Linux as the Conductor: Driving Pre-Compiled Audio DSP Kernels on C7x for Real-Time Processing

Date: 2026-03-29

Vishnu Pratap Singh

Texas Instruments

About us: TI Processors and Open source

Decades of contribution and collaboration

Ingrained culture to give back to the community

Upstream FIRST!

Focus on long term, sustainable and quality products

Upstream and opensource ecosystem in device architecture

Upstream FIRST mentality!

Agenda

- Problem Statement
- Proposed Solution
- Software Architecture
- Linux-Controlled Multi-channel Audio Offload Example
- ARM vs C7x + ARM: Real-Time Performance Comparison
- Current Status and Next Steps
- Q&A

Disclaimers

- This is a technology presentation, not product-readiness or roadmap commitment
- Opinions presented here are that of the speakers and may not reflect that of Texas Instruments Inc.
- Some Illustrative visuals generated using AI tools; performance data is based on measured results.

Problem Statement

❖ **Inefficient CPU Usage**
ARM (Linux) struggles with compute-heavy real-time workloads, leading to high CPU load.

❖ **Complex Linux ↔ DSP Integration**
Interfacing between Linux and DSP involves custom IPC, memory handling, and synchronization complexity.

❖ **Poor Scalability**
Systems become CPU-bound, limiting support for multi-channel and advanced processing pipelines.

❖ **Inefficient Data Movement**
Multiple memory copies and unmanaged buffers increase latency and overhead.

❖ **Underutilization of DSP Resources**
DSP cores exist but remain underused due to integration difficulty and lack of simple workflows.s

Existing solutions usually trade Simplicity for performance; The problem is not is dsp availability for compute – It's making DSP compute usable for Linux in simpler way.

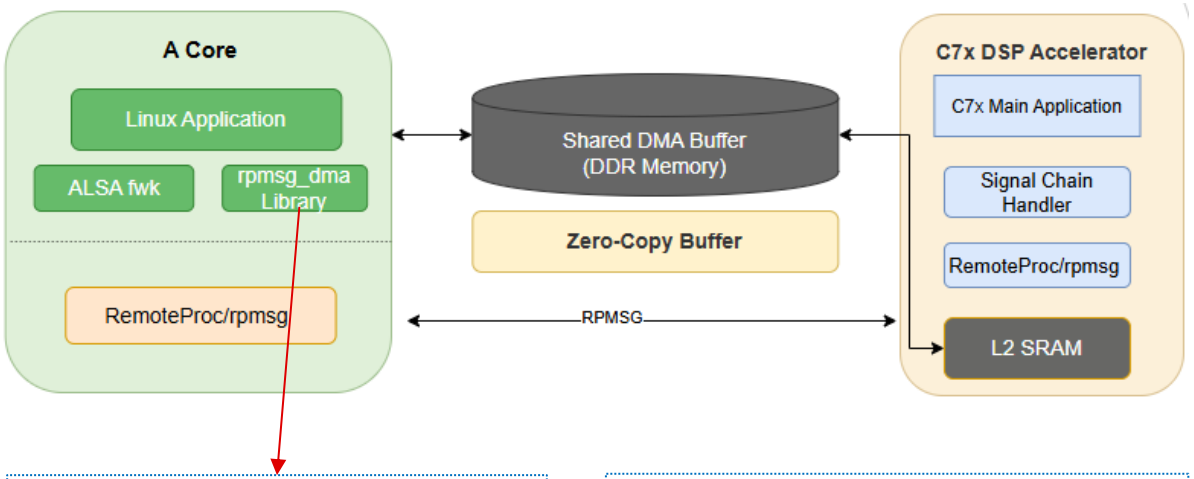
Proposed Solution: Linux Controls, DSP accelerates

Core Idea

- Keep Linux as the system orchestrator
- Use DSP as a compute accelerator
- Separate control plans(Linux) and data plane (shared memory)

