Zephyr RTOS porting to custom RISC-V based SoC

Mridul M S, Alphin Thomas
Software Engineers, TOSIL SYSTEMS Pvt. Limited
Trivandrum, Kerala, India.

INTRODUCTION

To enable real-time capabilities and expand its ecosystem, a significant milestone was achieved by porting the Zephyr Real-Time Operating System (RTOS) to the Shakti Vajra SoC. Originally, Shakti only supported a 32-bit version of Zephyr tied to the outdated Parashu SoC. This effort successfully upgraded support to a 64-bit environment, ensuring compatibility with modern toolchains, peripherals, and use cases.

The case study highlights the end-to-end process of bringing up Zephyr RTOS on the Shakti platform. Testing was conducted both on QEMU-based emulation and real hardware (Arty A7 100T FPGA with Vajra SoC) to ensure robust functionality.

This porting effort not only demonstrates the adaptability of Zephyr RTOS to custom RISC-V architectures but also establishes a foundation for deploying production-ready real-time applications on Shakti-based platforms.

THE CUSTOMER

The customer is a research-oriented organization engaged in open-source hardware development, focusing on RISC-V based processors and SoC architectures. Their goal is to enable an ecosystem for prototyping, research, and deployment of custom embedded solutions on FPGA platforms.

THE REQUIREMENT

- **Toolchain Setup:** Install and configure RISC-V GCC cross-compilation toolchain (RV64 support)
- **Boot & Firmware:** Use OpenSBI/BBL for system initialization and booting Zephyr.
- **Testing & Validation:** Run Zephyr test suite on Shakti QEMU and Arty A7-100T FPGA then Validate UART console.
- **Integration:** Ensure compatibility with existing Zephyr drivers and Verify 64-bit support extended from earlier 32-bit Parashu port.

Scope of Work: The scope of this case study focuses on the successful porting and bring-up of the Zephyr RTOS on the Shakti Vajra (C-Class RV64) custom RISC-V SoC, implemented on the Arty A7 100T FPGA and validated on both hardware and QEMU. The work involved upgrading Zephyr support from the legacy 32-bit Parashu SoC to 64-bit, integrating the RV64 GCC toolchain, and modifying kernel architecture-specific components to support the Vajra SoC's features such as UART. The development process included environment setup, toolchain integration, FPGA configuration with the Shakti core bitstream, and extensive testing through bare-metal applications, Zephyr test suites, and system-level bring-up on both emulation and FPGA hardware. This comprehensive effort not only enabled real-time processing capabilities on a custom RISC-V SoC but also demonstrated the feasibility of running production-grade RTOS and embedded Linux on the Shakti ecosystem.

SOLUTION PROVIDED

Toolchain and Build Setup: The porting process began with setting up the RISC-V 64-bit cross- compilation toolchain using riscv64-unknown-elf-gcc, ensuring compatibility with the Shakti VAJRA SoC. The Zephyr build environment was configured using west, enabling reproducible builds, incremental updates, and consistent project management across development machines.

Peripheral Enablement: The porting process included enabling UART, which verified using Zephyr sample applications.

Testing and Validation:: Verified booting of Zephyr on both QEMU and the Arty A7 100T FPGA board running the Shakti Vajra SoC. Conducted functional tests such as UART console output.

Technical Specifications

- **Target Platform:** Shakti VAJRA SoC on Arty A7-100T FPGA
- **Processor Core:** 64-bit C-Class in-order 5-stage pipeline
- **RTOS:** Zephyr Real-Time Operating System
- **Toolchain:** RISC-V RV64 GCC cross-compiler integration
- **Board Support Package (BSP):** Custom BSP creation for Vajra SoC
- **Memory Map:** 256 MB DDR, 4 KB ROM
- **Testing Platforms:** QEMU (Shakti RISC-V emulator) and Arty A7-100T FPGA (hardware validation)

Hardware Development

The hardware development for Zephyr RTOS porting to the Shakti Vajra (C-Class RV64) SoC was carried out on the Arty

A7-100T FPGA platform, where the Shakti core bitstream was synthesized and loaded using Xilinx Vivado to enable a 64-bit RISC-V environment. This FPGA-based hardware setup provided a robust platform for integrating the updated Zephyr RTOS, enabling real-time task scheduling, peripheral interfacing, and system testing directly on custom RISC-V hardware.
As shown in Fig. 1, the Arty A7-100T development board is used for Zephyr RTOS porting.

