

# 2025 Activity Report

RESEARCH CENTRE: Inria Lyon Centre

IN PARTNERSHIP WITH: Institut national des sciences appliquées de Lyon, Générateur de Ressources et d'Activités Musicales Exploratoires

---

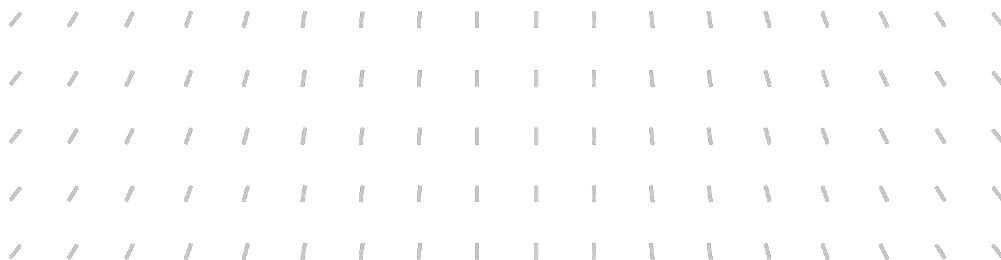
Project-Team

## EMERAUDE

EMbEdDED pROgrammable AUDio systEMs

---

*In collaboration with* Centre d'innovation en télécommunications et intégration de services



## **Project-Team EMERAUDE**

*Creation of the Project-Team: 2022 March 01*

Each year, Inria research teams publish an Activity Report presenting their work and results over the reporting period. These reports follow a common structure, with some optional sections depending on the specific team. They typically begin by outlining the overall objectives and research programme, including the main research themes, goals, and methodological approaches. They also describe the application domains targeted by the team, highlighting the scientific or societal contexts in which their work is situated. The reports then present the highlights of the year, covering major scientific achievements, software developments, or teaching contributions. When relevant, they include sections on software, platforms, and open data, detailing the tools developed and how they are shared. A substantial part is dedicated to new results, where scientific contributions are described in detail, often with subsections specifying participants and associated keywords. Finally, the Activity Report addresses funding, contracts, partnerships, and collaborations at various levels, from industrial agreements to international cooperations. It also covers dissemination and teaching activities, such as participation in scientific events, outreach, and supervision. The document concludes with a presentation of scientific production, including major publications and those produced during the year.

## Keywords

### Computer sciences and digital sciences

- A1.1.2. – Hardware accelerators (GPGPU, FPGA, etc.)
- A2.2. – Compilation
- A5.7.1. – Sound
- A5.7.2. – Music
- A5.7.5. – Synthesis
- A5.9. – Signal processing
- A8.10. – Computer arithmetic

### Other research topics and application domains

- B6.6. – Embedded systems
- B9.2.1. – Music, sound

## Contents

<b>Project-Team EMERAUDE</b>	<b>1</b>
<b>1 Team members, visitors, external collaborators</b>	<b>5</b>
<b>2 Overall objectives</b>	<b>6</b>
<b>3 Research program</b>	<b>9</b>
3.1 Embedded Audio Systems and FPGAs	10
3.2 Optimization of Arithmetic for FPGAs	11
3.3 The Faust Programming Language and its Ecosystem	13
<b>4 Application domains</b>	<b>15</b>
4.1 Spatial active noise control	15
4.2 Virtual acoustics/spatial audio	15
4.3 Industrial acoustics	16
4.4 Medicine/sonification	16
4.5 Low-latency audio effect processors and synthesizers	16
4.6 Digital luthiery	16
<b>5 Highlights of the year</b>	<b>17</b>
5.1 JIMLAC-25	17
5.2 Awards	17
<b>6 Latest software developments, platforms, open data</b>	<b>17</b>
6.1 Latest software developments	17
6.1.1 FloPoCo	17
6.1.2 Syfala	18
6.1.3 FAUST	18
<b>7 New results</b>	<b>18</b>
7.1 Interaction within the team	18
7.2 Immersive Audio	19
7.2.1 Distributing spatial audio	19
7.2.2 Real-Time Auralization on FPGA	20
7.2.3 Auralization of Paleoacoustics landscapes in Chauvet Cave	20
7.2.4 The Space Bar, an Embedded WFS Sound System	20
7.3 Acceleration of AI inference	21
7.3.1 HAtorch: Hardware-aware quantization-aware training for CNNs	21
7.3.2 Towards frugality in Natural Language Processing classification models	22
7.4 Optimization of arithmetic cores	23
7.4.1 Double-Word Decomposition in a Combined FP16, BF16 and FP32 Dot Product Add Operator	23
7.4.2 Reconfigurable constant multiplication	23
7.4.3 Constraint programming models for multiple constant multiplication	24
7.4.4 Optimization for other application domains	25
7.5 From FLoPoCo to MLIR dialects and flows	25
7.6 The FAUST Programming Language and its Ecosystem	26
7.6.1 The FAUST programming language	27
7.6.2 The FAUST compiler	27
7.6.3 The FAUST ecosystem	29
<b>8 Bilateral contracts and grants with industry</b>	<b>31</b>
8.1 Bilateral contracts with industry	31

<b>9 Partnerships and cooperations</b>	<b>31</b>
9.1 International initiatives	31
9.1.1 Associate Teams in the framework of an Inria International Lab or in the framework of an Inria International Program	31
9.2 National initiatives	32
9.2.1 ANR DISTrib	32
9.2.2 PEPR HoliGrail	32
<b>10 Dissemination</b>	<b>33</b>
10.1 Promoting scientific activities	33
10.1.1 Scientific events: organisation	33
10.1.2 Scientific events: selection	33
10.1.3 Journal	34
10.1.4 Invited talks	34
10.1.5 Leadership within the scientific community	34
10.2 Teaching - Supervision - Juries - Educational and pedagogical outreach	34
10.2.1 Juries	35
10.3 Popularization	35
10.3.1 Productions (articles, videos, podcasts, serious games, ...)	35
10.3.2 Participation in Live events	35
10.3.3 Others science outreach relevant activities	36
<b>11 Scientific production</b>	<b>36</b>
11.1 Major publications	36
11.2 Publications of the year	37
11.3 Cited publications	40

# 1 Team members, visitors, external collaborators

## Research Scientists

- Stéphane Letz [GRAMÉ]
- Romain Michon [INRIA, ISFP, HDR]
- Anastasia Volkova [INRIA, Researcher]

## Faculty Members

- Tanguy Risset [Team leader, INSA LYON, Professor Delegation, HDR]
- Christine Solnon [INSA LYON, Professor, HDR]
- Florent de Dinechin [INSA LYON, Professor, HDR]

## Post-Doctoral Fellows

- Aurélien Delage [INSA LYON, Post-Doctoral Fellow, until Aug 2025]
- Romain Fontaine [INSA LYON, from Sep 2025]
- Romain Fontaine [INSA LYON, Post-Doctoral Fellow, until Aug 2025]
- Louis Ledoux [INSA LYON, Post-Doctoral Fellow]
- Xiao Peng [INSAVALOR, Post-Doctoral Fellow, until Mar 2025]

## PhD Students

- Bastien Barbe [INSA LYON]
- Romain Bouarah [INSA LYON]
- Theo Cantaloube [INSA LYON, from Oct 2025]
- Eric Chen [THALES, CIFRE]
- Oregane Desrentes [KALRAY, CIFRE, until Sep 2025]
- Adrien Pichon [UBS]
- Benjamin Quiedeville [GRAMÉ, CIFRE]
- Florian Rascoussier [IMT ATLANTIQUE]
- Thomas Rushton [INRIA]
- Clemens Wegener [INRIA, from Dec 2025]

## Technical Staff

- Pierre Cochard [INSA LYON]
- Yann Orlarey [INRIA, Engineer, until Sep 2025]

## Interns and Apprentices

- Mohammad Ali [GRAME, from Sep 2025]
- Elise Bachet [INSA LYON, from Jun 2025 until Aug 2025]
- William Barran [INSA LYON, from Jun 2025 until Aug 2025]
- Bastien Candela Marty [INSA LYON, Intern, from Jun 2025 until Sep 2025]
- Theo Cantaloube [INRIA, Intern, from Mar 2025 until Jul 2025]
- Adam Guglielmino [INRIA, Intern, from Sep 2025]
- Remi Guillotte [INRIA, Intern, from Mar 2025 until Sep 2025]
- Simon Jacquin [INRIA, Intern, from Mar 2025 until Sep 2025]
- Leslie Mendoza [INSA LYON, from Apr 2025 until Jul 2025]

## Administrative Assistants

- Sylvie Boyer [INRIA]
- Anouchka Ronceray [INRIA]
- Linda Soumari [INSA LYON]

## External Collaborator

- Yann Orlarey [POLE EMPLOI, from Oct 2025]

## 2 Overall objectives

The goal of the Emeraude project-team<sup>1</sup> is to combine the multidisciplinary skills of CITI laboratory and Grame-CNCM to foster the development of new programming tools and signal processing techniques for embedded audio systems.

Grame-CNCM<sup>2</sup> is a National Center for Musical Creation (CNCM<sup>3</sup>) hosting a research team specialized in music technology. Grame is also the inventor of the FAUST programming language,<sup>4</sup> which has met great success in the audio processing community. The skills in compilation, embedded systems, and FPGAs of former Inria Socrate team members, as well as the experience acquired in signal processing is also useful for research in audio and acoustic signal processing.

Embedded programmable audio systems are ubiquitously used in our day-to-day life. Whether it's in our headphones or our car to carry out active noise cancellation, in virtual home assistants (e.g., Alexa, Google Home, etc.), or in modern musical instruments and sound systems, they are everywhere. Real-time audio processing is known to be relatively computationally expensive, but progress in processor architectures in recent years – including microcontrollers, microprocessors, Digital Signal Processors (DSPs), Graphics Processing Unit (GPUs), etc. – have made computing power much more affordable. The generalization of hardware floating point support, and the availability of high-level IDEs (Integrated Development Environments) for these new architectures has made them accessible to audio programmers.

Programming embedded audio systems requires specific skills: Digital Signal Processing (DSP), low-level C/C++ programming, and a deep understanding of system architecture. Few engineers (whether they are on the DSP or the programming side) fully master all these domains, and even fewer people in the maker community. The scientific credo of the Emeraude Inria-Insa joint project-team is that **Domain Specific**

<sup>1</sup>Throughout the document, we refer to Emeraude as “the Emeraude project-team,” being aware that the official denomination should be “Insa-Inria joint project-team.”

<sup>2</sup>[www.grame.fr](http://www.grame.fr)

<sup>3</sup>Centre National de Création Musicale (CNCM) is a Label of the Ministry of Culture. Grame is the first CNCM in France.

<sup>4</sup>[faust.grame.fr](http://faust.grame.fr)

**Languages (DSLs) are a major technical evolution to enable audio programming on emerging embedded systems.** There currently exists a few software solutions addressing audio programming such as libpd [46] or the SOUL programming language,<sup>5</sup> but none of them is as generic and as universal as FAUST [88], which has been developed at Grame for more than 15 years.

Emeraude uses the FAUST programming language as the main platform for experimenting fundamental research. FAUST [88] is a DSL for real-time audio signal processing. A screenshot of the FAUST IDE is shown in Fig. 1. FAUST is widely used for audio plugin design (i.e., effects and synthesizers), DSP research, mobile and web app design, etc. The success of FAUST is due to its natural data-flow paradigm and on a compiler “translating” DSP specifications written in FAUST into a wide range of lower-level languages (e.g., C, C++, Rust, Java, LLVM bitcode, WebAssembly, etc.). Thanks to its highly re-targetable compilation flow, generated DSP objects can be embedded into template programs (wrappers) used to turn a FAUST program into a specific ready-to-use object (e.g., standalone, plug-in, smartphone app, webpage, etc.).

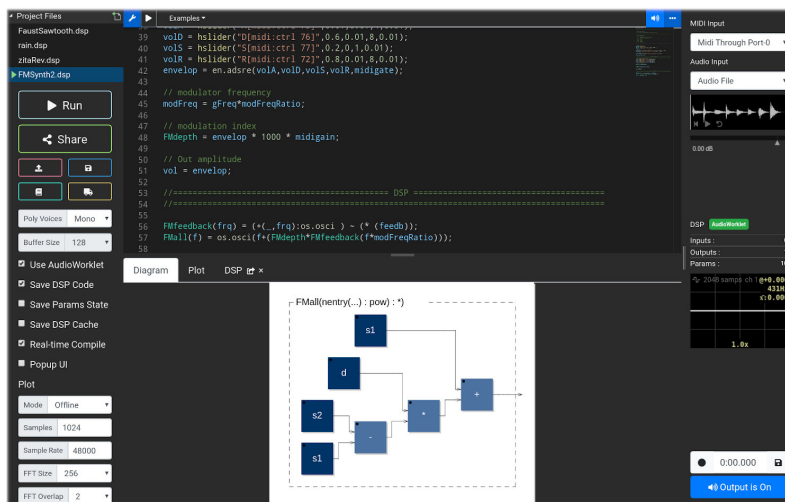


Figure 1: The FAUST Web IDE allowing for the compilation of FAUST programs on any machines without having to install any particular tool.

While FAUST was not originally designed with embedded audio systems in mind, its development took a significant turn in that direction by targeting an increasingly large number of hardware platforms such as the Teensy<sup>6</sup> [81] and the ESP-32 microcontrollers<sup>7</sup> [82], the SHARC Audio Module DSP,<sup>8</sup> the BELA,<sup>9</sup> the ELK,<sup>10</sup> etc. Since FAUST can generate various types of standalone programs for Linux, it can also target most embedded Linux systems such as the Raspberry Pi or the BeagleBone for real-time audio signal processing applications. This recent availability of FAUST compilation on tiny embedded systems and micro-controllers in particular opens the door to the creation of innovative audio objects. Fig. 2 shows the Gramophone, a device designed by the Grame team and that is used in schools to teach basic science concepts to children.

FAUST is now a well-established language in the audio DSP community. It is used both in academia for teaching in prestigious institutions such as Stanford University,<sup>11</sup> Aalborg University, the University of Michigan, and in the industry (e.g., moForte Inc.,<sup>12</sup> ExpressiveE). FAUST is also used as a prototyping tool at Korg, Apple, Google, Tesla, etc.

While embedded audio systems are already widespread, many limitations remain, especially for real-time applications where *latency* plays a crucial role. For instance, efficient active control of sound where audio

<sup>5</sup>[soul-lang](https://soul-lang.com/)

<sup>6</sup>[pjrc.com/teensy](https://pjrc.com/teensy)

<sup>7</sup>[faustdoc.grame.fr/tutorials/esp32/](https://faustdoc.grame.fr/tutorials/esp32/)

<sup>8</sup>[wiki.analog.com/resources/tools-software/sharc-audio-module/faust](https://wiki.analog.com/resources/tools-software/sharc-audio-module/faust)

<sup>9</sup>[bela.io](https://bela.io)

<sup>10</sup>[elk.audio](https://elk.audio)

<sup>11</sup>FAUST plays a central role in the curriculum at Stanford University’s Center for Computer Research in Music and Acoustics where it is used to teach signal processing, physical modeling, physical interaction design, etc.

<sup>12</sup>[www.moforte.com/faustandfurious](https://www.moforte.com/faustandfurious)

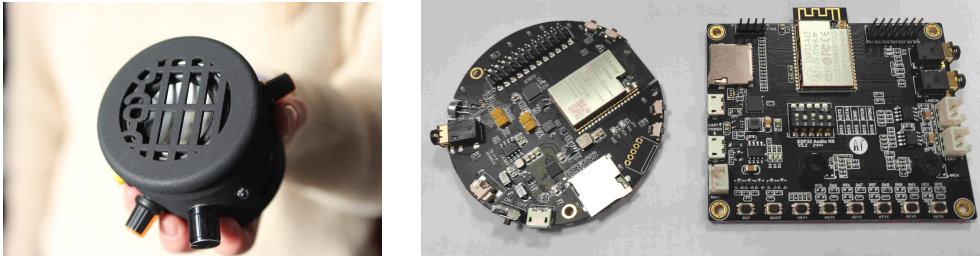


Figure 2: The *Gramophone* is a speaker/musical instrument programmable with FAUST designed to facilitate the teaching of programming, maths, and physics in middle and high schools. A picture of the board used inside it (an ESP-32 microcontroller programmed directly with a FAUST program) can be seen on the right-hand-side of the figure.

processing should be faster than the propagation of acoustical waves [60], digital musical instruments playability [73], digital audio effects, etc. cannot be deployed on lightweight systems. While latency can be potentially reduced on “standard” computing platforms such as personal computers, going under the “one millisecond threshold” is usually impossible because of audio samples buffering induced by software audio drivers.

Up to now, most of the research efforts on audio signal processing have been focused on throughput and computing power, leaving aside ultra-low latency as it seemed inaccessible on software platforms. We believe that **enabling ultra-low latency for audio application will open a wide range of new domains of application** from active acoustic control to new musical instruments (see Fig. 3, “stolen” from the ANR FAST project which started in 2021).

