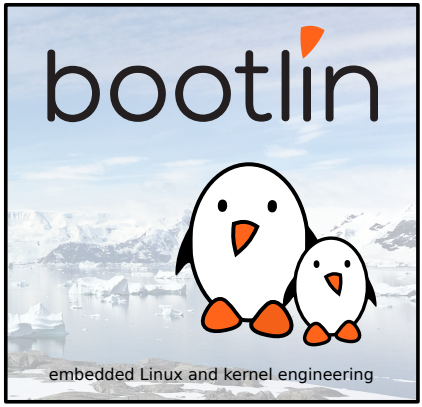
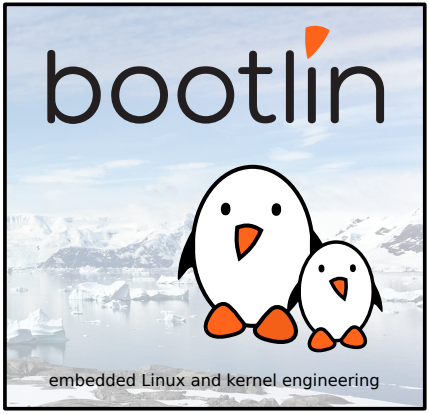
Unlocking the Potential of Suspend to RAM (S2R) using Linux in a Multi-Core, Multi-Firmware Automotive SoC

Gregory CLEMENT
gregory.clement@bootlin.com



Gregory CLEMENT

- Embedded Linux engineer and trainer at Bootlin
  - Embedded Linux **expertise**
  - **Development**, consulting and training
  - Strong open-source focus
- Open-source contributor
  - Contributing to **kernel support** for the Armada 370, 375, 38x, 39x and Armada XP ARM SoCs and Armada 3700, 7K/8K ARM64 SoCs from Marvell.
  - Co-maintainer of mvebu sub-architecture (SoCs from Marvell Engineering Business Unit)
  - Living near **Lyon**, France



Fast and Furious: Accelerating Car Start-Up with Complex Socs

- User Expectation: Instant Car Readiness
- Car should be ready to use upon door opening
- Minimal tolerance for waiting time (even a dozen seconds)
- Complex systems require more time to boot
- Manufacturers are investing time and money to address this issue when using modern SoCs

Reducing Boot Time: Solutions and Limitations

- Fast Boot Solutions:
  - Falcon boot in U-Boot
  - DMA to load the system while hardware is initialized
  - And even more, see other ELC talks about it
- Limitations of Fast Boot Solutions:
  - Efficient mostly only until the first init process
  - Userspace initialization remains a significant challenge
- Beyond Booting: Restoring Systems from Suspend to Ram:
  - An alternative to booting for instant system readiness
  - Avoids the need for userspace initialization

Challenges of Suspend to RAM for Instant System Readiness

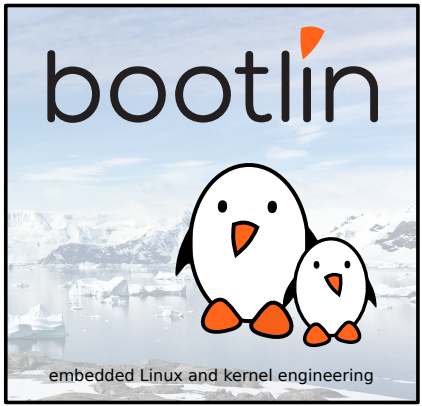
- Ensuring the integrity of memory and hardware state upon resume
- Consuming as little power as possible to maintain the system in suspend for sufficient time
- Achieving a fast resume time
- Finding the correct balance between these three factors is a difficult task

Overview of this talk

- **Hardware & Low-Level Firmware Overview**
  - TI DRA821 SoC (Jacinto family)
  - Components (firmware/hardware) involved in S2R
- **Implementing S2R Support**
  - Techniques and strategies for TI DRA821 SoC
- **Future Challenges & Improvements**
  - Addressing obstacles and enhancing S2R performance