Key Advantage

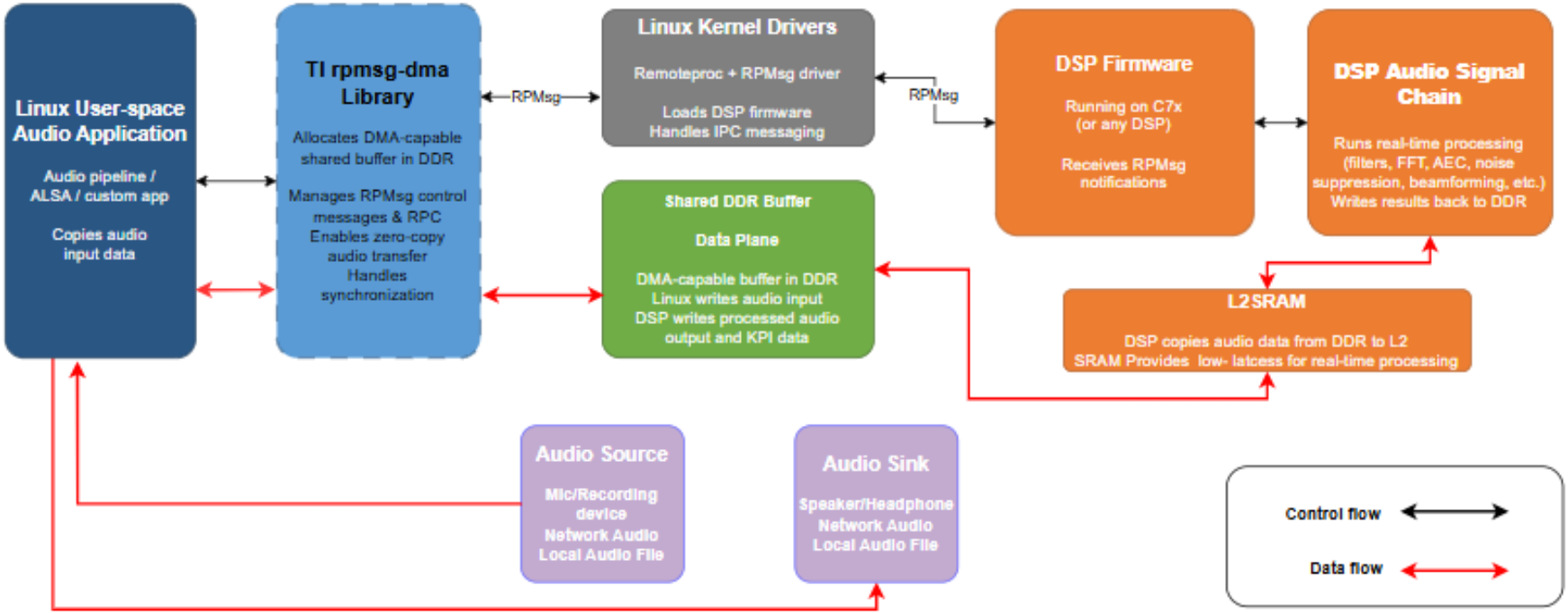
- No need to move application logic from Linux
- Reuse pre-built DSP kernels
- Zero-copy buffer model (no redundant ARM+DSP copies)
- Deterministic, low-latency processing
- Clean, reusable heterogeneous compute pattern



- Manages ARM → DSP communication (RPMsg/RPC)
- Allocates and manages shared DMA buffers (DDR)
- Handles synchronization & low-latency messaging

- Linux writes data → shared DMA buffer (DDR)
- Notifies DSP via RPMsg
- DSP copies data → L2 SRAM
- DSP executes signal chain
- Results/KPI written back → shared buffer
- DSP notifies Linux