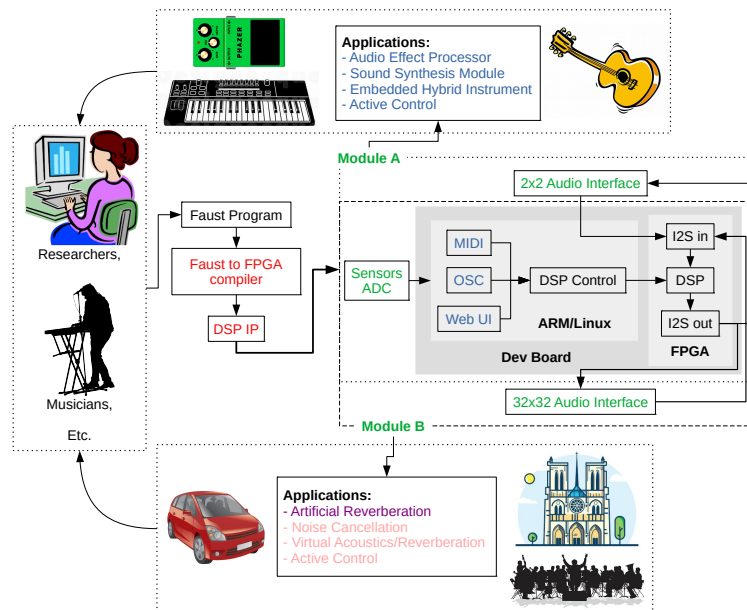


Figure 3: Example of target applications for ultra-low latency audio processing on FPGA: module A and module B are two possible “products” based on the same faust2FPGA compilation flow.

FPGAs (Field Programmable Gate Arrays) can help solve current limitations of traditional computing platforms used for musical and artistic applications, especially in terms of audio latency. FPGAs are known for their high computational capabilities [50, 89] and their very low-latency performances [107]. They also provide a large number of GPIOs (General Purpose Inputs and Outputs) which can be exploited to implement modern real-time multi-channel processing algorithms (e.g., sound field capture using a very large number of

digital microphones [95], active sound control over a large spatial region [110], etc.).

But FPGAs remain extremely complex to program, even with state-of-the-art high-level tools,<sup>13</sup> making them largely inaccessible to DSP researchers, musicians, digital artists, and maker communities. There are currently only a few examples of professional FPGA-based real-time audio DSP systems (i.e., Antelope Audio,<sup>14</sup> Korora Audio<sup>15</sup>) and in these applications, FPGAs are dedicated to a specific task and not exploited as user-programmable devices.

Emeraude provides a combination of skills that is unique in the world: audio signal processing, compilation, high-level synthesis, computer arithmetic, FPGA programming, acoustics, and embedded system design. This gives a hint on what initially motivated the creation of Emeraude: a compiler from FAUST to FPGA as considered in the SyFaLa project<sup>16</sup> enabling very low latency processing (less than 100  $\mu$ s, or equivalently between 1 and 5 sample latency).

The objective of the research axes described in the next section is to deeply understand and enable new compilation flows for audio signal processing.

### 3 Research program

The Emeraude project team was officially created in March 2022, though we had been working together for two years prior. At that time, we had decided to organize the team around four research axes:

1. Ultra-Low Audio Latency on FPGA
2. Advanced Arithmetics for Digital Audio
3. Digital Audio Signal Processing
4. Language, Compilation, Deployment and Interfaces for Audio Signal Processing

However, it soon became clear that it was challenging to separate axis 3 (Digital Audio Signal Processing) from axes 1 and 4. Specifically, we have a very active research group in embedded audio systems and FPGAs (with four permanent members) and a strong group in FPGA-based arithmetic (with two permanent members). Within the embedded audio systems group, certain topics are directly concerned with FAUST, the language design, the compiler, and the ecosystem. Therefore, the three research focuses presented in this document represent the most effective way to organize the team going forward in a balanced manner:

1. Embedded Audio Systems and FPGAs
2. Optimization of Arithmetic for FPGAs
3. The FAUST Programming Language and its Ecosystem

The recent arrival of Christine Solnon and her students (four people as of September 2024) has been in planning for six months and impacts the team's structure. Christine Solnon is a renowned researcher in graph algorithms and constraint programming and also has a solid background in optimization more broadly. Many of the problems we study give rise to unique optimization challenges. For example, optimizing the bit width in FIR or IIR filters can be framed as a complex integer linear programming problem, while certain compilation problems addressed in the FAUST compiler can be expressed as graph algorithms. More generally, optimization has applications in numerous scientific fields, and we intend to establish it as an important axis for the Emeraude team in the next years.

---

<sup>13</sup>FPGAs are configured/programmed using a Hardware Description Language (HDL) such as VHDL or Verilog. The learning curve and the electrical engineering skills required to master these types of environments make them out of reach to the real-time audio DSP community. Solutions exist to program FPGAs at a higher level (i.e., LabVIEW: [www.ni.com/fr-fr/shop/labview.html](http://www.ni.com/fr-fr/shop/labview.html), Vivado HLS: [www.xilinx.com/HLS](http://www.xilinx.com/HLS) for instance), but none of them is specifically dedicated nor adapted to real-time audio DSP. On the contrary, high-level tools tend to add abstraction layers which translate to buffers, hence latency.

<sup>14</sup>[en.antelopeaudio.com](http://en.antelopeaudio.com)

<sup>15</sup>[www.kororaudio.com](http://www.kororaudio.com)

<sup>16</sup>The SyFaLa project (*Synthétiseur Faible Latence sur FPGA – faust.grame.fr/syfala*) initiated the idea of VHDL compilation from FAUST by coupling the FAUST compiler and high-level synthesis tools of FPGA vendors.

### 3.1 Embedded Audio Systems and FPGAs

**Participants:** Florent de Dinechin, Stéphane Letz, Romain Michon, Yann Orlarey, Tanguy Risset.

The main objective of this research axis is to enable the construction of audio systems reacting with a latency smaller than (or at least comparable to) the duration of a single audio sample.

Low-latency digital audio processing might seem easy: computer systems operate at GHz frequencies whereas audible sound stops at about 20 kHz (high-resolution sound processing means 192 kHz sample frequency; CD-quality is 44.1 kHz). Concerning sound propagation, electronic data may be transmitted at speeds close to the speed of light while sound travels one million times slower. Still, achieving ultra-low latency remains a huge technical challenge. For the main applications of mass-produced audio devices (mostly sound playback and telephony), a latency of a thousand audio cycles translates to an audible delay that is barely noticeable. However, for the applications envisioned in Emeraude, sound must be captured, processed, and emitted with sub-millisecond latencies.

For that, we need to provide a real compilation flow from high-level audio DSP programs to FPGA IPs. Our proposal is to target a new FAUST architecture backend for FPGA-based platforms as depicted in Fig. 4. One of the challenges here is the optimization of the module generated by FAUST. The real breakthrough will be obtained with the use of two recent technologies in the FAUST compilation workflow: (i) *High Level Synthesis* (HLS) for compiling FAUST programs to VHDL and (ii) *fixed-point support* in the code generated by the FAUST compiler, building on the expertise developed at CITI around the FloPoCo project (and studied in next research axis: §3.2).

In Audio, sampling rate is between 20kHz and 200kHz. The sampling rate has of course an impact on achievable latency: at 48kHz, one sample arrives every  $20\mu\text{s}$  and the achievable latency is limited to one sample because of the audio codec (ADC/DAC) serial protocol. However, what is called “low latency” in current systems is usually close to 1ms (50 samples at 48kHz). Various systems, both in the industry and in academia, have been targeting low audio latency through the use of different hardware solutions. The most affordable ones are embedded Linux systems enhanced with dedicated audio hardware. They run audio signal processing tasks outside of the operating system. The BELA [79] and the Elk,<sup>17</sup> which belong to this category, can achieve relatively low latency with buffer sizes as low as 8 samples.

Microcontrollers have been used more and more in recent years for sound synthesis and processing because of their increasing power. The Teensy [81] and the ESP32 [82] are good examples of such systems. When programmed “bare-metal” (i.e., without an OS), their latency can be similar to that of dedicated/specialized embedded Linux systems (buffer size of 8 samples as well).

Digital Signal Processors (DSPs) can target even lower latency with buffer sizes as low as 4 samples and provide tremendous amounts of computational power for signal processing applications. Their programming needs specific developer tools, making them less accessible than the other types of systems mentioned in this section. Additionally, many of them do not provide native support for floating-points computations, further increasing the complexity of their programming. The Analog Devices SHARC Processor<sup>18</sup> is a leader on the market which can be used as a prototyping system through the SHARC Audio Module. It also provides an official FAUST support.

The only way to take audio latency one step further down is to use FPGAs, which is what we plan to do in this research axis.

Programming FPGAs is usually done with a hardware description language (VHDL or Verilog). Developing a VHDL IP<sup>19</sup> is extremely time consuming. Hence, FPGA programmers have two possibilities: re-using existing IPs and assembling them to compose a circuit solving their problem (as proposed by LABVIEW<sup>20</sup>), or using High-Level Synthesis to *compile* a VHDL specification from a higher-level description.

High Level Synthesis (HLS) [86] has been referred to for decades as *the* mean to enable fast and safe circuit design for programmers. However, the design space offered to a hardware designer is so huge that

<sup>17</sup>[elk.audio](http://elk.audio)

<sup>18</sup>[www.analog.com/en/products/processors-dsp/dsp/sharc.html](http://www.analog.com/en/products/processors-dsp/dsp/sharc.html)

<sup>19</sup>IP stands for Intellectual Property, it is the common denomination for *hardware library*, i.e., a circuit design that can be re-used as for instance a software library.

<sup>20</sup>[www.ni.com/fr-fr/shop/labview.html](http://www.ni.com/fr-fr/shop/labview.html)

no automatic tool is able to capture all the constraints and come up with the *optimal* solution (which does not exist anyway since multiple objectives are to be optimized). Many HLS tools have been proposed (i.e., Pico [96], CatapultC [43], Gaut [99], to cite a few) dedicated to specific target application domains. Most of the existing tools start from a high-level representation that is based on a programming language (i.e., C, C++, or Python) which is *instrumented* using pragmas to guide the HLS process.

Using HLS today still requires very specific skills [68] to write a source description that is correctly processed by the tools, but we believe that this technology has reached a certain level of maturity and can now be foreseen as a valuable tool for audio designers.

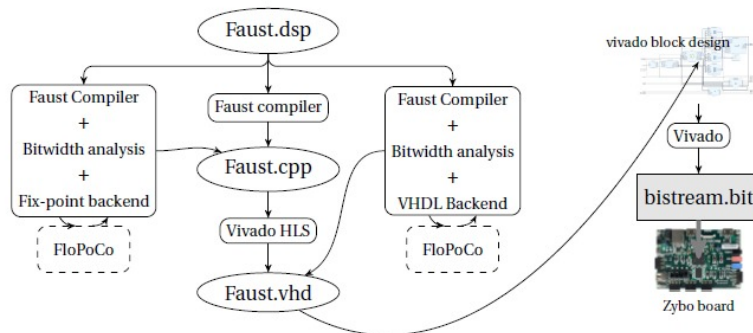


Figure 4: The complete faust2FPGA flow targeted by this research axis. Different possible compilation flows for generating VHDL from a FAUST program will be studied.

Another goal is to adapt the different design flows to target high-performance FPGA boards, such as the *Genesys ZU* based on a Zynq Ultrascale FPGA for instance. These new targets are used for the compute-bound studied algorithms. High computing power implies the introduction of parallelization techniques in the compilation flow (either using the HLS process or by direct VHDL generation from FAUST). This research direction might require the parallelization techniques (Polyhedral tools in particular) developed within Inria in particular (e.g., CASH, Taran, CAMUS, CORSE, etc.).

The main outcome of this research axis, namely the new open-source compilation flow from FAUST to FPGA is useful in many contexts: for musicians, acoustic engineers or mechanical vibration engineers. In order to convince these people to use it, we are prototyping a large number of audio treatments (e.g., filters, reverb effects, etc.) and study the resulting performances – in terms of latency and computing power – depending of the configuration chosen for the flow.

### 3.2 Optimization of Arithmetic for FPGAs

**Participants:** Florent de Dinechin, Anastasia Volkova, Christine Solnon, Yann Orlarey.

In this research axis, Emerald builds upon the expertise developed in Socrate in application-specific arithmetic. Florent de Dinechin is an expert of computer arithmetics in general (including floating-point [85] and alternatives [55, 103]) but also in arithmetics for FPGAs, in particular with the FloPoCo project [54]. Anastasia Volkova specializes in error analysis and optimization of computer arithmetic, with a focus on fixed-point operator design for digital signal processing and machine learning. Christine Solnon, as an expert in graph algorithms and constraint programming, brings a unique insight into efficient design-space exploration and optimization of mathematical models. Their expertise is helping us addressing challenges related to low-latency digital audio by combining complementary approaches: compilation of digital audio to fixed-point arithmetic, an arithmetic-centered approach to digital filter design, and the scheduling and tiling problems. In these three directions, audio applications fuel research that has an impact well beyond audio.

**Audio-to-fixed easier than float-to-fixed** In audio processing, we know that the inputs and outputs are fixed-point data, and we also have a lot of domain knowledge about audio physics. This gives serious hope that FAUST audio can be compiled directly to fixed-point. This is a requirement for FPGAs, but it will also reduce the latency and power consumption on software targets if we can use their integer units. It will also enable the compilation of FAUST to ultra-low-power microcontrollers without floating-point hardware.

“Domain-specific” is the key word here making us confident that a problem that is generally intractable (float-to-fixed conversion) can be addressed with little or no modification to a FAUST program. The challenge here is to keep this additional work so high-level and sound-related that it is not a burden for a musician or a sound engineer. A central objective is that FAUST programmers should not need to become fixed-point experts. They should actually not be anymore aware of the underlying arithmetic than they currently are with floating-point. Being high-level is a key reason for the success of FAUST.

**Automated error analysis for hardware computing just right** The main issue is to understand how arithmetic errors propagate, are amplified, are accumulated, etc. in a computation and in a circuit. This is called *error analysis*. Then a general technique [58] is to add enough bits to the right of internal fixed-point formats so that errors accumulate in these bits and the overall error accumulation does not hinder the final quality of the result. Error analysis is also managed by a worst-case combination, but here there is nothing implicit or hidden. This is therefore a comparatively well understood problem, and there is no reason to believe it cannot be fully automated in a compiler that is already able to derive the format information, building on the experience accumulated when designing complex FloPoCo operators [57, 56, 47, 102, 109].

**Digital filters as arithmetic objects** Digital filters are essential components of everyday electronics like radios, mobile phones, etc., but also in audio systems of course. Their design is a core topic in digital signal processing and control theory, one that has received significant research interest for the better part of the last half century. A lot of effort has gone into constructing flexible filter design methods. For designing software-based digital filters with floating-point coefficients, there are many powerful approaches that are relatively easy to use by the filter designer (all the more as they rely on over-dimensioned floating-point operators). When designing hardware, things are not that simple for several reasons:

- algorithms developed for software-implemented filters cannot be transferred directly to hardware: what is a constraint in software (e.g., “use a 32-bit fixed-point format”) becomes a degree of freedom in hardware design (“What is the smallest fixed-point format that can be used?”);
- another degree of freedom comes from different available realization techniques to implement the arithmetic itself, for instance the construction of multipliers by constants.

A consequence is that popular tools, such as the popular `fdatool` (filter design and analysis tool) from Matlab’s Signal Processing toolbox, offer a complex interface, requiring a tedious hand-tuning process, and expect some domain expertise. Such tools input a frequency response, and decompose the filter implementation problem in three steps: 1/ the filter design (FD) step consists in finding a filter with ideal (high precision) coefficients that adheres to the frequency response; 2/ the quantization (Q) step converts the obtained coefficients to hardware-friendly fixed-point values; 3/ the implementation (I) step generates a valid hardware description (e.g., a VHDL or Verilog description) using the quantized coefficients.

The objective of this research axis is to offer an optimal solution to the global FD + Q + I problem. Optimal techniques exist for each of the FD, Q and I steps in isolation. The combination of the FD & Q steps have been studied since the 1960’s [69], and can even be regarded as solved for certain practical instances of fixed-point Finite Impulse Response (FIR) design [70]. A large body of work also exists for the I step, with recent optimal approaches [40, 71, 72]. However, these approaches are only optimal for a given set of coefficients, and therefore strongly depend on the FD and Q steps.