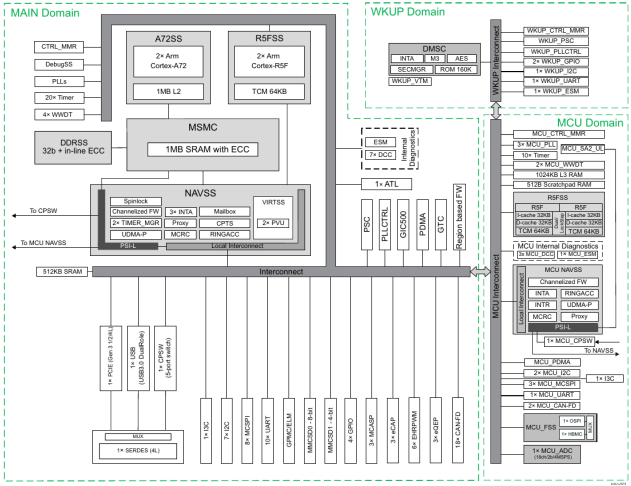
Hardware overview of the TI DRA821

Gregory CLEMENT
gregory.clement@bootlin.com

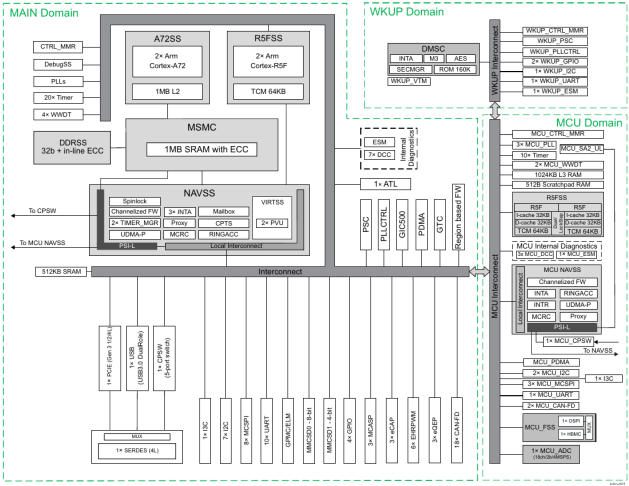


Jacinto Global Architecture

- K3 family: multicore SoC architecture platform
- Designed for low power, high performance and high integrated device architecture
- Jacinto is part of the K3 family and targets automotive



TI DRA821 SoC Architecture: Multicore Processor Overview



- 2 Cortex A72
  - ARM 64 bits
  - MAIN domain
  - Linux, TF-A, U-Boot
- 2x2 Cortex R5
  - ARM 32 bits
  - 2 in MAIN domain
  - 2 in MCU domain
  - DM firmware, U-Boot and custom firmwares
- 1 cortex M3
  - ARM 32 bits
  - WKUP domain
  - Secure firmware

Beyond the SoC: PMIC & DDR in Suspend to RAM

- Power Management IC (PMIC)
  - Powers SoC and DDR
  - Handles wake-up in deep suspend to RAM
  - Implementation differences (one or two PMICs)
- DDR Memory
  - Retains data during suspend
  - Independent of DDR controller during suspend

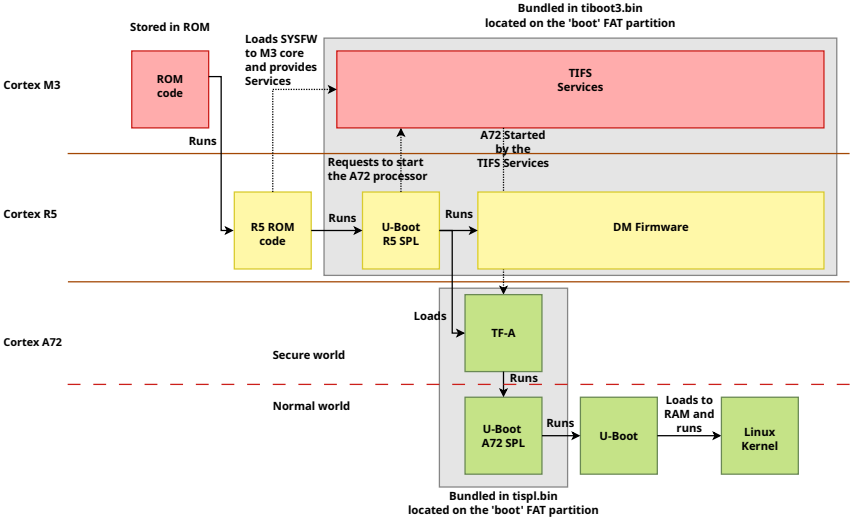
Active & MCU Low Power Modes

- Active Mode (Normal Mode)
  - All cores and power domains powered on
  - PM actions: runtime PM, DVFS, AVFS, CPU idle...
- MCU Mode
  - A72 cores and MAIN power domain shut down
  - Controllers in MAIN power domain also off