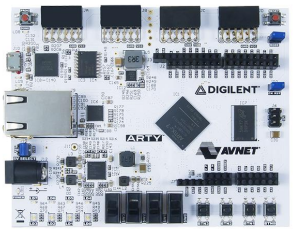

Fig. 1: Artix-7 FPGA Development Board

Software/Firmware Development

The software and firmware development effort primarily focused on porting the Zephyr Real-Time Operating System (RTOS) to the Shakti Vajra (C-Class RV64) SoC deployed on the Arty A7-100T FPGA board. Initially, Shakti only supported a 32-bit version of Zephyr based on the Parashu SoC, which was outdated. The objective was to upgrade this to a 64-bit architecture, ensuring compatibility with the Vajra core. This required modifying Zephyr's architecture-specific files, integrating the RV64 GCC toolchain, peripherals, and device tree configurations.
The development process also included building and validating the environment across both QEMU emulation and the physical Arty A7-100T hardware. Firmware such as OpenSBI and Berkeley Boot Loader (BBL) were used as intermediate boot layers to initialize the hardware and hand over control to the kernel. Extensive testing was performed with bare-metal applications (e.g., "Hello World" and memory tests) before integrating Zephyr. Final validation confirmed that Zephyr booted reliably on the Vajra SoC, enabling real-time processing capabilities on a fully open-source RISC-V platform.

System Integration & Testing

The integration process involved bringing up Zephyr RTOS on the Shakti Vajra SoC (C-class RV64) using both QEMU-based simulation and hardware validation on the Arty A7-100T FPGA. The system setup included configuring the RISC-V GCC toolchain. Architecture-specific modifications were applied to Zephyr to support the 64-bit Shakti platform, including updates to kernel files and driver layers.
Testing was conducted in stages to ensure reliability across different environments. Initial validation began on QEMU to simulate the Shakti core, confirming kernel boot-up and basic task scheduling. This was followed by deploying the system on the Arty A7-100T FPGA, where bare-metal tests such as UART communication. The successful integration demonstrated the feasibility of porting Zephyr to a custom RISC-V SoC and enabled a production-ready foundation for real-time applications on the Shakti Vajra platform.

CHALLENGES

Toolchain Compatibility Issues

Challenge: Initially, Zephyr only supported 32-bit (RV32) Shakti versions based on Parashu SoC. Porting to 64-bit Vajra (RV64) required toolchain alignment, as the default Zephyr build system was not fully compatible with RV64 targets.

Solution: Integrated and validated the RV64 GCC toolchain with Zephyr, updating build configurations and architecture-specific files to ensure smooth compilation and linking for 64-bit binaries.

Kernel and Driver Integration

Challenge: The Zephyr kernel required modifications to accommodate the Shakti Vajra's unique hardware features (PLIC, CLINT, pin mux, and memory constraints). Missing or incompatible drivers caused boot failures during early bring-up.

Solution: Adapted kernel-level initialization, added Shakti-specific device drivers, and implemented minimal UART and timer support to ensure successful RTOS boot and multitasking.

FPGA Hardware Testing and Debugging

Challenge: Testing on the Arty A7 100T FPGA introduced hardware-specific challenges like DDR initialization, timing issues, and UART output mismatches compared to QEMU simulation.

Solution: Validated core functionality on QEMU first, then iteratively tested on FPGA. Debugged hardware issues using serial console logs and memory tests, ensuring consistent operation across both virtual and physical platforms.

