve-bin-tool, confirming no remaining critical CVEs. * Documented CVE IDs, patch status, and applied fixes for compliance and traceability. |
| **Project 7:** Industrial Protocol Integration, System Optimization, and Network Features on AM437x  **Platform:** TI AM437x (Custom Board)  **Embedded Vertical:** Industrial Automation / Smart Gateway  **OS:** Yocto-based Embedded Linux (Tiny RootFS)  **Language:** C, Shell  **Protocols:** Modbus TCP, CAN 2.0B, RSTP, DHCP, SNTP  **Tools:** libmodbus, CAN-utils, mstpd, BusyBox utilities  **1.Modbus to CAN Gateway Implementation**  **Objective:**Enable seamless communication between PC-based Modbus clients and multiple CAN-connected modules using the AM437x as a Modbus TCP Server and CAN bridge.  **Roles & Achievements:**   * Developed Modbus TCP Server on AM437x to handle incoming function codes and map them to CAN-based read/write commands. * Implemented CAN fragmentation mechanism to send/receive data longer than 8 bytes using custom protocol logic and CRC validation. * Designed MAC filter-based routing to manage multiple CAN modules behind the AM437x. * Ensured Modbus register space mapped cleanly to device-specific CAN messages and statuses. * Verified end-to-end communication: PC (ModScan32) → AM437x → CAN Modules, supporting bi-directional control and telemetry.   **2. Board Boot Time Optimization**   * Reduce boot time from ~1 minute to <20 seconds for critical industrial use cases.   **Roles & Achievements:**   * Created Tiny RootFS with BusyBox and minimal dependencies. * Removed all unnecessary system services, network daemons, and graphical components. * Optimized systemd and inittab entries to parallelize initialization and minimize delays. * Kept only essential libraries (e.g., libmodbus, libc, libnl) to support core features.   **3. SNTP & Time Zone Setup**   * Ensure real-time clock accuracy with local time zone settings for Kolkata (IST).   **Roles & Achievements:**   * Enabled SNTP client on Tiny RootFS using BusyBox ntpd or custom implementation. * Configured local NTP server fallback with AM437x acting as SNTP client. * Applied TZ=Asia/Kolkata for system-wide date/time synchronization. * Verified RTC + software clock sync during boot and runtime.   **4. DHCP Client with Custom System Daemon**   * Enable dynamic IP assignment without using traditional DHCP daemons (e.g., dhclient, systemd-networkd).   **Roles & Achievements:**   * Wrote a lightweight DHCP client script (optioMax) in shell with interface detection and IP lease handling. * Integrated script into boot process and ensured interface comes up with minimal latency. * Monitored link state using ethtool and managed fallback to static IP if DHCP fails.   **5. RSTP Support for Ethernet Ring Topology**   * Ensure network resilience using Rapid Spanning Tree Protocol (RSTP) for industrial ring topology.   **Roles & Achievements:**   * Enabled RSTP support using mstpd (standalone IEEE 802.1D-2004 variant). * Configured mstpctl with port priorities, path costs, and bridge priorities. * Verified ring topology behavior with <1-second failover during link break scenarios. * Tuned link detection timeouts for faster convergence in harsh industrial environments. |