**Arithmetic-oriented scheduling and tiling for low-latency audio** Finally, we also want to formally insert arithmetic considerations in the global problem of distributing a very heavy computation between space (we have up to several thousands multipliers in an FPGA, and many more if we multiply by a constant) and time (we have thousands of FPGA cycles within one audio cycle). These are well-researched compilation issues, called the scheduling and tiling problems. There is local expertise in Lyon (in particular in the CASH

team and its spin-off XtremLogic<sup>21</sup>) who have worked on these problems for FPGAs. However, scheduling and tiling techniques so far consider each operation as having a standard, constant cost (e.g., multiplication costs  $c_m$  and has latency  $t_m$ , addition costs  $c_a$  in space and  $t_a$  in time). This is a very crude simplification if one attempts to optimize each operator, think for multiplications by constant for instance. The availability of many audio-related case studies in Emeraude will allow us (hopefully in collaboration with CASH) to develop arithmetic-aware scheduling and tiling techniques that will eventually prove useful well beyond the world of digital audio.

**Towards provably optimal solutions** Recent arrivals of Christine Solnon and Anastasia Volkova into the team bring more focus into the optimization. Given a mathematical object to implement in hardware (for instance the 2D norm  $\sqrt{x^2 + y^2}$ ), the standard practice so far has been to 1/ define a family of hardware algorithms parameterized by architectural parameters (typically the number of bits for the intermediate results), 2/ express the constraints for a given solution to be acceptable (typically that the hardware should provide a faithful or correct rounding of the exact result), and 3/ finally define a heuristic that provides the parameters for an acceptable solution with good performance (performance being, for instance, area or delay of the hardware). The shift is to replace the heuristic in the third step with a mathematical model that can be solved with standard solvers (ILP, SAT or CP) to provide hardware operators with *provably optimal* performance. This approach was pioneered by Martin Kumm about ten years ago, and used in Emeraude recently for squarers [44], elementary function evaluators [53], and digital filters [108]. In the coming years we will apply this approach more broadly to operators for digital signal processing (in collaboration with axes 1 and 3) and artificial intelligence accelerators (in collaboration with other members of the PEPR IA). We will also use optimization for core classical arithmetic problems such as function approximation and evaluation, compressor tree synthesis, or word-length optimization, again in collaboration with Axis 3.

### 3.3 The Faust Programming Language and its Ecosystem

**Participants:** Florent de Dinechin, Stéphane Letz, Romain Michon, Yann Orlarey, Tanguy Risset, Christine Solnon.

Audio signal processing is an applied field where each result, algorithm, method, or tool ends up being validated by the human ear. This validation requires efficient tools to rapidly prototype audio signal processing algorithms. For many years, languages and tools for audio DSP have been developed by researchers to ease the implementation and the deployment of new audio processing algorithms. The FAUST programming language and environment were invented in that context at Grame-CNCM. Emeraude continues to bring new developments around these tools.

**The FAUST language and its compiler** A large part of Emeraude’s research results is visible thanks to the FAUST ecosystem development. FAUST has gained an international recognition, especially since it is used for teaching at Stanford University (in the context of courses on signal processing, physical interaction design, etc.) and for developing new audio plugins [83]. The efforts needed to keep FAUST as the most efficient language for real-time audio processing involve research in: language design, compiler design, and development of DSP libraries.

One of the reason of FAUST’s success is that it is both a *language* and an *environment* for audio signal processing. The FAUST compiler typically generates high-level codes (in languages such as C, C++, etc.), following every compiler’s goal: providing better code than manually written code. For that, it has to stick to the most recent compiler technologies and processors evolutions [77]. For instance, a back-end for WebAssembly was recently added to the FAUST compiler [76]. An important deployment step was the embedding of the FAUST compiler in a web browser [75] which makes it easily accessible on all computers.

**FAUST language design research in Emeraude** The current design of FAUST, inspired by lambda-calculus, combinatory logic and John Backus’ functional programming, has to be extended to face new challenges, in particular multi-dimensional and multi-rate signals and linear algebra.

<sup>21</sup>xtremlogic

FAUST allows for the description of synchronous mono-rate scalar DSP computations. This is sufficient to implement most time-domain algorithms such as filters, oscillators, waveguides, etc. However, this makes the implementation of frequency-domain algorithms (i.e. based on FFT) very inefficient, not to say impossible. One of our goals is to extend the language to enable multi-rate as well as vector computations. While we already have a working prototype for this, some challenges have yet to be overcome.

Along the lines of the previous point, FAUST currently doesn't provide any support for efficient matrix operations and more generally linear algebra. This prevents the implementation of some classes of DSP algorithms such as Finite-Difference Time-Domain (FDTD) method for physical modeling. The skills of former Socrate members on seminal Alpha language [59] and polyhedral optimization are very useful here.

Support for the main target programming languages in FAUST is essential. Recently added languages (WebAssembly, Rust, and CMajor) have opened many new opportunities. The FPGA target, studied in §3.1, introduces new challenges such as the ability to use fixed-point arithmetic or the use of HLS for targeting hardware platforms (e.g., VHDL, Verilog, etc.). Other "exotic" architectures such as GPUs or MPSoCs should be studied for compute-bound algorithms.

Musicians have to deal with a large variety of operating systems, software environments and hardware architectures. FAUST is designed to favor an easy deployment of DSP programs on all these targets by making a clear separation between computation itself, as described by the program code, and how this computation should be related to the external world. This relation (with audio drivers, GUIs, sensors, etc.) is described in specific *architecture files* [62]. Architecture files concern both hardware (i.e., audio interfaces/sound cards) as well as software control interfaces (e.g., GUI, OSC,<sup>22</sup> MIDI), new lutheries (e.g., SmartFaust, Gramophone), Web platforms (Web audio Plugin), etc. One of the goal of the work of Emeraude on FAUST is to ease the programming of these audio systems.

**FAUST ecosystem and DSP libraries** FAUST users are very attached to its ecosystem, including native applications, online and "embedded" audio applications, Just In Time (JIT) compiler, etc. Recent developments include a JIT FAUST compiler on the Web, a JIT compiler in the Max/MSP environment, tools to find the best compilation parameters and ease compilation for multiple CPUs. This is constantly evolving to answer to users' demand.

The FAUST DSP libraries currently implement hundreds of functions/objects ranging from simple oscillators and filters to advanced filter architectures, physical models, and complete ready-to-use audio plugins. These libraries are at the heart of FAUST's success and international position. Julius Smith<sup>23</sup> (Stanford professor) is one of the most respected figures in the field of audio DSP and one of the main contributors to the FAUST libraries. One of the ambitions of the Emeraude team is to maintain and extend this tool to make it as exhaustive and as universal as possible. Along these lines, new developments made to the language presented above (e.g., multi-rate, linear algebra, etc.) should be ported to the libraries. Finally, dedicated libraries targeting specific hardware platforms (e.g., microcontrollers, FPGAs) should be made available too.

**Machine learning for digital signal processing** Machine learning and deep learning in particular, are playing an increasingly important role in the field of audio DSP. Researchers are revisiting the algorithmic techniques of signal synthesis and processing in the light of machine learning, for instance for speech processing [64]. Recent breakthroughs such as the use of machine learning use in the context of Differentiable Digital Signal Processing (DDSP) [61] demonstrate its power. The extension of FAUST applications to artificial intelligence began with the PhD work of Benjamin Quiédeville, expanding the scope of FAUST in the field of AI. The objective is the introduction of an autodifferentiation primitive for FAUST programs, aimed at machine learning applications. The development of new backends is planned, particularly for MLIR, MOJO, and GPUs, as well as support for emerging architectures, especially in the context of game engines and VR/AR applications.

**Embedded systems for audio processing** As Emeraude's name suggests it, the implementation of audio Digital Signal Processing on embedded hardware is at the heart of the project. We naturally rely on the FAUST language for these implementations. The skills of Emeraude members in compilation and embedded systems

<sup>22</sup>Open Sound Control: HTTP-based communication protocol heavily used in the field of computer music.

<sup>23</sup>[ccrma.stanford.edu/jos](http://ccrma.stanford.edu/jos)

are used to add new embedded target for audio processing, in particular FPGAs, as explained previously. This action is a mix of research and engineering work, it should be very useful for the dissemination of audio processing programming.

Haptics is a huge topic, especially in the field of New Interfaces for Musical Expression (NIME), which has been studied for many years [51, 105]. It has always been tightly coupled to physical modeling because this sound synthesis technique provides natural connections to the physical world. A big part of the challenge is technological because haptics require ultra low-latency and high sampling resolution in order to be accurate. This is at the heart of Emeraude's goals.

Virtual and Augmented Reality (VR/AR) is not limited to immersive 3D graphics, sound also has an important role to play in that emerging field. Lots of work have been done around using VR environments as a creative tool for audio [74, 49, 48]. While many VR-based musical instruments have been created in the past [97], little work has been done around implementing interfaces specifically targeting VR/AR audio environments, especially in the context of 3D sound. This is something that we plan to explore as part of Emeraude.

Finally, beside ergonomic and HCI aspects, the design of musical interfaces is impacted by various kinds of technical limitations that we plan to address as part of Emeraude. First, just like for real-time audio processing, latency plays a crucial role in this context. Similarly, the "time resolution" (e.g., the sampling rate of the interfaces) can have a huge impact, especially when targeting specific kinds of instruments such as drums. Finally, the "spatial resolution" (e.g., the number of sensor points per squared centimeters on a tabletop interface) also impacts its quality. In this context, we would like to develop an embedded, high-resolution, high-sampling-rate, multi-touch multi-dimensional (X and Y + pressure) interface/instrument leveraging the development carried out in the previous axes. This work would be followed by a user study to measure the impact of this type of advanced system on perception.

## 4 Application domains

Emeraude aims at being a world leading research team on audio systems, carrying out fundamental research. However, Emeraude's research topics do belong to the spectrum of applied research. Hence, discoveries made in the context of Emeraude should be illustrated with experimental prototypes and demonstrations. Here is a brief overview of various application fields where research developed in Emeraude could be applied.

### 4.1 Spatial active noise control

Noise control is a major issue in many industries: transport, construction, multimedia, etc. Active noise control techniques can help to partially remedy this problem.

However, the implementation of such approaches requires several microphones and loudspeakers, whose signal processing must be done in real-time and faster than the propagation time of the acoustical waves. In these applications, FPGA solutions are therefore the most suitable way to program such devices, and the flow proposed in §3.1 is of great interest in this context.

For instance, it could be used for single-channel controllers: a theme already developed, for example for active headsets [41]. In that case, low latency allows for fully digital feedback control to be implemented. More generally, the feedback control previously limited to small, non-modular spaces, can be extended to a variety of situations, given the flexibility and adaptability of digital filters. Another extension would be the implementation of multichannel controllers: experiments have already been performed for the implementation of multichannel feedforward FPGA controllers with the development of architectures adapted from the FXLMS reference algorithm [98]. This allows developments to be considered in a real-world context.

### 4.2 Virtual acoustics/spatial audio

Controlling noise is only one of the applications of the aforementioned system. There is a rather strong interest at the moment for the replication of virtual acoustic spaces for "immersive experiences." Stanford is currently discussing the possibility of integrating a virtual acoustics component to the replica of the Chauvet cave in Ardèche with the scientific director of the Chauvet cave program. The idea would be to make acoustic measurements of the real cave and to set up a system which, by capturing the position of the visitor's head,

would allow him to hear the guide's voice as if he were in the real cave (in 3D). Emeraude (Romain Michon) is part of the think-tank on this topic.

Research around Virtual Reality (VR) and Augmented Reality (AR) systems is very active today: immersive/augmented experience: audio guides, AR headsets implementing binaural rendering, augmented acoustics experience, with a strong focus on the development of systems supporting binaural rendering. Emeraude will be active in this domain too.

### 4.3 Industrial acoustics

Industrial developments of active noise control systems have so far been limited either to small spaces (e.g., active headsets, low-frequency ducts for aeraulic systems, etc.) or to noises of a particular nature (e.g., periodic noise from propeller aircraft, land vehicle engines, etc.). Our FPGA-based solution, which offers low latency and high computational capabilities, would enable the extension of controlled volumes, and the possibility of active noise control over any kind of noise. This includes for instance the automotive sectors where the reduction of road noise inside the passenger compartment is a big concern [67].

Another application would be the active treatment of boundary conditions with the realisation of “smart surfaces” for absorption [78, 42], or vibro-acoustic isolation [80, 65, 111]. The development of active material is based on multi-channel control systems combining global control and decentralized feedback systems. The use of FPGAs would enable them to be applied on a large scale, in buildings and also in transport systems (e.g., aircraft, turbojet nacelles, etc.). The LMFA is developing both the experimental means (i.e., MATISSE and CAIMAN test benches, ECL-B3 test bench from Equipex PHARE, etc.), and the numerical codes of acoustic propagation [52, 101], within the framework of a strong partnership with Safran Aircraft Engines (ANR ADOPSY and ARENA industrial chairs). The development of a high-level compiler dedicated to Acoustic Digital Signal processing on FPGAs is therefore of high interest for many researchers in acoustic for numerous industrial applications.

### 4.4 Medicine/sonification

There is a trend in the medical world towards the “sonification” of medical data such as EEGs, etc. The idea behind this concept is that our brain can process time series much faster and with much more precision if they are “encoded” as sound than if they are plotted on a graph. For instance, trained doctors can spot patterns which are characteristics of seizures in EEGs just by listening to their sonified version, which would not be possible just by looking at the corresponding plot. In that context, a “brain stethoscope” which basically sonifies the output signal of an EEG cap in real-time is currently being developed and will be released soon.<sup>24</sup> This type of development will be greatly simplified by the tools developed by Emeraude.

### 4.5 Low-latency audio effect processors and synthesizers

Custom low-latency synthesizers and sound processors (i.e., audio effects) are currently mostly out of reach to people in the audio and music technology communities. Indeed, the high-level programming environments used by these groups (e.g., Max/MSP, SuperCollider, etc.) cannot be used to program embedded audio platforms targeting low-latency applications. Instead, they were meant to be executed on personal computers which have potentially way more audio latency than embedded systems. Providing people in these communities with a tool (from §3.1) solving this problem would completely revolutionize the way they approach their tool chain.

### 4.6 Digital luthiery

Since the 1980s, digital equipment has become deeply embedded in all parts of the popular music production, distribution and consumption chain. In a market whose worldwide sales exceed 15 billion euros, digital instruments (also known as “Digital Luthiery”) are only the latest chapter in the long history of music technology. Digital instruments have sped up the evolution process by increasing accessibility of musical equipment to practitioners, especially young people, who can now achieve at home with inexpensive devices

---

<sup>24</sup>[chrischafe.net/brain-stethoscope-news](http://chrischafe.net/brain-stethoscope-news)

the kind of professional-calibre sounds that previously would have needed a large recording studio. Modern musical instruments are all in need of some form of embedded audio processing in which Emeraude could play a central role.

Grame is actively contributing to this effort by creating tools easily accessible to the maker community: open platform to design musical instruments, educational tools, etc.

## 5 Highlights of the year

### 5.1 JIMLAC-25



Figure 5: 2025 Linux Audio Conference participants.

In 2025, Emeraude organized JIMLAC-25<sup>25</sup> in collaboration with GRAME-CNCM. This major scientific and artistic event combined the *2025 Journées d'Informatique Musicale (JIM)* and the *2025 Linux Audio Conference (LAC)* and it took place at INSA Lyon on June 23-28, 2025. With a total of six days of scientific conference and six concerts, it hosted around 200 participants. Romain Michon was the general chair of JIMLAC-25 and Stéphane Letz the technical chair.

### 5.2 Awards

Orégane Desrentes and Florent de Dinechin received a Best Paper Award for their article [12] at the the [ARITH 2025](#) conference. Florent de Dinechin received the Best Presentation Award at the [2nd FPGA Developers' Forum meeting](#) organized at CERN.

## 6 Latest software developments, platforms, open data

### 6.1 Latest software developments

#### 6.1.1 FloPoCo

**Name:** Floating-Point Cores, but not only

**Keyword:** Synthesizable VHDL generator

**Functional Description:** The purpose of the open-source FloPoCo project is to explore the many ways in which the flexibility of the FPGA target can be exploited in the arithmetic realm.

**URL:** <http://flopoco.org>

**Contact:** Florent De Dinechin

---

<sup>25</sup>[jimlac25.inria.fr](http://jimlac25.inria.fr)

**Participant:** 2 anonymous participants

**Partners:** ENS Lyon, Insa de Lyon, Inria, Fulda University of Applied Science

### 6.1.2 Syfala

**Name:** Low-Latency Synthesizer on FPGA

**Keywords:** FPGA, Compilers, High-level synthesis, Audio signal processing

**Functional Description:** The goal of Syfala is to design an FPGA-based platform for multichannel ultra-low-latency audio Digital Signal Processing programmable at a high-level with Faust and C++ and usable for various applications ranging from sound synthesis and processing to active sound control and artificial sound field/room acoustics.

A series of tools are currently being developed around SyFaLa. SyFaLa is freely accessible on GitHub: <https://github.com/inria-emeraude/syfala>.