Advanced Low Power Modes

- I/O Retention Mode
  - Entire SoC OFF except for a small part managing wake-up GPIOs
- OFF Mode
  - SoC completely off
  - PMIC as the only wake-up source

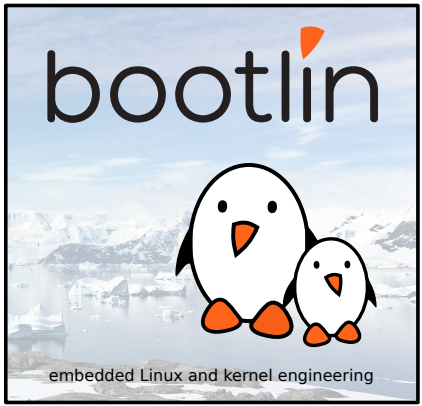
Boot sequence: cores and firmwares involved



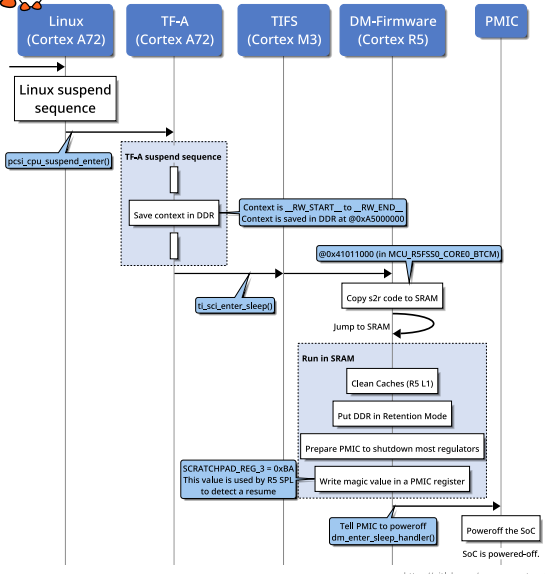
- TI Foundational Security: hardware firewall, CPU control (closed source)
- Device Manager firmware: clock and pin control
- Communication to DM and TIFS through TISCI(API)