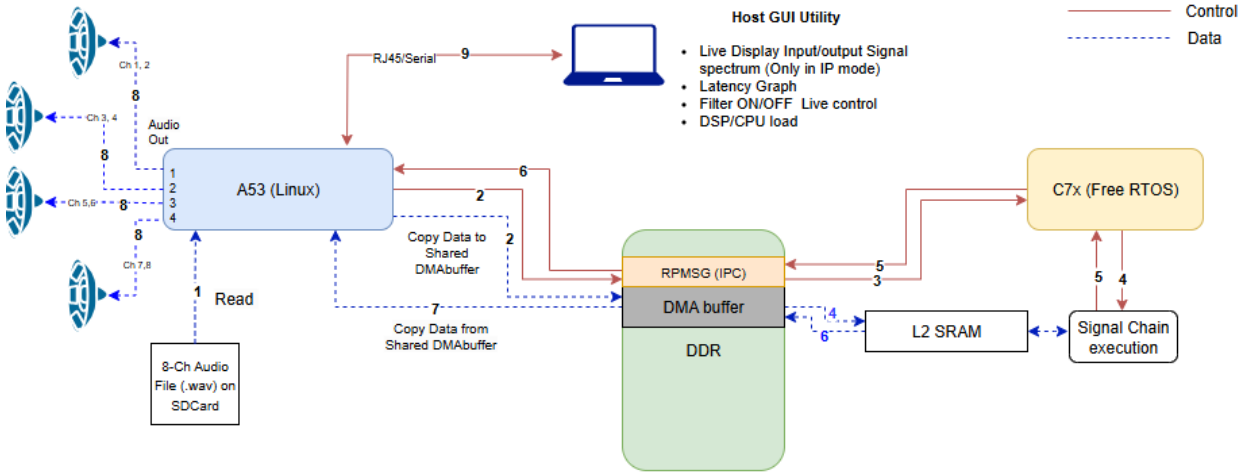
Software Architecture



Linux-Controlled Multi-channel Audio Offload Example

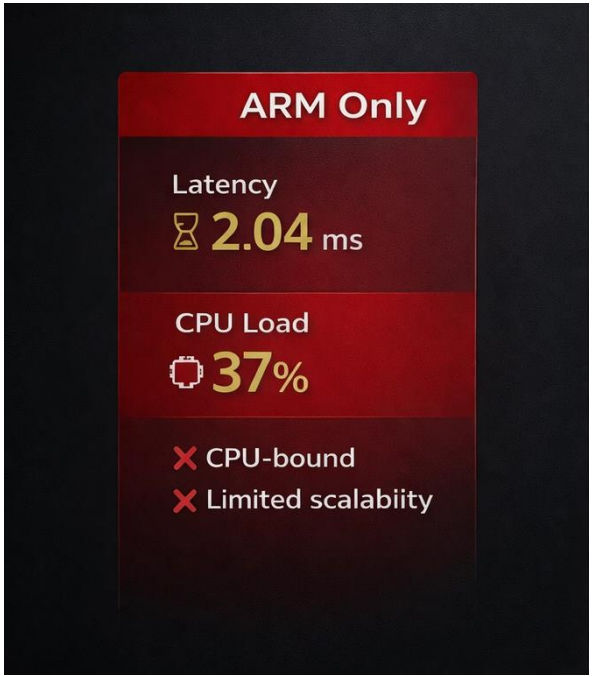
What is AM62D?

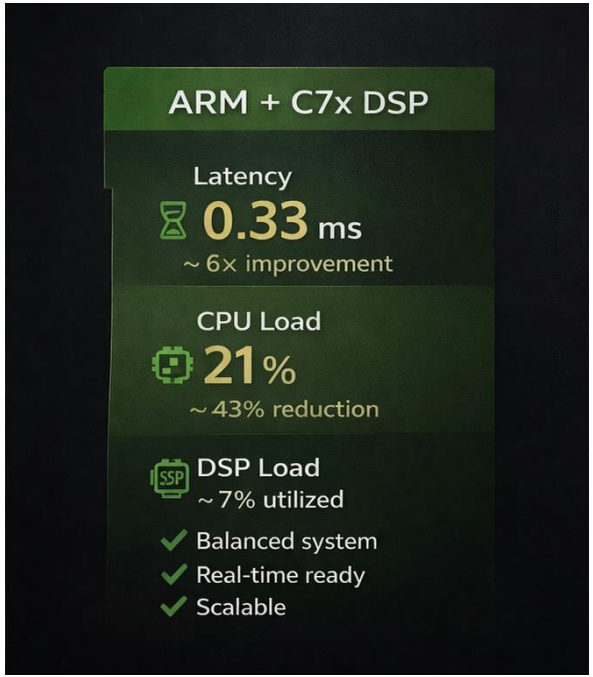
- Sitara AM62D Audio & Signal Processing SoC
- Expandable hardware for premium audio
- Quad Cortex-A53 running Linux
- C7x vector DSP for signal processing
- R5F cores for real-time control
- McASP interfaces for multichannel audio