**URL:** <https://faust.grame.fr/syfala/>

**Contact:** Tanguy Risset

### 6.1.3 FAUST

**Name:** Functional Audio Stream

**Keywords:** Audio, Functional programming

**Functional Description:** The core component of Faust is its compiler. It allows to "translate" any Faust digital signal processing (DSP) specification to a wide range of non-domain specific languages such as C++, C, LLVM bit code, WebAssembly, Rust, etc. In this regard, Faust can be seen as an alternative to C++ but is much simpler and intuitive to learn.

Thanks to a wrapping system called "architectures," codes generated by Faust can be easily compiled into a wide variety of objects ranging from audio plug-ins to standalone applications or smartphone and web apps, etc.

**URL:** <https://faust.grame.fr/>

**Contact:** Yann Orlarey

**Partners:** GRAME, Insa de Lyon, Inria

## 7 New results

### 7.1 Interaction within the team

During 2025, Emeraude team members strengthened collaborations between the team's research axes (embedded audio and FPGA, computer arithmetic and optimisation), resulting in several concrete outcomes. First, the computer arithmetic and optimisation groups initiated joint work on new constraint programming models for designing efficient hardware architectures for multiplication circuits, which led to the publication of two papers [11, 9]. This collaboration notably included the supervision by Christine Solnon and Anastasia Volkova of Théo Cantaloube (a master internship that evolved into a PhD thesis), focusing on the intersection of the two research axes. Second, Romain Michon and Anastasia Volkova collaborated through the co-supervision of a master student on the optimisation of sigma-delta DAC circuits, with results published at the DASIP 2026 conference. Third, Pierre Cochard and Louis Ledoux made significant progress [34, 34, 2] toward proposing a complete MLIR flow for simple Faust programs leveraging the FLoPoCo tool, thereby further reinforcing the links between arithmetic tools developed by the team and signal processing applications. Finally, an article summarizing coupling research result on audio on FPGA and hardware design [38] untitled "Frugalité et conception de circuits pour le traitement du signal audio numérique" was published in French in "Revue Francophone d'Informatique et Musique".

## 7.2 Immersive Audio

### 7.2.1 Distributing spatial audio

**Participants:** Thomas Rushton, Romain Michon, Tanguy Risset.

In [21], we introduced a low-cost, open, and scalable platform for distributed spatial audio rendering. It seeks to reduce the financial and technical barriers imposed by conventional immersive audio systems. For that we proposed a distributed architecture in which sound field synthesis algorithms, such as Wave Field Synthesis (WFS) or ambisonics, are parallelized across many inexpensive, networked embedded audio processors (see Fig. 6). We focused on the central technical challenge of achieving sufficiently precise time synchronization between physically separate audio devices to preserve wavefront integrity. To address this, we demonstrated how the IEEE 1588 Precision Time Protocol (PTP), implemented using open-source software on low-cost microcontroller platforms (specifically the Teensy 4.1), can be used both to align audio start times and to continuously correct sampling frequency drift by conditioning each device's audio phase-locked loop. Using an experimental setup with multiple microcontroller-based audio nodes synchronized via a PTP-capable Ethernet switch, we show that without correction, clock drift accumulates to the millisecond range, whereas with PTP-derived sampling frequency conditioning, relative timing errors are reduced by three orders of magnitude to the microsecond level, corresponding to sub-millimeter acoustic discrepancies. These results demonstrate that accurate, synchronous audio reproduction is feasible using inexpensive hardware and standard networking infrastructure, validating our approach as a practical foundation for accessible distributed spatial audio systems and motivating future work on scalability, integration with conventional audio workflows, and more computationally demanding applications such as real-time virtual acoustics and auralization.

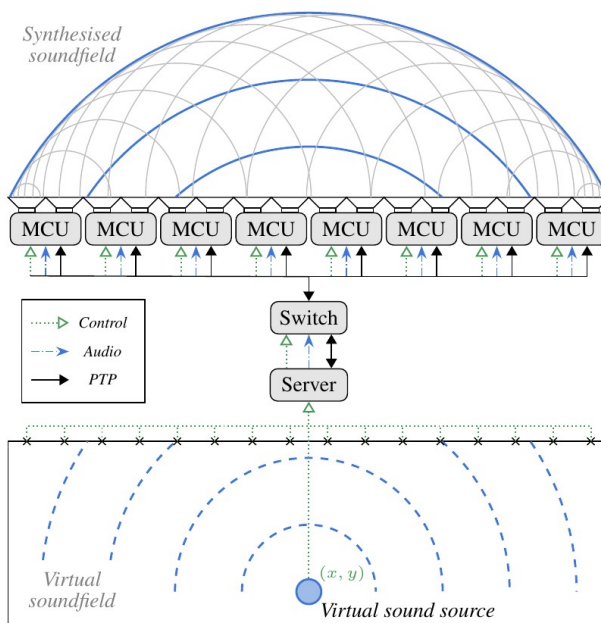


Figure 6: The holophonic effect of primary-source WFS is created by applying appropriate per-loudspeaker delays to a virtual sound source; delays are independent and can be computed in distributed fashion. A server delivers audio and control data to a collection of Microcontroller-based signal processors via an ethernet switch; each microcontroller is informed of its position, and the position of the virtual sound source; microcontroller audio clocks are conditioned via PTP to match that of the server. Synchronicity amongst the group of microcontrollers is necessary to ensure the integrity of the synthesised wavefront.

### 7.2.2 Real-Time Auralization on FPGA

**Participants:** Rémi Jeunehomme, Romain Michon, Tanguy Risset, Pierre Cochard, Stéphane Letz.

In [15], we investigated the feasibility of real-time auralization on FPGAs by implementing and evaluating convolution-based artificial reverberation algorithms tailored to FPGA architectures. We focused on the computational challenges posed by long room impulse responses and explored FPGA-based solutions as an alternative to traditional CPU- and GPU-based approaches, motivated by the low latency, parallelism, and scalability offered by reconfigurable hardware. Using Syfala [91] and High-Level Synthesis (HLS), we implemented two uniformly partitioned overlap-save convolution algorithms in C: a time-domain approach (TUPOLS) based on direct convolution, and a frequency-domain approach (FUPOLS) relying on FFT-based circular convolution. We analyzed their architectural suitability, detailing memory organization, dataflow pipelining, numerical format choices, and hardware-specific optimizations required to meet real-time constraints on a Xilinx Zynq-based FPGA platform. We showed that while the time-domain approach is severely limited by latency and resource consumption for impulse responses longer than about one second, the frequency-domain approach achieved real-time performance for impulse responses exceeding ten seconds with acceptable FPGA resource usage. This work opens the way to potential large-scale multichannel auralization applications on FPGA.

### 7.2.3 Auralization of Paleoacoustics landscapes in Chauvet Cave

**Participants:** Romain Michon.

In [22], we reported on an ongoing interdisciplinary study carried out in the context of the MIRAGES associate team (see §9.1.1) aimed at measuring, analyzing, and auralizing the paleoacoustic landscapes of Chauvet Cave (south of France), in order to better understand how sound may have shaped human experiences of the cave during the Upper Paleolithic. We described the methodological and conservation-driven constraints that governed our work, including restricted access to the cave, limitations on equipment, and the fact that measurements were confined to modern walkways, leaving large portions of the cave acoustically undocumented. To address these challenges, we carried out several field campaigns between 2022 and 2024, during which we collected thousands of impulse responses using low-impact measurement protocols based on exponential sine sweeps, omnidirectional and Ambisonic microphones, and portable loudspeakers. We analyzed the resulting data to assess signal quality and extract room-acoustic parameters such as reverberation time, clarity, and definition, showing that despite logistical constraints, most measurements achieved sufficiently high signal-to-noise ratios for reliable analysis. We then discussed how these measurements informed multiple auralization frameworks, including offline convolution, web-based interactive tools, real-time multichannel installations, virtual reality environments, and museum exhibits, each offering different balances between realism, interactivity, and accessibility. This work opens the way to future research directions, including predictive acoustic modeling based on reconstructed Paleolithic geometries, improved material characterization, and expanded interactive auralization platforms, positioning this work as a foundational step toward scientifically grounded yet explicitly speculative reconstructions of prehistoric soundscapes for both research and public engagement.

### 7.2.4 The Space Bar, an Embedded WFS Sound System

**Participants:** Benjamin Quiedeville, Romain Michon, Pierre Cochard, Stéphane Letz.

From November to March, an embedded WFS system was designed and built at the laboratory in the context of an exhibition at the Grenoble INRIA centre, based on an earlier prototype created by the Emeraude



Figure 7: Acoustical measurements in the Salle Hillaire of the Chauvet cave.

team. It leverages the Syfala toolchain [91] developed in the Emeraude team to create an FPGA based, frugal and autonomous device capable of spatialising a high number of audio sources on 32 speakers using a Wave Field Synthesis (WFS) [100] algorithm and controlled via a standard game controller.

The construction involved the building of the casing and the electronic wiring of the FPGA board on special multi channel audio interfaces [90] and the programming of the integrated ARM CPU using a custom Alpine Linux distribution provided by Syfala. The role of this program is to open audio files and send the audio data alongside position control signals to the FPGA. The project involved the publication of an article in the proceeding of the *Journées de l'Informatique Musicale* conference in Lyon in June 2025.

The composer Frédéric Khan has worked in collaboration with GRAME on the production of a musical piece to be integrated in the device in march 2026, a special version of the control program was implemented to accomodate for composers for enhanced creativity and ease of writing. Additional resources used during the work were [92, 106, 104].

## 7.3 Acceleration of AI inference

### 7.3.1 HATorch: Hardware-aware quantization-aware training for CNNs

**Participants:** Bastien Barbe, Romain Bouarah, Anastasia Volkova, Florent de Dinechin.

Popular machine learning frameworks like PyTorch and TensorFlow provide complete toolchains to design, train, quantize and deploy convolutional neural networks (CNNs) on standard hardware (CPUs, GPUs, TPUs, NPUs, microcontrollers). However, custom hardware targets like FPGAs, ASICs and research accelerators typically use non-standard number representations as well as limited operator sets, and are poorly served by current lowering backends. Existing backend-driven approaches rigidly force training to their internal formats and often hide lowering choices behind opaque layers and closed software development kits. Conversely, unconstrained quantization that permits arbitrary numeric and operator freedom often produces models that are difficult to implement efficiently, leading to an hardware gap between the trained and deployed model. This work introduces HATorch, a PyTorch-based hardware-aware training framework that reduces this gap by enhancing quantization with hardware architecture model on the arithmetic operator/format level. HATorch supports custom hardware-friendly quantization flows, exposing lowering decisions for transparent model-hardware co-design. We demonstrate the toolflow on the new hardware-friendly multipliers based on [9].

This QAT framework is a shared contribution of two PhD students from the team, Bastien Barbe and Romain Bouarah. The conference paper was accepted in 2025 and will be presented in early 2026 at the 8th AccML Accelerated Machine Learning Workshop at the HiPEAC conference in Krakow.

HATorch is also being used for the exploration of other type of neural network architectures, based on Kolmogorov-Arnoldi Networks (KANs) in the preliminary work [33].

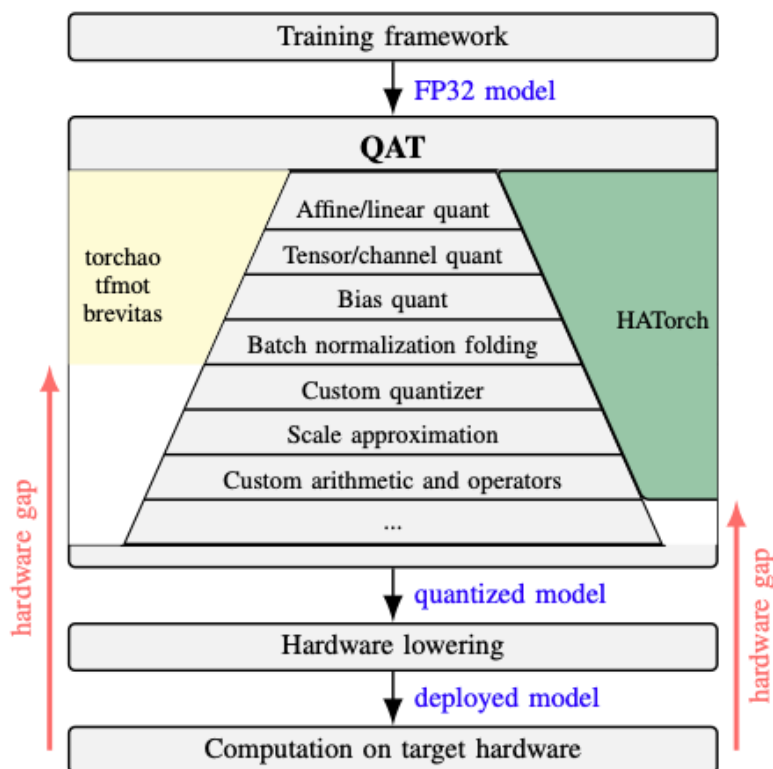


Figure 8: Classical flows for lowering of trained models to HW exhibit a a substantial performance gap. HATorch incorporates HW model into the quantization-aware training process in a clear and transparent way, permitting utilization of unconventional arithmetics and reducing the final performance mismatch between software and hardware

### 7.3.2 Towards frugality in Natural Language Processing classification models

**Participants:** Anastasia Volkova.

This line of work is part of a broader collaboration on mixed-precision quantization-aware training (QAT) for NLP models, involving Anastasia Volkova, Cédric Gernigon, Richard Dufour, and Xavier Pillet from Nantes University.

The first contribution, based on [14] and to be published at DASIP 2026, investigates mixed-precision quantization for BERT inference, with the goal of reducing memory and computational costs while preserving accuracy. Unlike most prior work, the study jointly considers mixed-precision quantization of both weights and activations (see Fig. 9), integrates knowledge distillation into the quantization pipeline, and evaluates the impact of quantizing the embedding layer beyond token weights. Experiments on the SQuAD and GLUE benchmarks show that mixed-precision configurations achieve substantial reductions in resource usage without degrading performance.

The second contribution (to be presented at HiPEAC AccML workshop) extends this work to the question of whether low-bitwidth representations can be learned jointly with new domain-specific knowledge during fine-tuning. Focusing on biomedical NLP under strong hardware constraints, it studies the interaction between domain-adaptive pre-training, fine-tuning, and extreme quantization. Experiments on French biomedical

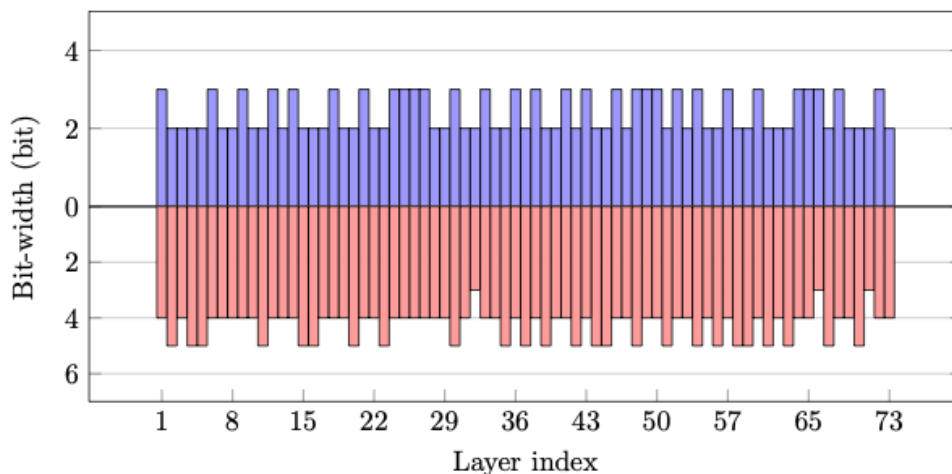


Figure 9: Example of learned mixed-precision formats for BERT layers on the CoLa dataset. Weights (blue, on top) and activations (red, in the bottom) have variable precisions learned during training.

data show that domain-specialized models are more robust to aggressive quantization than generalist ones, and that structural ternary quantization enables stable training in very low-precision regimes. Even limited domain adaptation significantly improves convergence, indicating that domain learning and adaptation to low numerical precision can be achieved simultaneously.

## 7.4 Optimization of arithmetic cores

### 7.4.1 Double-Word Decomposition in a Combined FP16, BF16 and FP32 Dot Product Add Operator

**Participants:** Oregane Desrentes, Florent de Dinechin.

This work [12] introduces a fused Dot Product Add (DPA) operator that supports mixed-precision operations for both machine learning and numerical computing by combining FP16, BF16, and FP32 multiplicands within a single hardware unit. The key technical innovation is the use of an intermediate floating-point format (E9S12)—with a 9-bit exponent and 12-bit significand—that enables an exact double-word decomposition of FP32 multiplicands and efficient internal representation of lower-precision inputs. The resulting operator is correctly rounded for FP16 products accumulated into FP32 and also emulates the IEEE-compliant FP32 fused multiply-add when operating on decomposed FP32 inputs, simplifying hardware compared to traditional double-word software techniques. Synthesis results for a 4 nm technology node show that this combined operator improves both performance and accuracy over software decomposition approaches.