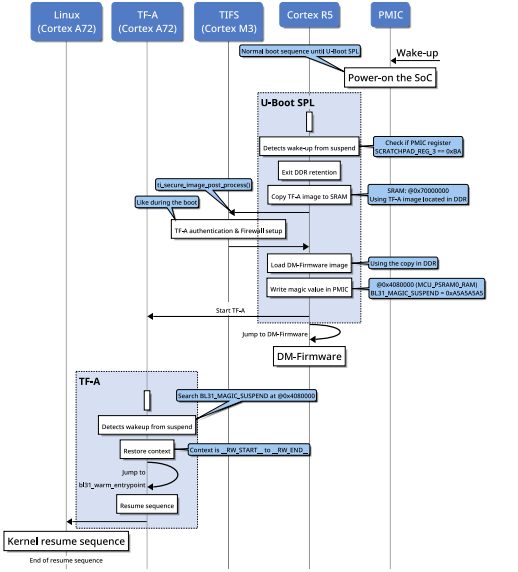
Integrating S2R on the SoC

Gregory CLEMENT
gregory.clement@bootlin.com



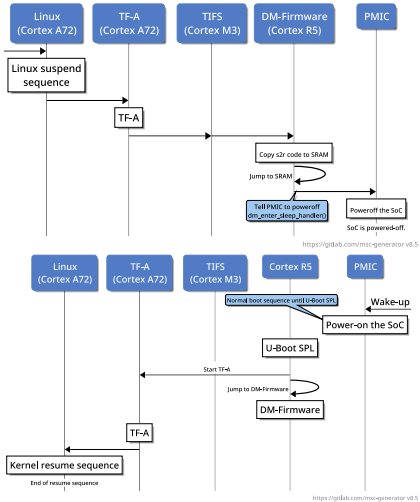
Overall Suspend and Resume Sequence





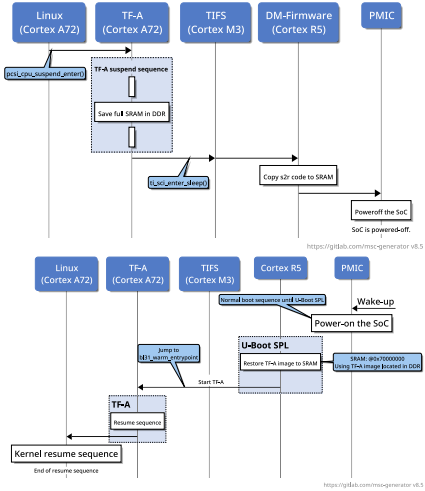
Linux Suspend & Resume Implementation

- Standard sequence through TF-A (Trusted Firmware A)
- Driver-specific considerations
  - General assumption: controller not completely off
  - Reset sequences required for each resume
  - Save and restore register values or configurations



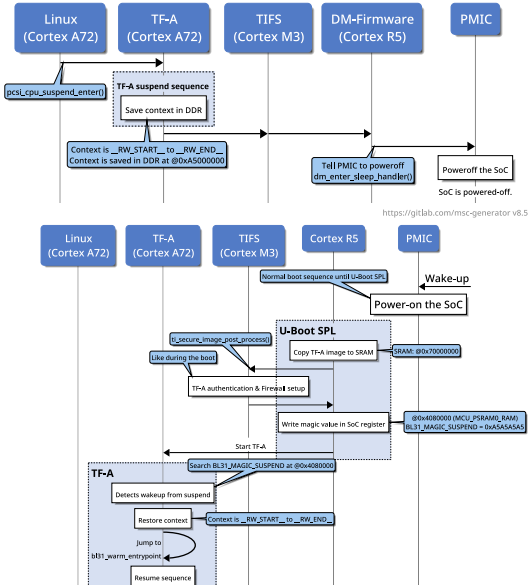
Handling TF-A for Suspend & Resume

- TF-A resides in SRAM by default
- Challenge: SRAM data loss during SoC OFF
- Solution: Save & restore TF-A memory
  - Reuse `bl31_warm_entrypoint` function in TF-A
  - Pass resume address for resuming
- Initial implementation: Copy full SRAM content to DDR and restore during boot



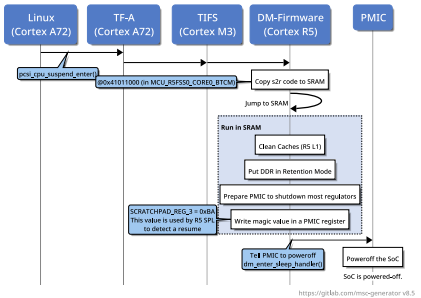
Navigating TF-A Firewall Limitations for Resume

- TIFS Firewall Restoration Limitations
  - Hardware firewall installation: configured only if the image is signed
- Memory Management Adjustment
  - Save context instead of full memory
- TF-A Cold Boot Process
  - Early check: resume or standard boot?
  - Restore context and jump to resume function (if resume)



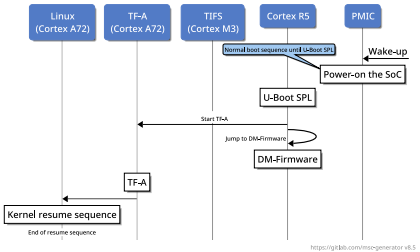
DM Firmware Adaptations for Suspend Part

- Final software executing before SoC shutdown
- Responsibilities:
  - Put DDR into self-refresh to maintain data integrity
  - Send suspend sequence to PMIC for power management and wake-up handling
- Code moved to SRAM
  - Addressing DDR self-refresh limitation
  - Guaranteeing correct execution of the code
- Storing essential data for resume
  - Magic number in PMIC register for SPL resume detection
  - TF-A resume address in DDR (removed with cold boot TF-A)