Maintaining Compatibility with Older Repository

Challenge: The Shakti repository was 3–4 years old, leading to compatibility issues with the latest Zephyr versions and dependencies.

Solution: Manually patched legacy code, updated device tree structures, and refactored portions of the build system to synchronize with current Zephyr requirements while retaining Shakti-specific configurations.

ACHIEVEMENTS

Successfully ported Zephyr RTOS to the Shakti Vajra (C-Class RV64) SoC, upgrading support from an outdated 32-bit implementation to a robust 64-bit architecture. Comprehensive hardware validation on the Arty A7 100T FPGA board confirmed stable operation and real-time processing capabilities. The Figure 2 shows the successful booting of the Zephyr RTOS on the Shakti Vajra architecture, demonstrating the integration and functionality of the system.

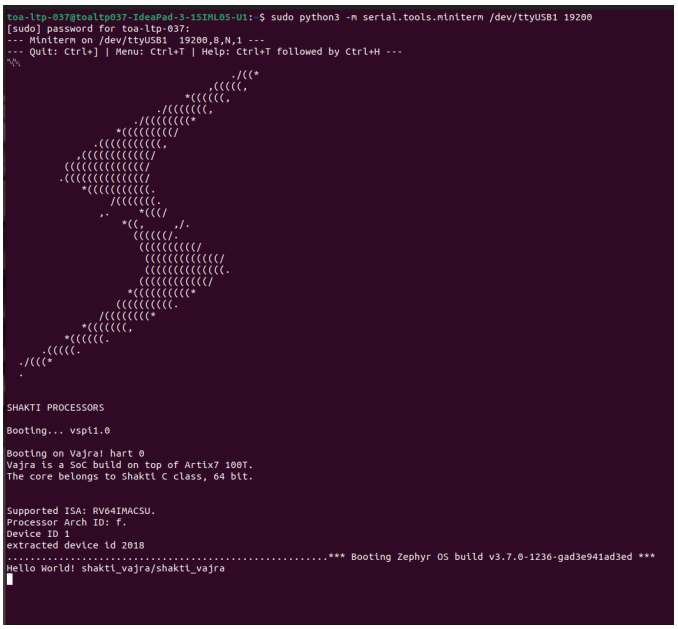

Fig. 2: Booting of the Zephyr RTOS on the Shakti Vajra

In addition, the team achieved end-to-end bring-up of Zephyr RTOS on both QEMU (for early simulation and testing) and the Vajra SoC on FPGA hardware, enabling a flexible development workflow. This milestone significantly extended the ecosystem of open-source RISC-V platforms by demonstrating real-time operating system support on a custom SoC, paving the way for future embedded and industrial applications.

CUSTOMER BENEFITS

- **Accelerated Development Cycles with 64-bit Zephyr Support:** By porting Zephyr RTOS to the 64-bit Shakti Vajra (C-Class RV64) SoC, developers gain access to a modern, production-grade RTOS environment. This significantly reduces development time for real-time applications compared to the outdated 32-bit platform.
- **Enhanced Real-Time Capabilities for Embedded Systems:**The availability of Zephyr RTOS on the Shakti Vajra SoC ensures deterministic task scheduling, low-latency interrupt handling, and robust multi-threading support, enabling reliable performance for time-critical applications.
- **Hardware-Software Co-Validation on FPGA:** Testing on the Arty A7 100T FPGA provides a hardware-accurate environment, allowing developers to validate system functionality before ASIC production. This reduces design risks and accelerates the path from concept to deployment.
- **Open-Source Ecosystem with Long-Term Flexibility:**Leveraging open-source Zephyr RTOS, OpenSBI, and Linux kernel support on Shakti enables customers to avoid vendor lock-in, customize system components, and integrate future upgrades seamlessly.
- **Scalable Platform for Prototyping to Production:** With support for bare-metal, RTOS, and Embedded Linux, the Shakti Vajra SoC provides a single platform for diverse workloads—from lightweight IoT devices to more complex embedded systems—streamlining prototyping and scaling to production.