This contribution was recognised with the Best Paper Award at the ARITH 2025 conference, highlighting its significance in understanding and optimising the trade-offs inherent in mixed-precision hardware design.

### 7.4.2 Reconfigurable constant multiplication

**Participants:** Bastien Barbe, Louis Ledoux, Anastasia Volkova, Florent de Dinechin.

This work [9] has been done in the context of the PhD thesis of Bastien Barbe in collaboration with the team’s postdoc Xiao Peng. It addresses the design of efficient hardware multipliers in the context of extremely quantized deep neural networks, where multiplications are performed with constants drawn from a small,

fixed set. Rather than relying on standard numerical formats (e.g. int4), which constrain both representation and hardware design, the approach assumes that only a carefully chosen set of integer constants needs to be supported. It proposes dedicated reconfigurable shift-and-add architectures capable of implementing these constants efficiently by switching between pre-computed computation graphs. In this setting, constants are represented implicitly through configuration bits that select the appropriate circuit, allowing values of larger magnitude to be handled with fewer bits than standard encodings. Experimental results show that the proposed algorithms outperform existing heuristics, support a larger number of constants simultaneously, and lead to more efficient multiplier designs in terms of hardware cost.

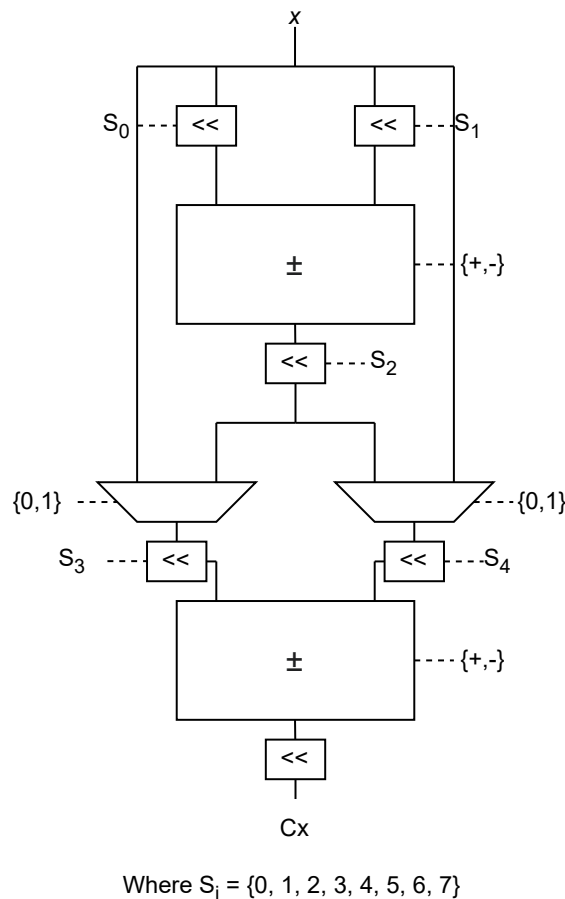


Figure 10: Example 2-adder RSCM (reconfigurable single constant multiplier)

### 7.4.3 Constraint programming models for multiple constant multiplication

**Participants:** Anastasia Volkova, Christine Solnon, Theo Cantaloube.

The Multiple Constant Multiplication (MCM) problem arises in many applications such as, for example,

digital signal processing. Given a set  $T$  of target constants, the goal of MCM is to find the most efficient way for multiplying an input number with each constant in  $T$ , where multiplications are realized through bit-shifts and additions, and where intermediate results may be shared to produce different target constants. State-of-the-art methods are based on Integer Linear Programming (ILP), and suffer from numerous performance and scalability bottlenecks.

In [11] we have proposed for the first time a Constraint Programming (CP) model for minimizing the number of adders for the MCM. Compared to the state-of-the-art ILP approach, CP does not suffer from the curse of linearization, hence permits significantly simpler formulations of the mathematical model. It is also more efficient, especially for the hardest instances. We have also introduced a pseudo-polynomial time algorithm which is able to efficiently solve some instances.

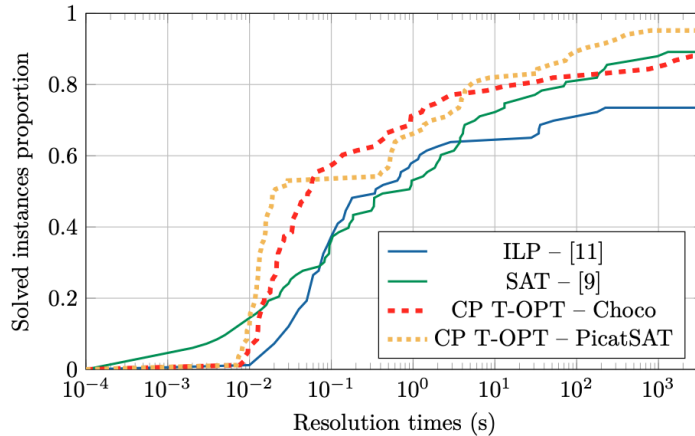


Figure 11: Performance comparison of state-of-the-art approaches to our CP model on a comprehensive benchmark.

This work has been done in the context of a 6-month internship of Theo Cantaloube, co-supervised by Anastasia Volkova and Christine Solnon. Theo Cantaloube has started a PhD thesis in October 2025. The goal of this thesis is to investigate the capabilities of CP for solving this kind of problems.

#### 7.4.4 Optimization for other application domains

**Participants:** Romain Fontaine, Xiao Peng, Christine Solnon.

Beyond the application to arithmetic, which is at the core of the research topics of Emeraude, we have also applied optimization and constraint programming techniques to other problems: surface coverage by tethered robots [4], the travelling salesman problem with time windows [5, 7], and graph generation [19].

### 7.5 From FLoPoCo to MLIR dialects and flows

Emeraude-MLIR is a multi-level arithmetic optimization compilation framework. It can be viewed as an end-to-end compilation flow that targets a wide range of inputs, including DSP applications such as Faust, machine learning workloads expressed in PyTorch and LLaMA, and HPC kernels such as BLAS routines and polyhedral benchmarks from PolyBench. These inputs are mapped to multiple backends, including FPGA, ASIC, and software targets through LLVM.

These compilation paths naturally expose dozens of lowering transformations, with a strong emphasis on arithmetic specialization. Many of these transformations are context-aware, for example selecting an accumulator bitwidth for a given sub-matrix block or adapting internal ROM sizes to FPGA resource constraints. The internal transformations and intermediate representations, expressed through a hierarchy

of dialects, encompass polynomial approximations of real-valued computations, IEEE 754-compliant floating-point combinational IR construction, and graph rewriting over structured control flow to capture exact accumulation semantics.

Overall, the flow has demonstrated its capabilities by lowering a LLaMA attention block to silicon and by producing a silicon-proven Faust soft-clipping function fabricated through a recent TinyTapeout shuttle, occupying wafer area in GlobalFoundries 180 nm technology.<sup>26,27</sup>

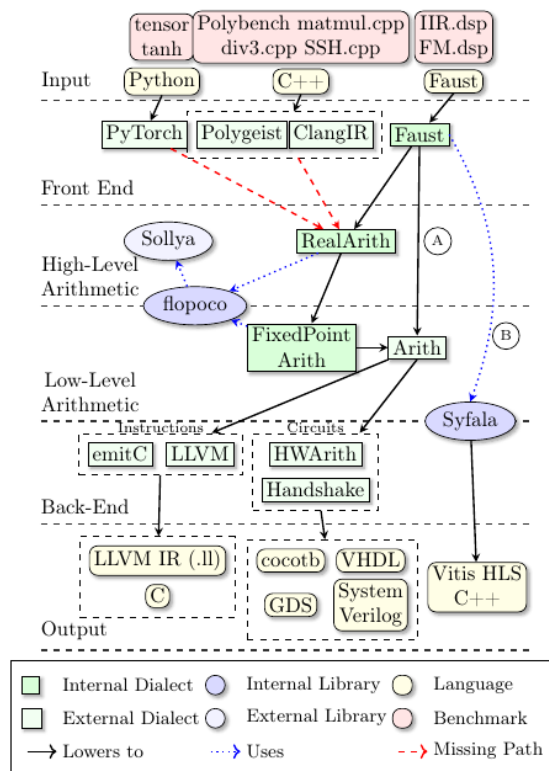


Figure 12: Emeraude MLIR flow

## 7.6 The FAUST Programming Language and its Ecosystem

**Participants:** Florent de Dinechin, Stéphane Letz, Romain Michon, Yann Orlarey, Tanguy Risset, Christine Solnon.

Audio signal processing is an applied field where success is ultimately determined by the human ear, requiring advanced tools to prototype and implement algorithms rapidly and efficiently. The FAUST programming language and environment, developed at Grame-CNCM in 2002 [87], represented a significant development by enabling researchers and developers to prototype and deploy audio processing algorithms more efficiently. At the time, FAUST was the first fully compiled audio programming language, which played an important role in making the field of real-time embedded audio systems more accessible.

Since its inception, FAUST has gained international recognition and is widely used in both academic and professional contexts. It has been adopted for teaching advanced topics such as signal processing and physical interaction design at Stanford University and for developing audio plugins [83].

Today, the FAUST research project, conducted by Emeraude, focuses on three interrelated areas:

<sup>26</sup><https://github.com/Bynaryman/ttcf0p2-faust-mlir-silicon>

<sup>27</sup>[https://tinytapeout.com/chips/ttcf0p2/tt\\_um\\_gf0p2\\_faust\\_top](https://tinytapeout.com/chips/ttcf0p2/tt_um_gf0p2_faust_top)

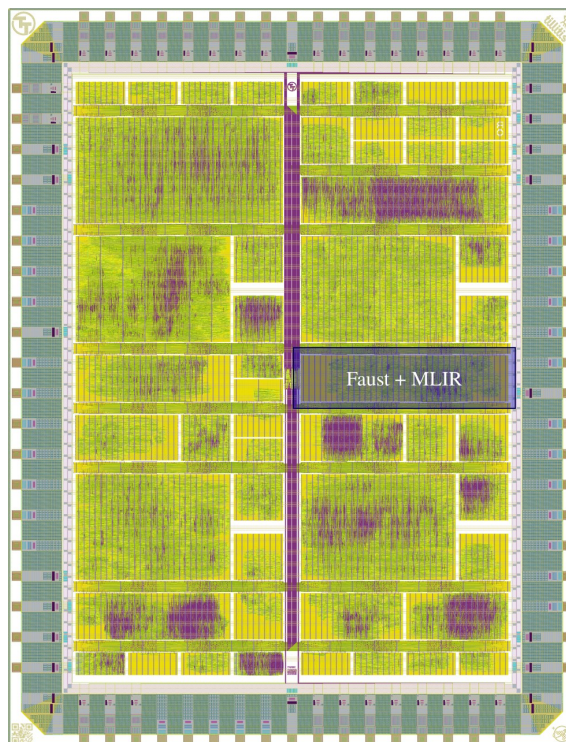


Figure 13: TinyTapeout GF180 silicon for the Faust MLIR soft-clipping design.

- **The FAUST Programming Language:** Developing a high-level language for sound synthesis and signal processing that is accessible to non-computer scientists.
- **The Compiler and Compilation Techniques:** Producing tools to automatically generate highly optimized code, comparable in efficiency to code written by experienced C programmers.
- **The Ecosystem:** Expanding and maintaining architecture files (which allow the same FAUST code to run on over twenty platforms), development tools, libraries, and documentation.

The following sections present the work carried out during the period in each of these areas.

### 7.6.1 The FAUST programming language

FAUST is a synchronous functional programming language inspired by lambda calculus, combinatory logic, and John Backus' FP. Its semantics is centered around the concept of audio circuits. Programming in FAUST primarily involves constructing new audio circuits by assembling primitive ones using an algebra of five composition operations. During the period, however, FAUST has evolved beyond simple circuit assembly.

### 7.6.2 The FAUST compiler

To support FAUST's evolving functionality and optimize performance across a wider range of platforms, recent advancements in the FAUST compiler—driven in part by the Syfala project—have introduced innovative compilation techniques that now target not only CPUs but also FPGAs, enabling efficient code generation and dynamic processing capabilities on diverse hardware architectures.

**FIR/IIR reconstruction** Research work on FIR and IIR reconstruction in the FAUST compiler continued in 2025 with the implementation of new techniques that accelerate and simplify the process.

Let's recall the project's starting point. The minimalist design philosophy of the FAUST language deliberately excludes high-level signal processing functions that could be expressed using lower-level

primitives. This is why the FAUST language does not have any filter-type primitives (FIR or IIR) since they can easily be expressed using delay lines and recursions. Thus, all common signal processing filters are defined in the standard library ‘filters.lib’, which contains over 150 of them.

Thanks to various optimization techniques, such as delay line sharing, there is no penalty for not having these filters as language primitives. However, this "unrolled" approach is not the most efficient with high-level synthesis tools like Vivado HLS.

This is why we introduced new compilation options that automatically reconstruct FIR and IIR filters in the compiler’s internal representations. While these options do not necessarily improve CPU performance, they are clearly very efficient on FPGAs in terms of latency and resource trade-offs.

The new approach consists of assembling binary additions and subtractions into n-ary weighted sums. Signals that appear with a fixed delay in these sums are grouped into ‘FIR[x,a,b,c,...]’ and recursive definitions of the type ‘x = y + FIR[x,a,b,c,...]’ where ‘y’ does not depend on ‘x’, are transformed into ‘IIR[y,a,b,c,...]’.

**Automatic differentiation** The main subject is the implementation of automatic differentiation tools for the FAUST programming language in order to extend its capabilities to solving machine learning problems.

Automatic differentiation is a mechanism in which a program can compute the derivative of an arbitrary function at runtime, avoiding the need for the programmer to differentiate it beforehand or relying on other methods like symbolic calculus engines (WX Maxima <sup>28</sup>, Sympy <sup>29</sup>) or discrete calculus tools. Automatic differentiation is necessary to perform gradient descents and thus necessary for many optimisation problems and machine learning.

Previous work has been done on this topic. David Braun created a Jax [45] backend for FAUST <sup>30</sup> allowing the use of its gradient computation tools and the integration in its deep learning ecosystem. But implementing the mechanism directly inside FAUST would then propagate to all the existing and future backends and make use of the special aspects of the language to compute gradients in a new way.

Thomas Rushton of the Emeraude team has worked in the context of two *Google Summer of Code* on implementing automatic differentiation in the compiler and in the syntax of the language. His work was published in the proceedings of the 2024 edition of the *FAUST International Conference* [94] and is the main tool used in the beginning of this PHD to experiment with simple examples.

Automatic differentiation is split into two modes: forward mode, and reverse mode, that present different performance characteristics depending on the use case. For each mode, several implementation methods have been explored depending on the programming environment. Many projects exist for the implementation of the main methods: source transformation [66] (generating the derivative code given the primal function), gradient tape [63] (recording all the operation in a first pass and unrolling the derivative in a second, backward pass), operator overloading [93], LLVM IR transformation [84] (implemented as a plugin in the *clang* compiler).

Here, Rushton’s library falls in between *source transformation* and *operator overloading* as it is leveraging the pattern matching capabilities of the FAUST language to do operator overloading and the compiler is generating new code for that derivative to the chosen backend. This shows that the specificities of the FAUST language, as a Domain Specific Language (DSL) can open new ways of computing automatic differentiation.

A part of the work was to implement several basic methods in the Julia<sup>31</sup> programming language to understand a maximum of details on *Forward* and *Reverse* mode in a procedural environment rather than a functional one. Now the main task is to continue the work done by Thomas Rushton and create, measure and test specific examples to compare against the state of the art.

A differentiated version of the NLMS algorithm (consisting in optimising an FIR filter to match an unknown LTI system) was tried and the current results show that the FAUST compiler is struggling too much to generate the C++ code for an interesting amount of filter coefficients (> 100). It showed some of the limits of the compiler that will need to be worked on for a working implementation of differentiation algorithms. These results were presented in the context of the annual Emeraude team seminary in May 2025.

It was later found that delay lines of variable lengths could not be differentiated in the time domain and thus could not be integrated in our automatic differentiation schemes. This was a blockage as a significant part of the FAUST ecosystem as well as many audio algorithms rely on that mechanism.