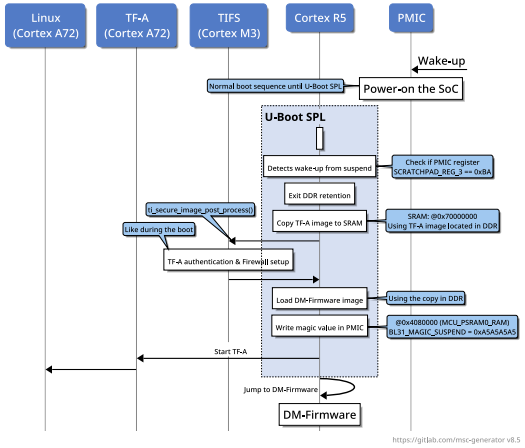
DM Firmware's Role in Resume & Linux Context Management

- No special adaptations required during the resume process
- Prerequisite: Devices managed by DM firmware deactivated by Linux during suspend
  - Ensuring smooth device activation during resume
- Linux handles context and device activation
  - Efficiently managing the resume process for DM firmware
- DM firmware boots normally after resuming, with no additional steps needed



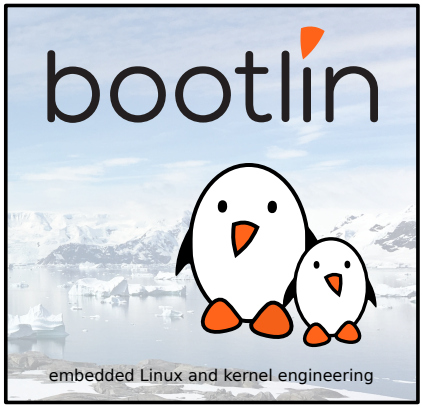
U-Boot SPL Changes & Optimizations During Resume

- U-Boot SPL changes only during resume
- SPL on R5 core involved in resume process
  - Detects resume by checking PMIC value
  - Exits DDR from retention instead of configuring
- TF-A & DM Firmware Image Reloading
  - Initially restored TF-A to SRAM from DDR
  - Now reloads TF-A image and DM firmware image
- Optimization: Load from DDR instead of storage re-read



Future challenge and improvement

Gregory CLEMENT
gregory.clement@bootlin.com



Recent Cache Management Improvement

- L3 cache disabled initially to avoid resume issues
- Issue: Unflushed cache causing outdated DDR data
- L3 cache management at SoC level
  - Access to specific registers required
  - Firewall prevented DM firmware register modification
- TIFS update needed and completed recently

Security Vulnerabilities in Current Implementation

- Challenges: Security weaknesses
- Feedback from TF-A developers
- SPL image trust issues during resume
  - TF-A community's perspective
- Current implementation concerns
  - TF-A context accessibility during resume
  - Hardware firewall applied too late

Enhancing Security with TF-A Context Encryption

- Encrypt TF-A context during suspend
  - Protecting data integrity and confidentiality
- TIFS as encryption/decryption candidate
  - New feature requirements
  - Limitations: PMIC access, DDR availability

A Collaborative Approach for Secure TF-A Decryption

- Solution: SPL and new TISCI call for decryption during resume
  - SPL's role: Exiting DDR from retention
  - TIFS's role: Decrypting TF-A after SPL's action
- Ensuring secure collaboration between SPL and TIFS

Firmware Management During Suspend to RAM

- Challenge: Handling firmware on R5 cores
  - Firmware self-management
    - Free and stop devices during suspend
    - Get and restore devices during resume
    - Save/restore SRAM in/from DDR if used
    - Notification about entering suspend
  - Alternative: Firmware unaware of system suspend
    - Device management by another component (e.g., DM firmware)
    - Restore firmware context addresses
- Linux communication
  - Restore communication channels by Linux kernel

Convergence of Suspend to RAM Support for K3 Family SoCs

- DRA821 not unique in K3 family
- Default suspend to RAM support for other K3 SoCs
  - Does not use OFF mode
- Plan: Enable OFF mode support for some SoCs
  - Maintain common code for I/O retention cases
- Goal: Convergence and shared implementation
- Ongoing effort

Questions? Suggestions? Comments?

Gregory CLEMENT
gregory.clement@bootlin.com

https://bootlin.com/pub/conferences/2024/elc/clement-s2r-dra821.pdf