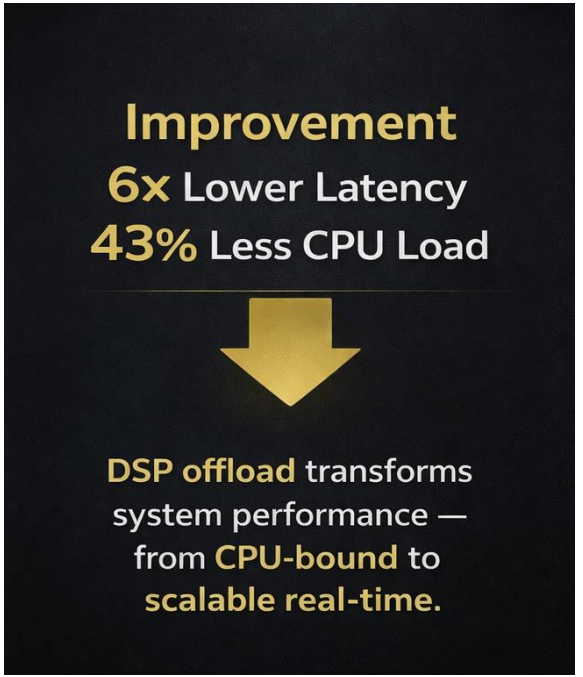


Signal chain execution on DSP

ARM vs C7x + ARM: Real-Time Performance Comparison







Signal chains are not strictly identical; results reflect practical system-level improvements with DSP offload.
Illustrative visuals generated using AI tools; performance data is based on measured results.

Current Status and Plan Ahead

Current Status

- Established Linux ↔ DSP offload architecture (ARM + C7x)
- Developed rpmsg-dma based communication framework
- Implemented shared DDR buffer model with zero-copy data flow
- Integrated with ALSA (Linux audio pipeline support)
- Achieved end-to-end real-time multichannel processing pipeline
- DSP execution optimized using L2 SRAM (low latency, deterministic)
- Validated with streaming and compute workloads

Next Steps

- Integrate with PipeWire (modern Linux audio stack)
- Extend to DSP + AI hybrid pipelines (feature extraction + inference)
- Enhance scalability (higher channels, bandwidth, workloads)
- Improve buffering & scheduling (multi-buffering, pipelining)
- Add profiling, tracing, and latency measurement tools

References

- https://www.ti.com/tool/AUDIO-AM62D-EVM#overview
- https://software-dl.ti.com/processor-sdk-linux/esd/AM62DX/latest/exports/docs/devices/AM62DX/index.html
- https://software-dl.ti.com/processor-sdk-linux/esd/AM62DX/latest/exports/docs/linux/Demo_User_Guides/index_Demos.html
- https://github.com/TexasInstruments/rpmsg-dma
- https://github.com/TexasInstruments/mcupsdk-core-k3
- https://github.com/TexasInstruments/mcupsdk-core-k3/tree/k3_main/examples/drivers/ipc/ipc_rpmsg_linux_audio_filter_offload
- https://github.com/TexasInstruments/mcupsdk-core-k3/tree/k3_main/examples/tisp/fft2d_dsp_offload/fft2d_linux_dsp_offload
- https://github.com/TexasInstruments/mcupsdk-core-k3/tree/k3_main/examples/tisp/sigchain_dsp_rt/sigchain_biquad_cascade/am62dx-evm/c75ss0-0_freertos_linux
- https://software-dl.ti.com/mcu-plus-sdk/esd/AM62DX/latest/exports/docs/api_guide_am62dx/EXAMPLES_DRIVERS_IPC_RPMESSAGE_LINUX_AUDIO_FILTER_OFFLOAD.html
- https://software-dl.ti.com/mcu-plus-sdk/esd/AM62DX/latest/exports/docs/api_guide_am62dx/EXAMPLES_TISP_FFT2D_LINUX_DSP_OFFLOAD.html

Credits and Acknowledgement

- Barath Ramesh (TI C7x DSP Team)
- Anoop Krishnan (TI AM62Dx RTOS Team)
- Texas Instruments Inc.

Q&A

- Contact Information:
  - Vishnu Pratap Singh <v-singh1@ti.com>
- Also, on IRC @ libera.chat #linux-ti

Learn more about TI products

- https://www.ti.com/linux
- https://www.ti.com/processors
- https://www.ti.com/edgeai

Why choose TI MCUs and processors?

Scalability

Our products offer scalable performance that can adapt and grow as the needs of your customers evolve.

Efficiency

We design products that extend battery life, maximize performance for every watt expended, and unlock the highest levels of system efficiency.

Affordability

We strive to make innovation accessible to all by creating cost-effective products that feature state-of-the-art technology and package designs.

Availability

Our investment in internal manufacturing capacity provides greater assurance of supply, supporting your growth for decades to come.