<sup>28</sup><https://wxmaxima-developers.github.io/wxmaxima/index.html>

<sup>29</sup><https://docs.sympy.org/latest/index.html>

<sup>30</sup>[https://github.com/DBraun/DawDreamer/tree/main/examples/Faust\\_to\\_JAX](https://github.com/DBraun/DawDreamer/tree/main/examples/Faust_to_JAX)

<sup>31</sup><https://julia.org>

### 7.6.3 The FAUST ecosystem

**FAUST and Evolutionary / Genetic algorithms** Starting in september 2025, exploration of genetic algorithms and their potential integration with the FAUST programming language started. Genetic algorithms are an approach to automatic optimisation inspired by biology. Similar to classical gradient descent-based machine learning, a genetic algorithm is an iterative method that converges toward an optimal solution to a quantifiable problem. But unlike machine learning, genetic algorithms converge to the optimal solution with this scheme:

- generating a **generation of chromosomes** (a set of solutions);
- evaluating every chromosome and computing their respective loss function (called the fitness function) value.
- randomly selecting chromosomes for the next generation with a bias toward ones with the best fitness values.
- randomly modifying the selected chromosomes.
- repeating the process until convergence or for a given number of iterations.

For our use case, genetic algorithms have the advantages to guarantee the convergence of the optimisation given enough iterations, and to not require the differentiability of the system.

The work has been targeted toward implementing a genetic algorithm based optimisation tool that can optimize the parameters of a FAUST program given a target behaviour. For the experiments, the target systems were digital filters and the FAUST programs were designed to match the frequency response. A FIR (Finite Impulse Response) filter was chosen for the higher number of parameters (one per filter coefficient), better suited for genetic algorithms.

**FAUST MCP server architecture** In the FAUST ecosystem, an architecture is a software layer that connects compiled DSP code to the external world (audio drivers, user interfaces, communication protocols, etc.). FAUST architectures implement the glue code that bridges signal processing algorithms with various platforms and interaction modalities.

In early 2025, we developed a new component that allows FAUST audio applications to expose their sound parameters for control by AI assistants such as ChatGPT or Claude. This component relies on the Model Context Protocol (MCP) <sup>32</sup>, an open standard introduced by Anthropic in November 2024 that enables AI assistants to connect with external data sources and tools. The protocol defines a client-server architecture using JSON-RPC 2.0 communication over stdio or HTTP.

This FAUST architecture uses MCP's tool system to expose DSP parameters as callable functions. The MCP protocol distinguishes between tools (functions that can be invoked by AI clients), resources (data sources such as files, APIs, and databases), prompts (reusable templates for AI interactions), and servers (applications that expose tools and resources via the protocol). In our implementation, each FAUST widget becomes an MCP tool with a defined input schema. Tool names follow hierarchical naming based on FAUST UI structure. Parameter validation and range clamping are handled automatically, and communication occurs via JSON-RPC over stdin/stdout.

This FAUST MCP architecture is the first implementation enabling AI-driven control of audio DSP parameters through natural language. This approach opens up possibilities that go far beyond simple parameter adjustment. Users can now express complex temporal behaviors and relationships between parameters in natural language—for example, "gradually increase the reverb while slowly decreasing the delay feedback" or "oscillate the filter cutoff between 500Hz and 2kHz every three seconds while keeping the resonance stable."

This capability effectively transforms natural language into a form of performance notation or score, where intricate parameter evolutions can be described conversationally rather than programmed explicitly. Such an interface is particularly valuable for interactive sound installations, where visitors with no technical knowledge can engage with and perform the sonic environment using everyday language. Instead of confronting control

---

<sup>32</sup><https://docs.anthropic.com/en/docs/mcp>

panels or learning specific commands, they can simply describe what they want to hear, making sound art and experimental audio systems accessible to broader audiences while preserving the depth and complexity of the underlying synthesis.

**Wasmtime architecture** Wasmtime <sup>33</sup> is a high-performance WebAssembly runtime designed to execute WebAssembly modules outside the browser in an efficient and secure way. In particular it allows native applications to do JIT compilation starting from a pre-compiled wasm module. This makes it particularly interesting for real-time domains such as audio where performance and safety are critical. The wasmtime code is written in Rust, but offers C and C++ exports, so that pure C/C++ projects can easily link to wasmtime as a library.

The FAUST/wasmtime project explores how FAUST DSP code compiled to WebAssembly can be executed natively using Wasmtime. Its main goal is to make WebAssembly an alternative standalone deployment format for FAUST DSPs, without relying on JavaScript (used in a Node.js environment) or a web browser.

The libfaust library is used with its WebAssembly backend to produce a wasm module to be JIT compiled using wasmtime and executed on the fly. The wasm module is then wrapped with a specific architecture which connects it with the wasmtime machinery, exposing the internal API (initialization, compute, parameters access), accessing some needed functions from the wasmtime time (like math functions), and defining the memory block to be shared between the wasm module and the execution runtime.

The final API is exposed in a factory/instance model, where the DSP instance pointer can be used with already developed audio and controller managers, part of the FAUST general architecture machinery.

Two concrete use-cases have been developed:

- **faustwasmtime**: using JACK as the audio layer and accessing the control parameters with a GTK based GUI or a HTTPD controller.
- **faustbench-wasmtime**: to benchmark the DSP CPU consumption.

**faust2wwise tool** **faust2wwise** aimed to integrate the FAUST programming language with Audiokinetic's Wwise <sup>34</sup>, the multi-platform industry standard audio middleware in game development. This framework allows sound designers to set up audio environments, explore spatial audio, interactive music and real-time synthesis. The project implemented 2 different and complementary use-cases:

- a **faust2wwise** tool for statically compiling FAUST DSP code into Wwise modules, to be loaded in the Wwise environment.
- a plugin that embeds the libfaust and LLVM JIT compiler for dynamic FAUST DSP live-coding within Wwise, allows to explore different FAUST DSP programs at runtime.

Drawing on experience in C++ audio programming, DSP techniques, and plugin development, a practical connection has been built that allows audio designers and programmers to use FAUST's DSP capabilities directly within their Wwise workflows.

**faust2clap tool** The **faust2clap** tool generates a fully working CLAP<sup>35</sup> plugin starting from a FAUST DSP program. The tool supports both statically compiled and dynamically written and compiled (hot reload) DSP code:

- in the static mode model, a given DSP program is wrapped with additional C++ code then compiled into a CLAP plugin binary. Control parameters are exposed to be usable by the host application (typically a DAW that would load the plugin and instantiate it to generate or transform an audio track. Synthesizer and effect are supported, as well as MIDI controllable polyphonic instruments.
- a dynamic plugin has also been developed. It compiles and reloads any .dsp program while it is running, so that the user does not need to build a new binary or close and reopen the plugin in the host. This approach makes testing and development faster, and allows experiments to be carried out in the same environment in which the final plugin will be used, for example in a DAW such as Reaper.

<sup>33</sup><https://wasmtime.dev>

<sup>34</sup><https://www.audiokinetic.com/en/wwise/overview>

<sup>35</sup>Clever Audio Plugin, <https://cleveraudio.org/>

**FAUST PWA project** The project [18] aimed at moving FAUST instruments deployed on iOS and Android platforms, coded with native frameworks and languages, to web-based audio applications, focusing on the use of WebAssembly, the Web Audio API, and Progressive Web Applications (PWA). Its main goal is to simplify the deployment and maintenance of real-time musical applications while preserving performance and interactivity comparable to native solutions.

Initially, FAUST applications such as SmartFaust and GameLan were developed as native apps for iOS and Android, relying on platform-specific audio backends and programming languages, like Objective C on iOS and Java on Android for GUI implementations. Although this approach ensured low-latency audio and effective use of device sensors, it required maintaining multiple codebases and complying with restrictive app store distribution and validation processes.

The emergence of WebAssembly and modern web standards enabled a move toward a single and cross-platform web architecture. By compiling FAUST DSP code to WebAssembly, applications can run directly in browsers with near-native performance, while benefiting from browser security, sandboxing, and permissions management.

To support this transition, the `faustwasm` package was introduced as a JavaScript/TypeScript library. It automates the generation of self-contained web applications, supports MIDI by analyzing DSP metadata, efficiently integrates sensor data usage, and allows the use of soundfiles, making it suitable for interactive musical systems.

Finally, the adoption of the Progressive Web Application model allows FAUST applications to be installed easily on mobile devices via URLs or QR codes, with offline support and secure access to sensors. This approach greatly improves accessibility and usability, opening new opportunities for artistic creation and music education using web-based FAUST tools.

## 8 Bilateral contracts and grants with industry

**Participants:** Florent de Dinechin, Stephane Letz, Romain Michon, Yann Orlarey, Tanguy Risset, Anastasia Volkova, Christine Solnon.

### 8.1 Bilateral contracts with industry

The PhD thesis of Benjamin Quiédeville, in collaboration with GRAME-CNCM, includes a support contract of 30,000€ for the duration of the thesis.

The PhD thesis of Orégane Desrentes, in collaboration with Kalray, includes a support contract of 47,500€ for the duration of the thesis.

Anastasia Volkova, co-advises an industrial PhD thesis with Valeuriad company and Nantes University on the subject of frugal Natural Language Processing systems. The collaboration includes a 25,000€ support transferred to Emeraude for 2023-2027.

Florent de Dinechin co-advises an industrial PhD thesis with Thales on the subject of Vision transformers and FPGAs. The collaboration includes a 45,000€ support for 2025-2028.

## 9 Partnerships and cooperations

### 9.1 International initiatives

#### 9.1.1 Associate Teams in the framework of an Inria International Lab or in the framework of an Inria International Program

##### MIRAGES

**Title:** Modeling and Immersive Rendering of Archaeoacoustics in Grotto Environments

**Duration:** 2025 -> 2027

**Coordinator:** Romain Michon and Chris Chafe (cc@ccrma.stanford.edu)

**Partner Institution(s):** • Center for Computer Research in Music and Acoustics (CCRMA), Stanford University, USA

**Inria contact:** Romain Michon

**Summary:** As computing power has increased since the 2000s, virtual acoustics based on impulse response measurement and convolution have made it possible to recreate the sound of real spaces with high fidelity, driving growing interest from virtual reality, cultural institutions, and immersive media. While stereo convolution reverberation is now commonplace, delivering truly immersive, real-time, multi-speaker virtual acoustics remains challenging due to cost, system complexity, and computational demands. The MIRAGES associate team addresses these challenges through the Chauvet Cave as a unique case study, aiming to restore the missing acoustical dimension of its public replica by developing more immersive rendering techniques, leveraging FPGA platforms for high-performance real-time processing, and implementing an accurate, interactive simulation of the cave. Building on advances from the FAST ANR project and the PLASMA associate team that preceded it, MIRAGES seeks to create scalable, low-cost systems capable of driving hundreds or thousands of speakers and supporting advanced spatial audio methods such as Wave-Field Synthesis, with broad implications for research, industry, and public-facing audio experiences.

## 9.2 National initiatives

### 9.2.1 ANR DIStrib

Interest around spatial audio has been booming in recent years. An increasingly high number of movie theaters, concert halls, Virtual Reality (VR) platforms in museums, attractions in amusement parks, etc. are equipped with advanced spatial audio systems involving a large number of speakers. The automotive industry has also recently shown interest in spatial audio for its applications in the context of active noise cancellation. When considering interactive applications (i.e., virtual acoustics, soundscape rendering in VR, noise canceling, speaker correction, speech intelligibility enhancement, etc.) involving real-time operations and the ability to reprogram/customize the system, managing a large number of individual audio channels requires a tremendous amount of computational power and incredibly high bandwidths, which current systems fail to provide. The norm to implement such systems is to rely on a centralized software-based approach: a powerful computer connected to one or multiple audio interfaces providing a limited number of audio outputs. In that case, the bottleneck is the computer's throughput and hence its ability to manage a large number of audio streams in parallel with potential computations applied to each of them.

The goal of DIStrib is to rethink the way we approach spatial audio systems by relying on a distributed computing approach leveraging the computational power of large Field-Programmable Gate Arrays (FPGA). In this system, each FPGA is in charge of computing the sound of a limited set of speakers. Conversely, the distributed approach allows for a very large number of speakers to be targeted. In order to reach this goal, multiple challenges ranging from transmitting audio streams to a large number of audio devices with perfect synchronicity to running complex audio Digital Signal Processing (DSP) algorithms on FPGAs must be tackled. We believe that such a system has the potential to be highly disruptive in the field of spatial audio. DIStrib is also the occasion to explore the potential of artificial intelligence in the context of immersive sound and virtual acoustics rendering by relying on emerging platforms such as embedded NPUs (Neural Processing Unit).

DIStrib is a 42 months JCJC ANR project that started in October 2025. Romain Michon is the Principal Investigator (PI) of DIStrib.

### 9.2.2 PEPR HoliGrail

A key challenge to reach the maximal efficiency is to match algorithms with the underlying hardware. As the end of Moore's law steadily approaches, hardware becomes increasingly heterogeneous, and the interplay between algorithms and hardware even more important. From this point of view, artificial intelligence algorithms are not efficient when run on commodity hardware such as microprocessors and GPUs. The last decade has seen a boom of accelerators for common inference tasks (first in line was Google's TPU, followed by many others), now embedded in many consumer products. These accelerators offer hardware that better

matches the algorithms, but are still far from achieving the “inference per joule” performance of the Human brain. Training deep neural networks (DNNs) is even less efficient, requiring orders of magnitude more computation operations than inference.

We believe that there is a significant room for improvements in both “inference per joule” and “training per joule”. The vision of this action is to create a synergy with the research on the foundations of AI frugality (as proposed in SHARP action) to propose cutting-edge methods that significantly improve the energy efficiency of both inference and training of a model. We will propose (i) more compact and efficient number representations that still maintain a quality of inference or training close to the reference, (ii) hardware-aware training algorithms that enhance certain types of sparsity (e.g., more structured), coding compactness (aggressive quantization, maximum entropy) and tensor transformations. Most state-of-the-art solutions are agnostic of the hardware they run on. By taking advantage of this interplay between the hardware and the algorithms, we can achieve breakthroughs beyond current solutions, in particular by developing (iii) efficient hardware mechanisms, especially optimized to take advantage of sparsity, extreme quantization and ad-hoc number representations, together with (iv) compiler optimizations, to demonstrate the effectiveness of the proposed methods. Our approaches are holistic in the sense that they will jointly optimize the whole computing stack of AI, i.e., at the algorithm, arithmetic, compiler and hardware levels.

PEPR HOLIGRAIL kicked off in March 2024 and will end in 2029 (extension has been anticipated). The project combines following partners: Inria/IRISA Taran (Univ. Rennes, Inria, CNRS), List /LIAE (Université Paris Saclay, CEA), Inria Corse (Université Grenoble Alpes), TIMA SLS (CNRS, Université Grenoble Alpes), CITI lab (INSA Lyon / Inria), List / LVML (Université Paris Saclay, CEA). The scientific leaders are Olivier Bichler (CEA List) and Olivier Sentieys (Inria Taran).

## 10 Dissemination

**Participants:** Tanguy Risset, Romain Michon, Pierre Cochard, Florent de Dinechin, Anastasia Volkova, Stéphane Letz, Christine Solnon.

### 10.1 Promoting scientific activities

#### 10.1.1 Scientific events: organisation

##### General chair, scientific chair

- Romain Michon served as General Chair and Stéphane Letz as Technical Chair for the JIMLAC-25 conference which was mainly organized by Emeraude (see §5.1).

#### 10.1.2 Scientific events: selection

##### Member of the conference program committees

- Romain Michon served in the program committees of SMC (Sound and Music Computing conference), DAFx (Digital Audio Effects conference), NIME (New Interfaces for Musical Expression conference), ICMC (International Computer Music Conference).
- Tanguy Risset served in the program committees of ASAP 2025 and DATE 2025 (Design Automation and Test in Europe), on track " Architectural and Microarchitectural Design".
- Christine Solnon served in the programme committee of [CPAIOR 2025](#), and as senior PC member for [CP 2025](#).
- Florent de Dinechin served in the programme committees of [ARITH 2025](#), [ASAP 2025](#), [FCCM 2025](#), [FPL 2025](#).
- Anastasia Volkova served in the programme committees of [ARITH 2025](#) and [ASAP 2025](#)

### 10.1.3 Journal

#### Member of the editorial boards

- Christine Solnon is associate editor of **JAIR** and **Constraints**.

### 10.1.4 Invited talks

- Romain Michon was an invited keynote speaker at the Lambda Days conference<sup>36</sup> which took place in Krakow on June 12-13, 2025.
- Christine Solnon was an invited keynote speaker at the **31st International Conference on Principles and Practice of Constraint Programming (CP 2025)**, the **14th IAPR-TC15 Workshop on Graph-based Representations in Pattern Recognition (GBR 2025)**, and the **GreenDays 2025**
- Florent de Dinechin was an invited keynote speaker at the **ARITH 2025** conference and at the **national meeting of the GDR SoC2**. He was also invited to present one of his 2007 papers in a retrospective session of **ASAP 2025: ASAP since the Millennium: A Selection of Most Representative Papers [13]**.

