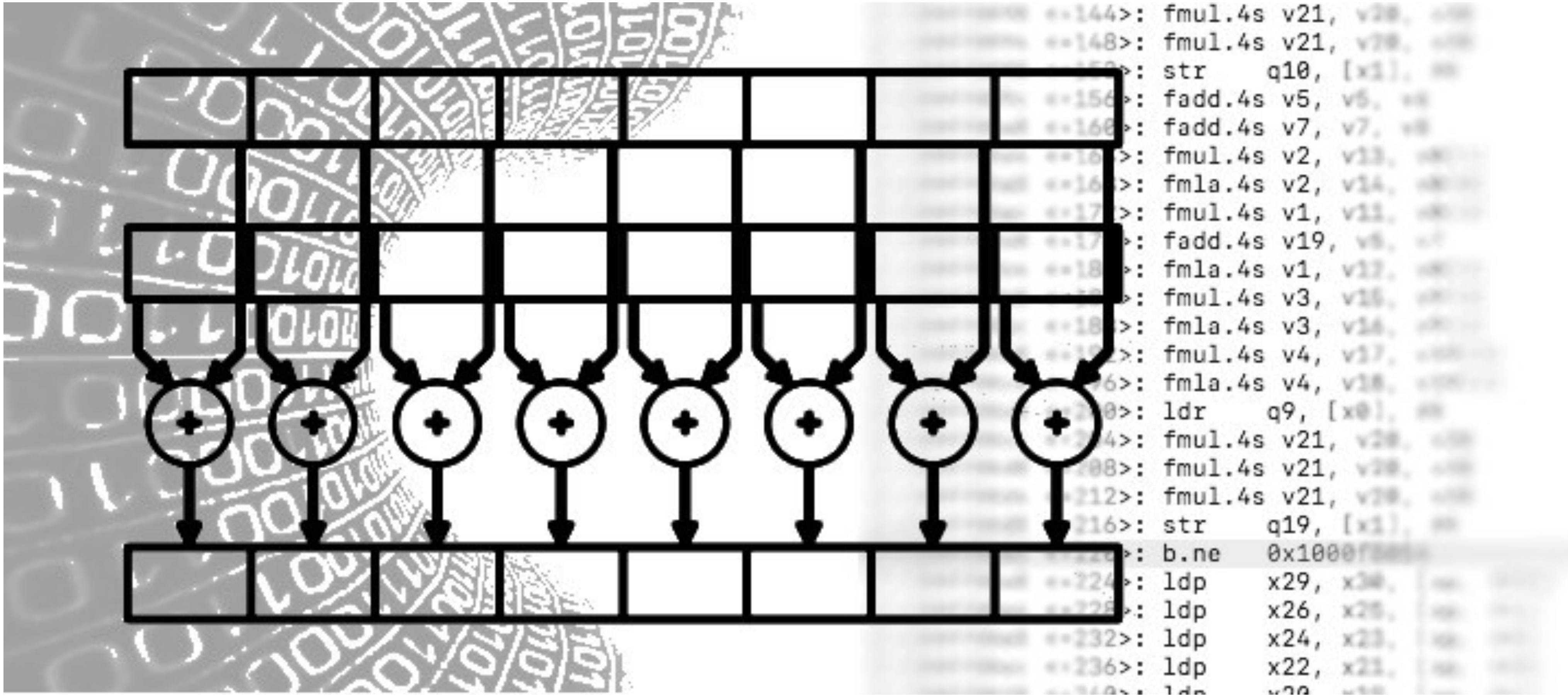
# SIMD signal processing with NEON



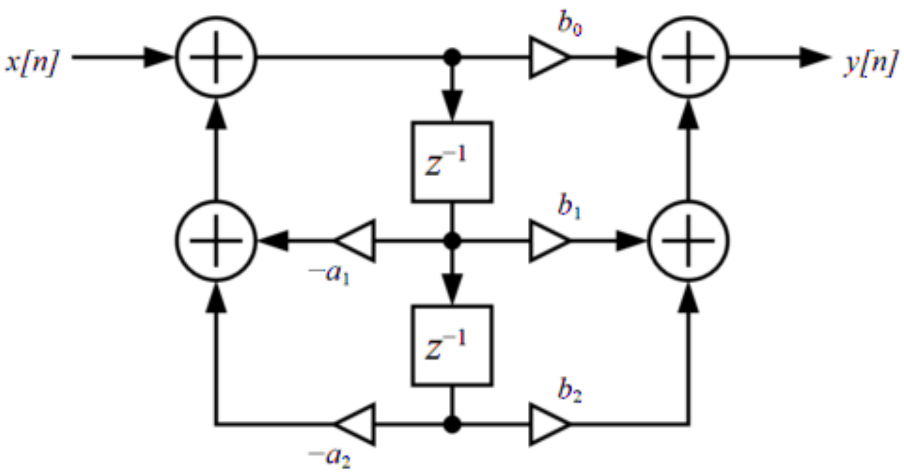

firmware-developments

# SUMMARY

In a previous article (ref.1) we demonstrated how we use CortexM0 and CortexM4 to implement signal processing algorithms where power and memory optimization are the key success criteria.

We detail here, one high-performance algorithm of our catalog.

IIR/BiQuad filters are a key building block used in digital signal processing. We describe a floating-point filter implementation using advanced SIMD NEON.