|  | **Project 8:** Industrial Automation Unit  **Platform:** NXP i.MX93 Processor  **Bootloader**: U-Boot  **Operating System:** Embedded Linux (Yocto Project)  **NPU**: Arm Ethos-U65 MicroNPU  **Roles & Expertise:**  **Board Support & OS Development**   * Ported the Yocto BSP to custom hardware based on the i.MX93 SoC. * Developed and customized Linux Kernel and U-Boot for the board. * Created and maintained a custom Yocto meta-layer integrated into the SDK and CI/CD build pipeline.   **Display and GUI**   * Configured SPI Display to auto-initialize on boot and show system IP address. * Integrated MIPI-DSI display panel support. * Enabled QT, RDP, and GUI Guider for graphical HMI and remote desktop control.   **NPU Acceleration & AI Integration**   * Enabled NPU acceleration using Arm Ethos-U65 with TensorFlow Lite models. * Used eIQ Toolkit, Vela compiler, and EthosU TFLite delegate to offload inference to NPU. * Benchmarked NPU models using benchmark\_model, tested models like face recognition, wake word detection, and object classification.   **Voice Interaction & Audio Systems**   * Integrated NXP’s VIT (Voice Intelligent Technology) with a PDM microphone for wake-word and speech-to-intent detection. * Configured ALSA, audio codecs, and sound card drivers. * Developed and debugged with various audio tools: pyaudio, pydub, ffmpeg, flac, portaudio, bluealsa, jack, and wireplumber.     **Peripheral Interfaces – Enablement & Testing**  Ported and validated interfaces on the custom i.MX93 board:   * USB * Wi-Fi / Bluetooth * SPI Display (ILI9341) * MIPI-CSI Camera * MIPI-DSI Display * Ethernet * Audio Codec + PDM Microphone * Wake Word Pipelines * PWM / TPM - GPIO Control (TPM/PWM modules) * Adafruit PCA9685 16-channel Servo Motor control using I2C and smbus2 in Python   **Industrial Communication Protocols**   * Integrated and tested BACnet and Modbus support for industrial automation applications. * Enabled serial and Ethernet communication between controller and sensors/actuators.   **Advanced Use Cases**  Driver Monitoring System (DMS) implemented using multiple eIQ models on i.MX93:   * Face Detection (Front-facing camera) * Face Landmark Detection * Iris Landmark Detection * Enabled real-time inference via NPU with low-latency camera capture and display.   **Embedded Printing (CUPS)**  Integrated CUPS (Common UNIX Printing System) on Yocto with support for:   * Ethernet * Wi-Fi * USB-based printers   Validated end-to-end printing from custom GUI apps and network devices within the LAN.  **Tools and Technologies Used:**   * Yocto Project (Kirkstone, v6.6) * Linux Kernel, U-Boot, Device Tree * eIQ Toolkit, CMSIS-NN, Vela, TFLite, TFLite Micro * GUI Guider, QT, RDP, Weston/Wayland * Python: pyaudio, pydub, smbus2, ffmpeg, flac, portaudio * ALSA, bluealsa, wireplumber, jack audio, VIT SDK * Modbus, BACnet, SPI/I2C/UART/GPIO/PWM * Wireshark, i2cdetect, v4l2, perf, dmesg, top |
| **Technical Skills** | * **Programming Languages –** C, C++, Embedded C, Shell Scripting, Data Structures * **Operating Systems –** Embedded Linux (Yocto, OpenWrt),FreeRTOS, Bare-metal, Android (basic), Windows (for tools), Linux System Programming. * **Software Tools –** GDB, Valgrind, Code Composer Studio,PlatformIO, Yocto Build System, CVE-bin-tool, UUU Tool, OpenSSL * **Microcontrollers / Processors –** TI AM437x, NXP i.MX8MM, STM32MP157, ESP32, Qualcomm IPQ5018 * **Networking –** TCP/IP, UDP, Modbus TCP, DHCP, SNTP, RSTP, VLAN, NAT, Socket Programming * **Protocols & Interfaces –** I2C, SPI, UART, CAN 2.0B, BLE, Wi-Fi, I2S, GPIO, EEPROM * **Embedded Concepts –** Bootloader (U-Boot), Device Drivers, Kernel Modules, Device Tree, Dual RootFS, SWUpdate, System Daemons, Multithreading (pThreads), IPC * **Sensors & Peripherals –** BME680 (Air Quality), Oxygen Sensor, HX711 (Weight), Microphone, Audio Codec, LEDs   **Security & Encryption –** TLS, RSA, AES, HMAC, CVE Patching |
| **Academic** | * **I completed my B.E. in Mechanical Engineering in 2021 from Apollo Engineering College, Chennai.** |