### 10.1.5 Leadership within the scientific community

- Christine Solnon is member of the executive committee of **GDR RADIA**.
- Anastasia Volkova is a part of the organizing committee for the "Embedded HPC" axis of GDR SoC2.
- Romain Michon is the president of the Sound and Music Computing (SMC) Network that steers the planning of the SMC conference, one of the most prestigious scientific event in the sound and music technology community.
- Romain Michon is a board member the Association Française d'Informatique Musicale (AFIM) and of GRAME-CNCM, both as secretary.

## 10.2 Teaching - Supervision - Juries - Educational and pedagogical outreach

- Tanguy Risset is professor at the Telecommunications Department of Insa Lyon.
- Florent de Dinechin is a professor at the Computer Science Department of Insa Lyon and head of the 4th year. He also teaches computer architecture at ENS-Lyon.
- Christine Solnon is a professor at the Computer Science Department of Insa Lyon.
- Romain Michon is a part-time associate professor at the Telecommunications Department of Insa Lyon.
- Romain Michon teaches 2 courses as part of the RIM/RAN Masters Program at the université of Saint-Étienne.
- Romain Michon teaches 2 one week workshops at Aalborg University in Copenhagen every year.
- Stephane Letz teaches 1 course as part of the RIM/RAN Masters Program at the université of Saint-Étienne.
- Anastasia Volkova teaches 1 course at the Telecommunications department of INSA and offers multiple semester projects.

---

<sup>36</sup>[www.lambdadays.org/lambdadays2025](http://www.lambdadays.org/lambdadays2025)

### 10.2.1 Juries

- Anastasia Volkova was in the PhD jury of:
  - Sami Ben Ali (Univ. de Rennes), as examiner.
- Romain Michon was in the PhD jury of:
  - David Fiero (U. Paris 8), as reviewer,
  - Alexandre d’Hooge (U. of Lille), as reviewer,
  - Daniel Picciola (U. Paris 8), as reviewer.
- Christine Solnon was in the PhD jury of:
  - Jean-Baptiste Sciau (IMT Albi) as president,
  - Jorge Mortes Alcaraz (IMT Atlantique) as president,
  - Stevan Stanovic (ENSICAEN) as president,
  - Guillaume Ghienne (IMT Atlantique) as president,
  - Léon Fauste (U. Grenoble Alpes) as co-director,
  - Fei Ge (ENTPE) as president,
  - Bachtiar Herdianto (IMT Atlantique) as examiner,
  - Julien Rouzot (U. Toulouse) as examiner,
  - Djawad Bekkoucha (U. Caen) as examiner,
  - Alexandre Ronsain (ENAC) as reviewer and president,
  - Fayad Ali Banna (U. Saint Etienne) as president,
  - Antoine Lhomme (U. Grenoble Alpes) as president,
  - Arsène Marzorati (INSA Lyon) as president.
- Florent de Dinechin was in the PhD jury of:
  - Cheolyong Bae (Linköping University, Suède),
  - Jonas Bertels (KULeuven, Belgique),
  - Quentin Milot (Université de Rennes).

## 10.3 Popularization

### 10.3.1 Productions (articles, videos, podcasts, serious games, ...)

Romain Michon served in the scientific committee of the “Ça résonne<sup>37</sup>” exhibit at the Maison des Mathématiques et de l’Informatique (MMI) of Lyon University.

Christine Solnon wrote a paper on women in computation [28], and a column for La Recherche [39].

### 10.3.2 Participation in Live events

Several concerts were organized during the JIMLAC-25 conference which was mainly organized by Emeraude, the concert list can be seen on JIMLAC program <sup>38</sup>

<sup>37</sup>[mmi.universite-lyon.fr/pour-le-grand-public/par-activites/exposition/](http://mmi.universite-lyon.fr/pour-le-grand-public/par-activites/exposition/)

<sup>38</sup><https://jimlac25.inria.fr/program/>



Figure 14: Logo of the Grame/Emeraude demonstration at Moduland

### 10.3.3 Others science outreach relevant activities

- Benjamin Quideville proposed a demonstration of the *spacebar* [20] at the "fête de la science" in October 2025.
- Romain Michon actively took part in the Chiche! program in 2025.
- Christine Solnon actively took part in the Chiche! program in 2025, and she gave a conference on AI for the staff of ENSSIB
- Anastasia Volkova actively participated in Comptoire des Sciences and Déclic programs in 2025.
- Louis Ledoux participated in the first edition of the Moduland electronic arts festival, organized by Le Séquenceur, by representing the Émeraude research team (INSA–Inria–GRAME). He presented an experimental electronic instrument during a long public demonstration, leading to numerous exchanges with festival attendees and practitioners. This highly successful inaugural edition offered strong visibility and contributed to the promotion of the team’s work, emphasizing its concrete artistic realizations and the close articulation between research, digital instrument design, and contemporary electronic art practices. <sup>39</sup> <sup>40</sup>

## 11 Scientific production

### 11.1 Major publications

- [1] F. de Dinechin and M. Kumm. *Application-Specific Arithmetic*. Springer International Publishing, 2024. DOI: [10.1007/978-3-031-42808-1](https://doi.org/10.1007/978-3-031-42808-1). URL: <https://inria.hal.science/hal-04715553>.

<sup>39</sup><https://www.instagram.com/p/DQEMOxiGCz/>

<sup>40</sup><https://www.periscope-lyon.com/concerts/moduland/>

## 11.2 Publications of the year

### International journals

- [2] L. Ledoux, P. Cochard and F. de Dinechin. ‘Towards Optimized Arithmetic Circuits with MLIR’. In: *Works in Progress in Embedded Computing Journal* 11.1 (2nd Sept. 2025), p. 4. DOI: [10.64552/wipiec.v11i1.90](https://doi.org/10.64552/wipiec.v11i1.90). URL: <https://hal.science/hal-05385229> (cit. on p. 18).
- [3] R. Michon, M. Ducceschi, P. Cochard, T. Skare, C. J. Webb and R. Russo. ‘Evaluating CPU, GPU, and FPGA Performance In the Context of Modal Reverberation: a Comparative Analysis’. In: *Frontiers in Signal Processing* 5 (4th Apr. 2025). DOI: [10.3389/frsip.2025.1522604](https://doi.org/10.3389/frsip.2025.1522604). URL: <https://hal.science/hal-05319252>.
- [4] X. Peng, F. Schwarzentruher, O. Simonin and C. Solnon. ‘Spanning-tree based coverage for a tethered robot’. In: *IEEE Robotics and Automation Letters* (9th Jan. 2025), pp. 1–8. URL: <https://hal.science/hal-04877205>. In press (cit. on p. 25).
- [5] O. Rifki and C. Solnon. ‘On the Phase Transition of the Euclidean Travelling Salesman Problem with Time Windows’. In: *Journal of Artificial Intelligence Research* 82 (7th Apr. 2025), pp. 2167–2188. DOI: [10.1613/jair.1.18334](https://doi.org/10.1613/jair.1.18334). URL: <https://hal.science/hal-05016544> (cit. on p. 25).
- [6] C. Solnon. ‘LAD2025, A constraint-based solver for the subgraph isomorphism problem’. In: *Artificial Intelligence (AIJ)* 352 (Mar. 2026), p. 104474. DOI: [10.1016/j.artint.2025.104474](https://doi.org/10.1016/j.artint.2025.104474). URL: <https://hal.science/hal-05464475>.

### Invited conferences

- [7] C. Solnon. ‘Anytime and exact search for planning problems: How to explore a DP-based state transition graph with A\*, CP and LS?’ In: 31st International Conference on Principles and Practice of Constraint Programming (CP 2025). Vol. 41. 2. Glasgow, United Kingdom, 2025. DOI: [10.4230/LIPIcs.CP.2025.2](https://doi.org/10.4230/LIPIcs.CP.2025.2). URL: <https://hal.science/hal-05104519> (cit. on p. 25).

### International peer-reviewed conferences

- [8] B. Barbe, R. Bouarah, F. de Dinechin and A. Volkova. ‘Reducing the Hardware Gap for Custom Accelerators through Quantization Aware Training’. In: *AccML 2026 - 8th Workshop on Accelerated Machine Learning*. Krakow (Cracovie), Poland, 27th Jan. 2026. URL: <https://inria.hal.science/hal-05466817>.
- [9] B. Barbe, X. Peng, A. Volkova and F. de Dinechin. ‘Towards optimal reconfigurable constant multipliers’. In: *28th Euromicro Conference on Digital System Design (DSD)*. 28th Conference on Digital System Design (DSD). Salerno, Italy: IEEE, 10th Sept. 2025, pp. 418–425. DOI: [10.1109/DSD67783.2025.00064](https://doi.org/10.1109/DSD67783.2025.00064). URL: <https://hal.science/hal-05094796> (cit. on pp. 18, 21, 23).
- [10] R. Bouarah and F. de Dinechin. ‘Hardware Fixed-Point 2D and 3D norms’. In: *32nd IEEE International Symposium on Computer Arithmetic - ARITH 2025*. El Paso, Texas, United States, 5th May 2025. URL: <https://inria.hal.science/hal-04986776>.
- [11] T. Cantaloube, X. Peng, C. Solnon and A. Volkova. ‘A new Constraint Programming model for the Multiple Constant Multiplication’. In: *ModRef 2025 - 24th workshop on Constraint Modelling and Reformulation*. Glasgow, United Kingdom, 2025. URL: <https://hal.science/hal-05251229> (cit. on pp. 18, 25).
- [12] O. Desrentes, B. Dupont de Dinechin and F. de Dinechin. ‘Double-Word Decomposition in a Combined FP16, BF16 and FP32 Dot Product Add Operator’. In: *ARITH 2025 - IEEE 32nd International Symposium on Computer Arithmetic*. El-Paso, Texas, United States, 4th May 2025. URL: <https://inria.hal.science/hal-04982397> (cit. on pp. 17, 23).
- [13] F. de Dinechin. ‘RETROSPECTIVE: Table-based polynomials for fast hardware function evaluation’. In: *36th IEEE International Conference on Application-specific Systems, Architectures and Processors*. ASAP 2025 - 36th IEEE International Conference on Application-specific Systems, Architectures and Processors. Vancouver (British Columbia), Canada, 28th July 2025. URL: <https://inria.hal.science/hal-05139896> (cit. on p. 34).

- [14] C. Gernigon, X. Pillet, A. Volkova and R. Dufour. ‘Training for Mixed-Precision Integer Weights, Activations and Embeddings in BERT’. In: *DASIP 2026: Workshop on Design and Architectures for Signal and Image Processing*. Krakow (Cracovie), Poland, 2026. URL: <https://inria.hal.science/hal-05322672> (cit. on p. 22).
- [15] R. Jeunehomme, R. Michon, T. Risset, P. Cochard and S. Letz. ‘TOWARDS REAL-TIME AURALIZATION ON FPGAS’. In: *SMC 2025 - 22nd Sound and Music Computing Conference*. Graz, Austria: Institute of Electronic Music, Acoustics, University of Music and Performing Arts Graz, 2025. DOI: [10.5281/zenodo.15836298](https://doi.org/10.5281/zenodo.15836298). URL: <https://inria.hal.science/hal-05298956> (cit. on p. 20).
- [16] H. Kadi, M. Devidal, R. Michon and J. Erhani. ‘Real-Time Musical Instruments Recognition for Scenography Purposes’. In: *Proceedings of the 19th Linux Audio Conference*. Linux Audio Conference 2025. Lyon, France, 25th June 2025. URL: <https://hal.science/hal-05096057>.
- [17] L. Ledoux. ‘Design-Space Exploration of Serialized Floating-Point Division for DLP Architectures’. In: *DSD 2025 - 28th Euromicro Conference on Digital System Design*. Salerno, Italy, 10th Sept. 2025. URL: <https://hal.science/hal-05385247>.
- [18] S. Letz and J. Ribeau. ‘Applications audio web avec Faust : innovations techniques au service de la transmission’. In: *Proceedings of the 32th Journées d’Informatique Musicale*. Journées d’Informatique Musicale. Lyon, France, 23rd June 2025. URL: <https://hal.science/hal-05102348> (cit. on p. 31).
- [19] X. Peng and C. Solnon. ‘BFS-Based Canonical Codes for Generating Graphs with Constraint Programming’. In: *31st International Conference on Principles and Practice of Constraint Programming (CP 2025)*. Vol. 340. 41. Glasgow, United Kingdom, Aug. 2025. DOI: [10.4230/LIPIcs.CP.2025.41](https://doi.org/10.4230/LIPIcs.CP.2025.41). URL: <https://hal.science/hal-05104512> (cit. on p. 25).
- [20] B. Quiédeville, R. Michon, T. Risset and S. Letz. ‘The Space Bar: An Embedded WFS Sound System’. In: *Proceedings of the 32th Journées d’Informatique Musicale*. JIMLAC 2025 - Journées de l’Informatique Musicale. Lyon, France, 23rd June 2025. URL: <https://hal.science/hal-05102322> (cit. on p. 36).
- [21] T. Rushton, R. Michon and T. Risset. ‘All Together Now: A Synchronous Platform for Distributed Spatial Audio’. In: *22nd Sound and Music Computing Conference (SMC2025)*. Graz, Austria: Institute of Electronic Music, Acoustics, University of Music and Performing Arts Graz, July 2025. DOI: [10.5281/zenodo.15838463](https://doi.org/10.5281/zenodo.15838463). URL: <https://inria.hal.science/hal-05209243> (cit. on p. 19).
- [22] L. Valentin, J. Abel, J. Berger, C. Chafe, J. Chowning, C. Fritz, N. Farzaneh, M. Kolar, R. Michon, S. Martin, P. Svensson and M. Wright. ‘Measurement and Auralization of Paleoacoustics landscapes in Chauvet Cave’. In: *Proceedings of Forum Acusticum*. 11th Convention of the European Acoustics Association Forum Acusticum / EuroNoise 2025. Forum Acusticum / EuroNoise 2025. Málaga, Spain: European Acoustics Association, 25th Dec. 2025, pp. 3855–3864. DOI: [10.61782/fa.2025.0546](https://doi.org/10.61782/fa.2025.0546). URL: <https://cnrs.hal.science/hal-05450067> (cit. on p. 20).

### National peer-reviewed Conferences

- [23] O. Bellenguez, N. Brauner, C. Solnon and A. Tsoukias. ‘Enseigner les enjeux sociétaux et environnementaux des algorithmes à travers un outil de navigation routière’. In: *ROADEF 2025 : 26ème congrès annuel de la Société Française de Recherche Opérationnelle et d’Aide à la Décision*. Champs sur Marne, France, 26th Feb. 2025. URL: <https://hal.science/hal-05006132>.

### Conferences without proceedings

- [24] A. Delage and C. Solnon. ‘Un modèle de théorie des jeux pour l’étude de stratégies de réduction de l’impact environnemental humain’. In: *26ème congrès annuel de la société française de recherche opérationnelle et d’aide à la décision (ROADEF)*. Marne-la-Vallée, France, 26th Feb. 2025, pp. 226–227. DOI: [10.1086/226707](https://doi.org/10.1086/226707). URL: <https://hal.science/hal-05321562>.

- [25] O. Desrentes, F. de Dinechin and B. Dupont de Dinechin. ‘Reciprocal Square Root Accelerated Using Hardware and Software Techniques’. In: RAIM Meeting 2025: 16th Rencontres de l’Arithmétique en Informatique Mathématique – A Tribute to Jean-Michel Muller. Lyon, France, 3rd Nov. 2025. URL: <https://inria.hal.science/hal-05461486>.
- [26] R. Fontaine and C. Solnon. ‘Large-scale experiments on the Traveling Salesman Problem with Time Windows’. In: ROADEF 2025 - 26ème congrès annuel de la Société Française de Recherche Opérationnelle et d’Aide à la Décision. Champs-sur-Marne, France, Feb. 2025. URL: <https://hal.science/hal-05322713>.
- [27] X. Pillet, C. Gernigon, A. Volkova, R. Dufour, A. Granet and N. Greffard. ‘Quantization-aware training: a tradeoff between training and fine-tuning for domain-specific language models’. In: AccML 2026 - 8th Workshop on Accelerated Machine Learning. Krakow (Cracovie), Poland, 26th Jan. 2026. URL: <https://inria.hal.science/hal-05489187>.

### Scientific book chapters

- [28] C. Solnon. ‘Le calcul, une histoire de femmes’. In: *Le calcul à découvert*. CNRS Editions, 2025. URL: <https://hal.science/hal-04996437> (cit. on p. 35).

### Edition (books, proceedings, special issue of a journal)

- [29] *Proceedings of the 32nd Journées d’Informatique Musicale*. Journées d’Informatique Musicale. Lyon, France, 17th July 2025. URL: <https://hal.science/hal-05168345>.
- [30] R. Michon and P. Lecomte, eds. *Proceedings of the 19th Linux Audio Conference*. Proceedings of the 19th Linux Audio Conference. Lyon, France, 25th June 2025. URL: <https://hal.science/hal-05194352>.