The algorithm is optimized to consume less than 4 CPU cycles per sample on ARMv8 and 7.5 cycles on ARMv7.



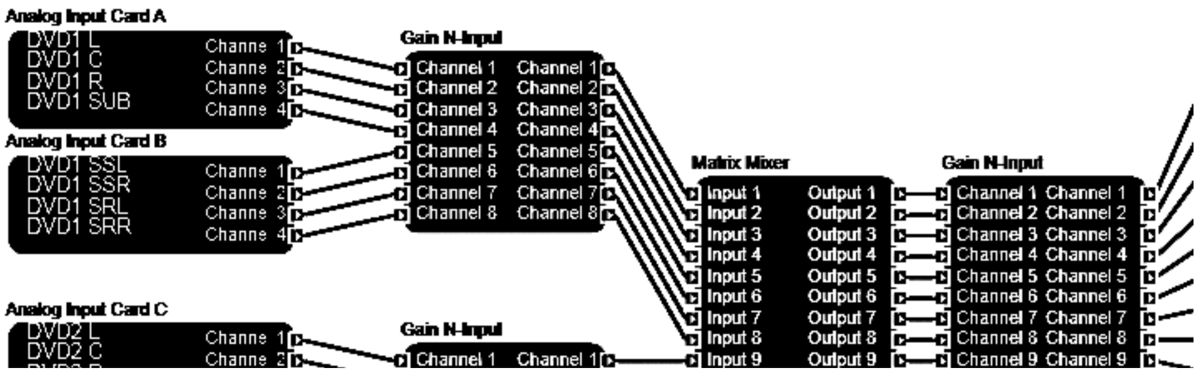

**IIR / BiQuad filter**

# PROBLEM

We want to design an **IIR / BiQuad filter**, to process non-interleaved audio samples.

IIR filtering is a challenge for the firmware designer because the pipeline depth gives a limit to the maximum data throughput, due to the recursive loops: you need to wait the computation of the recursive path before saving the next samples. The longer the pipeline depth and the longer it takes to compute the filtered audio samples.

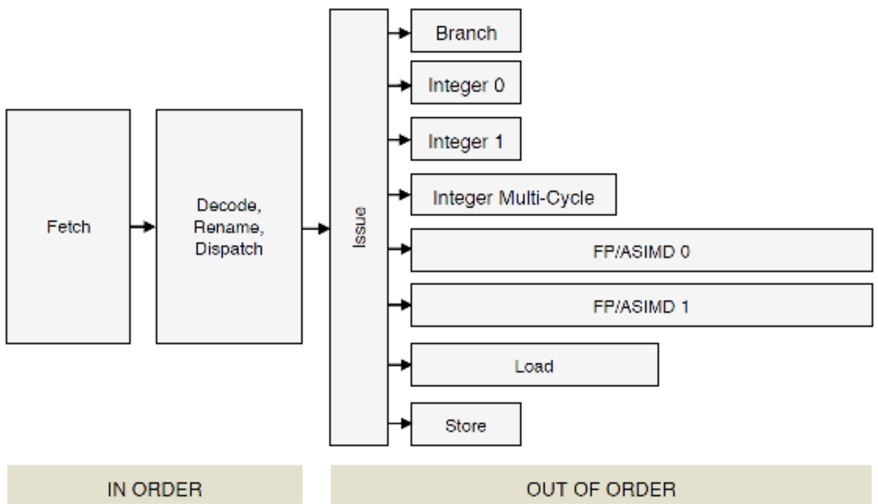
There is quite a high number of audio channels in multimedia audio systems like the one found in cars. Each channel of the original 5.1 format is processed through a cascade of IIR filters to compensate the frequency response of each loudspeakers, and to give some specific user experience depending on use-cases mixing (telephony, GPS voice, alarm, music, …). Consequently, the IIR filter must be implemented with optimized codes for power and latency reasons.




**Extract of the data paths found in high-end multimedia mixers**

**Time-to-market** is a key business success criterion for the design of consumer products. Here are some reasons to go with ARM processors and advanced SIMD NEON:

- NEON is integrated in the **trust zone** area while security is at risk with external coprocessors.
- NEON **buses are integrated** in the processor cache coherent interconnect which leads to low latencies compared to solutions using external coprocessors with ping-pong buffers for data exchanges.
- NEON implements a floating point multiply-accumulation with a **short pipeline** depth.
- NEON implements bypasses and **late forwarding** schemes between its out-of-order execution pipelines for low-latency multiply-add operations.
- The code development tools are supported by the open-source community.
- **Floating-point** accelerates signal-processing firmware development cycles compared to fixed-point arithmetic's.
- Floating-point improves the performance and the **dynamic range**, which is key for **high-resolution** audio applications.




**ARMv8 pipelines**

# SOLUTION

At Firmware-Developments we have cumulated years of expertise in firmware optimization topics, both on the problems of signal quality, standards, patents and low foot-print fixed-point implementations. **We will tune for you this IIR program which has below characteristics**:

- **Floating-point 32bits** processing with blocks of 8 samples processed in the critical loop
- **Number of cycles/sample = 3.75**: a block of 8 samples is processed in 30 cycles (ARMv8)
- **Number of cycles/sample = 7.5** (ARMv7)
- Code written in C with pieces of **ARMv7/v8** advanced SIMD NEON assembly.
- Some instructions slots are free in the critical loop to insert other integer computations without penalty.
- We have a bit-exact binary executable running on PC

# CONTACTS

Firmware Developments email : contact @ firmware-developments.com
Phone Number +33 698 846 090
Address : “Les Alcyons”, 5b Av. de l’Ilette, 06600 Antibes, France.

# REFERENCES


1 **Audio sample-rate conversion (SRC) on Cortex-M**
https://community.arm.com/b/612ad1625710445eb3e17ec0b22dcc97/posts/audio-sample-rate-conversion-src-on-cortex-m

2 **CortexA72 Software Optimization Guide.**
http://infocenter.arm.com/

3 **ARM Architecture Reference Manual ARMv8, for ARMv8-A architecture profile**
http://infocenter.arm.com/

4 **ARM CortexA Series - Programmer's Guide for ARMv8-A**
http://infocenter.arm.com/

5 **ARMv8-A Reference Manual**
http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0487a.k_10775/index.html

6 **Bit-exact simulator of the ARMv7 and ARMv8 codes of this IIR filter:**
http://firmware-developments.com/WEB/DOC/ARM/ARM_NEON/SIMULATOR/

7 **Choosing the Best Processor for your Audio DSP Application. AES137 (L. A. 2014)**
http://firmware-developments.com/WEB/DOC/REF/pd8_beckmann.pdf

8 **HARMAN Audio solutions**
http://audioarchitect.harmanpro.com/

9 **ARKAMYS Audio solutions**
https://www.arkamys.com/news-in-audio-digital-sound-processing/