### Reports & preprints

- [31] O. Bellenguez, N. Brauner, C. Solnon and A. Tsoukias. *Integrating ethical, societal and environmental issues into algorithm design courses*. 19th Jan. 2026. URL: <https://hal.science/hal-05414147>.
- [32] O. Bellenguez, N. Brauner, C. Solnon and A. Tsoukias. *Integrating ethical, societal and environmental issues into algorithm design courses*. 19th Jan. 2026. URL: <https://hal.science/hal-05046104>.
- [33] R. Bouarah, B. Barbe, A. Volkova and F. de Dinechin. *KANHard: Training Hardware-Friendly Kolmogorov-Arnold Networks*. 28th Nov. 2025. URL: <https://hal.science/hal-05388006> (cit. on p. 22).
- [34] P. Cochard and L. Ledoux. *FloPoCo and MLIR: A Multi-Level Compilation Framework for Many Intents*. Inria Lyon; INSA lyon; CITI - Centre d’Innovation en Télécommunications et Intégration de services, 18th Sept. 2025. URL: <https://hal.science/hal-05412130> (cit. on p. 18).
- [35] M. L. Reyna Cruz, K. Ruiz-Rohena, Y. I. Guel, H. Salgado, L. Taldir, E. Pena Ramos, N. Cervantes, T. Rivero, M. Ceberio, C. Lauter and A. Volkova. *Contribution to Error Analysis of Deep Neural Networks: Case of the Activation Functions*. 14th Nov. 2025. URL: <https://inria.hal.science/hal-05367563>.

### Other scientific publications

- [36] P. Cochard, L. Forget, F. de Dinechin and L. Ledoux. ‘Towards Multi-Level Arithmetic Optimizations’. In: EuroLLVM 2025 -. Berlin, Germany, 14th Apr. 2025. URL: <https://hal.science/hal-05063466>.
- [37] L. Ledoux, F. de Dinechin and P. Cochard. ‘Arithmetic Lowering with Emerald-MLIR: Bridging Tensor and DSP Kernels to Silicon Datapaths’. In: PEPR IA Embarquée Workshop. Aussois (France), France, Jan. 2026. URL: <https://hal.science/hal-05489427>.

- [38] M. Popoff, R. Michon, T. Risset, P. Cochard, L. Ledoux, S. Letz, Y. Orlarey and F. de Dinechin. ‘Frugalité et conception de circuits pour le traitement du signal audio numérique’. In: *Revue Francophone d’Informatique et Musique*. Frugalité, pérennité et création 11 (Dec. 2025). doi: [10.56698/rfim.961](https://doi.org/10.56698/rfim.961). URL: <https://hal.science/hal-05489376> (cit. on p. 18).

### Scientific popularization

- [39] C. Solnon. ‘Peut-on remplacer le raisonnement humain par des algorithmes?’ In: *La Recherche* 580 (Jan. 2025). URL: <https://hal.science/hal-04881777> (cit. on p. 35).

### 11.3 Cited publications

- [40] L. Aksoy, E. da Costa, P. Flores and J. Monteiro. ‘Exact and Approximate Algorithms for the Optimization of Area and Delay in Multiple Constant Multiplications’. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27.6 (2008), pp. 1013–1026 (cit. on p. 12).
- [41] P. R. Benois, P. Nowak and U. Zölzer. ‘Fully Digital Implementation of a Hybrid Feedback Structure for Broadband Active Noise Control in Headphones’. In: *2017 Proceedings of the 24th International Congress on Sound and Vibration*. 2017 (cit. on p. 15).
- [42] B. Betgen and M.-A. Galland. ‘A New Hybrid Active/Passive Sound Absorber with Variable Surface Impedance’. In: *Mechanical systems and signal processing* 25.5 (2011), pp. 1715–1726 (cit. on p. 16).
- [43] T. Bollaert. ‘Catapult Synthesis: A Practical Introduction to Interactive C Synthesis’. In: *High-Level Synthesis: From Algorithm to Digital Circuit*. Ed. by P. Coussy and A. Morawiec. Dordrecht: Springer Netherlands, 2008, pp. 29–52 (cit. on p. 11).
- [44] A. Böttcher, M. Kumm and F. de Dinechin. ‘Resource Optimal Squarers for FPGAs’. In: *International Conference on Field-Programmable Logic and Applications (FPL)*. Belfast, United Kingdom: IEEE, Aug. 2022. doi: [10.1109/FPL57034.2022.00018](https://doi.org/10.1109/FPL57034.2022.00018). URL: <https://inria.hal.science/hal-03922311> (cit. on p. 13).
- [45] [SW] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne and Q. Zhang, *JAX: composable transformations of Python+NumPy programs* version 0.3.13, 2018. URL: <http://github.com/jax-ml/jax> (cit. on p. 28).
- [46] P. Brinkmann, P. Kirn, R. Lawler, C. McCormick, M. Roth and H.-C. Steiner. ‘Embedding PureData with libpd’. In: *Proceedings of the Pure Data Convention*. Vol. 291. Citeseer. 2011 (cit. on p. 7).
- [47] N. Brunie, F. de Dinechin, M. Istoan, G. Sergent, K. Illyes and B. Popa. ‘Arithmetic Core Generation Using Bit Heaps’. In: *Field-Programmable Logic and Applications*. Sept. 2013 (cit. on p. 12).
- [48] Z. Buckley and K. Carlson. ‘Towards a Framework for Composition Design for Music-Led Virtual Reality Experiences’. In: *Proceedings of the 2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. Osaka, Japan, 2019 (cit. on p. 15).
- [49] A. Camci and R. Hamilton. ‘Audio-first VR: New perspectives on musical experiences in virtual environments’. In: *Journal of New Music Research* (2020) (cit. on p. 15).
- [50] J. Choi, M. Kang, Y. Kim, C.-H. Kim and J.-M. Kim. ‘Design space exploration in many-core processors for sound synthesis of plucked string instruments’. In: *Journal of Parallel and Distributed Computing* 73.11 (2013), pp. 1506–1522 (cit. on p. 8).
- [51] L. Chu. ‘Haptic feedback in computer music performance’. In: *Proceedings of International Computer Music Conference*. Vol. 96. 1996, pp. 57–58 (cit. on p. 15).
- [52] Y. Deng, D. Dragna, M.-A. Galland and A. Alomar. ‘Comparison of Three Numerical Methods for Acoustic Propagation in a Lined Duct with Flow’. In: *25th AIAA/CEAS Aeroacoustics Conference*. 2019, p. 2658 (cit. on p. 16).

- [53] O. Desrentes and F. de Dinechin. ‘Using integer linear programming for correctly rounded multipartite architectures’. In: *FPT 2022 - International Conference on Field Programmable Technology*. Hong Kong, China, Dec. 2022. URL: <https://inria.hal.science/hal-03844218> (cit. on p. 13).
- [54] F. de Dinechin. ‘Reflections on 10 years of FloPoCo’. In: *26th IEEE Symposium of Computer Arithmetic (ARITH)*. June 2019 (cit. on p. 11).
- [55] F. de Dinechin, L. Forget, J.-M. Muller and Y. Uguen. ‘Posits: the good, the bad and the ugly’. In: *Conference on Next-Generation Arithmetic*. 2019, pp. 1–10 (cit. on p. 11).
- [56] F. de Dinechin and M. Istoan. ‘Hardware implementations of fixed-point Atan2’. In: *22nd IEEE Symposium of Computer Arithmetic (ARITH-22)*. 2015, pp. 34–41 (cit. on p. 12).
- [57] F. de Dinechin, M. Istoan and G. Sergent. ‘Fixed-Point Trigonometric Functions on FPGAs’. In: *SIGARCH Computer Architecture News* 41.5 (2013), pp. 83–88 (cit. on p. 12).
- [58] F. de Dinechin and M. Kumm. *Application-specific arithmetic*. Springer, to appear, 2021 (cit. on p. 12).
- [59] F. Dinechin, P. Quinton and T. Risset. ‘Structuration of the ALPHA language’. In: Nov. 1995, pp. 18–24. DOI: [10.1109/PMMP.1995.504337](https://doi.org/10.1109/PMMP.1995.504337) (cit. on p. 14).
- [60] S. Elliott. *Signal Processing for Active Control*. Elsevier, 2000 (cit. on p. 8).
- [61] J. Engel, L. ( Hantrakul, C. Gu and A. Roberts. ‘DDSP: Differentiable Digital Signal Processing’. In: *Proceedings of the International Conference on Learning Representations*. 2020 (cit. on p. 14).
- [62] D. Fober, Y. Orlarey and S. Letz. ‘FAUST Architectures Design and OSC Support.’ In: *International Conference on Digital Audio Effects*. Ed. by IRCAM. Paris, France, 2011, pp. 231–216. URL: <https://hal.archives-ouvertes.fr/hal-02158816> (cit. on p. 14).
- [63] L. Hascoet and V. Pascual. ‘The Tapenade automatic differentiation tool: Principles, model, and specification’. In: *ACM Trans. Math. Softw.* 39.3 (May 2013). DOI: [10.1145/2450153.2450158](https://doi.org/10.1145/2450153.2450158). URL: <https://doi.org/10.1145/2450153.2450158> (cit. on p. 28).
- [64] Y. He, T. N. Sainath, R. Prabhavalkar, I. McGraw, R. Alvarez, D. Zhao, D. Rybach, A. Kannan, Y. Wu, R. Pang, Q. Liang, D. Bhatia, Y. Shangguan, B. Li, G. Pundak, K. C. Sim, T. Bagby, S.-Y. Chang, K. Rao and A. Gruenstein. ‘Streaming End-to-end Speech Recognition For Mobile Devices’. In: *CoRR* abs/1811.06621 (2018). arXiv: [1811.06621](https://arxiv.org/abs/1811.06621). URL: <http://arxiv.org/abs/1811.06621> (cit. on p. 14).
- [65] Y. Hu, M.-A. Galland and K. Chen. ‘Acoustic Transmission Performance of Double-Wall Active Sound Packages in a Tube: Numerical/Experimental Validations’. In: *Applied acoustics* 73.4 (2012), pp. 323–337 (cit. on p. 16).
- [66] M. Innes. ‘Don’t Unroll Adjoint: Differentiating SSA-Form Programs’. In: 2019. arXiv: [1810.07951](https://arxiv.org/abs/1810.07951) [cs.PL]. URL: <https://arxiv.org/abs/1810.07951> (cit. on p. 28).
- [67] W. Jung, S. J. Elliott and J. Cheer. ‘Local Active Control of Road Noise inside a Vehicle’. In: *Mechanical Systems and Signal Processing* 121 (2019), pp. 144–157 (cit. on p. 16).
- [68] R. Kastner, J. Matai and S. Neuendorffer. ‘Parallel Programming for FPGAs’. In: *ArXiv e-prints* (May 2018). arXiv: [1805.03648](https://arxiv.org/abs/1805.03648) (cit. on p. 11).
- [69] J. Knowles and E. Olcayto. ‘Coefficient Accuracy and Digital Filter Response’. In: *IEEE Transactions on Circuit Theory* 15.1 (1968), pp. 31–41 (cit. on p. 12).
- [70] D. M. Kodek. ‘LLL algorithm and the optimal finite wordlength FIR design’. In: *IEEE Transactions on Signal Processing* 60.3 (2012), pp. 1493–1498 (cit. on p. 12).
- [71] M. Kumm. ‘Multiple Constant Multiplication Optimizations for Field Programmable Gate Arrays’. PhD thesis. Wiesbaden: Springer Wiesbaden, Oct. 2015 (cit. on p. 12).
- [72] M. Kumm. ‘Optimal Constant Multiplication using Integer Linear Programming’. In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. 2018 (cit. on p. 12).
- [73] N. Lago and F. Kon. ‘The Quest for Low Latency’. In: *Proceedings of the International Computer Music Conference (ICMC-04)*. Miami, USA, 2004 (cit. on p. 8).

- [74] M. Lanham. *Game Audio Development with Unity 5.X*. New York, USA: Packt Publishing Ltd., 2017 (cit. on p. 15).
- [75] S. Letz, S. Denoux, Y. Orlarey and D. Fober. ‘Faust audio DSP language in the Web’. In: *Linux Audio Conference*. Mainz, Germany, 2015, pp. 29–36. URL: <https://hal.archives-ouvertes.fr/hal-02159002> (cit. on p. 13).
- [76] S. Letz, Y. Orlarey and D. Fober. ‘FAUST Domain Specific Audio DSP Language Compiled to WebAssembly’. In: *Companion Proceedings of the The Web Conference 2018*. WWW ’18. Lyon, France: International World Wide Web Conferences Steering Committee, 2018, pp. 701–709. DOI: [10.1145/3184558.3185970](https://doi.org/10.1145/3184558.3185970) (cit. on p. 13).
- [77] S. Letz, Y. Orlarey and D. Fober. ‘Work Stealing Scheduler for Automatic Parallelization in Faust’. In: *Linux Audio Conference*. Ed. by LAC. Utrecht, Netherlands, 2010. URL: <https://hal.archives-ouvertes.fr/hal-02158924> (cit. on p. 13).
- [78] B. Mazeaud and M.-A. Galland. ‘A Multi-Channel Feedback Algorithm for the Development of Active Liners to Reduce Noise in Flow Duct Applications’. In: *Mechanical Systems and Signal Processing* 21.7 (2007), pp. 2880–2899 (cit. on p. 16).
- [79] A. McPherson and V. Zappi. ‘An environment for submillisecond-latency audio and sensor processing on BeagleBone Black’. In: *Proceedings of the Audio Engineering Society Convention*. Warsaw, Poland, 2015 (cit. on p. 10).
- [80] M. Melon, P. Herzog, A. Sittel and M.-A. Galland. ‘One Dimensional Study of a Module for Active/Passive Control of Both Absorption and Transmission’. In: *Applied Acoustics* 73.3 (2012), pp. 234–242 (cit. on p. 16).
- [81] R. Michon, Y. Orlarey, S. Letz and D. Fober. ‘Real Time Audio Digital Signal Processing With Faust and the Teensy’. In: *Proceedings of the Sound and Music Computing Conference (SMC-19), Malaga, Spain*. 2019 (cit. on pp. 7, 10).
- [82] R. Michon, D. Overholt, S. Letz, Y. Orlarey, D. Fober and C. Dumitrascu. ‘A Faust Architecture for the ESP32 Microcontroller’. In: *Accepted to the Sound and Music Computing Conference (SMC-20)*. Turin, Italy, 2020 (cit. on pp. 7, 10).
- [83] R. Michon, J. Smith and Y. Orlarey. ‘New Signal Processing Libraries for Faust’. In: *Linux Audio Conference*. Ed. by V. Ciciliato, Y. Orlarey and L. Pottier. Saint-Etienne, France: CIEREC, 2017, pp. 83–87 (cit. on pp. 13, 26).
- [84] W. Moses and V. Churavy. ‘Instead of Rewriting Foreign Code for Machine Learning, Automatically Synthesize Fast Gradients’. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 12472–12485. URL: <https://proceedings.neurips.cc/paper/2020/file/9332c513ef44b682e9347822c2e457ac-Paper.pdf> (cit. on p. 28).
- [85] J.-M. Muller, N. Brunie, F. de Dinechin, C.-P. Jeannerod, M. Joldes, V. Lefvre, G. Melquiond, N. Revol and S. Torres. *Handbook of Floating-Point Arithmetic*. 2nd. Birkhäuser Basel, 2018 (cit. on p. 11).
- [86] R. Nane, V. Sima, C. Pilato, J. Choi, B. Fort, A. Canis, Y. T. Chen, H. Hsiao, S. Brown, F. Ferrandi, J. Anderson and K. Bertels. ‘A Survey and Evaluation of FPGA High-Level Synthesis Tools’. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35.10 (Oct. 2016), pp. 1591–1604 (cit. on p. 10).
- [87] Y. Orlarey, D. Fober and S. Letz. ‘An Algebra for Block Diagram Languages’. In: *International Computer Music Conference*. Ed. by ICMA. Gothenburg, Sweden, 2002, pp. 542–547. URL: <https://hal.archives-ouvertes.fr/hal-02158932> (cit. on p. 26).
- [88] Y. Orlarey, S. Letz and D. Fober. ‘New Computational Paradigms for Computer Music’. In: Paris, France: Delatour, 2009. Chap. Faust: an Efficient Functional Approach to DSP Programming (cit. on p. 7).
- [89] F. Pfeifle and R. Bader. ‘Real-time finite difference physical models of musical instruments on a field programmable gate array (FPGA)’. In: *Proceedings of the 15th International Conference on Digital Audio Effects (DAFx-12)*. York, UK, 2012 (cit. on p. 8).

- [90] M. Popoff. ‘Compilation of Real-Time Audio DSP on FPGA’. NNT number: 2024ISAL0128. PhD. Lyon: INSA Lyon, Dec. 2024 (cit. on p. 21).
- [91] M. Popoff, R. Michon, T. Risset, Y. Orlarey and S. Letz. ‘Towards an FPGA-Based Compilation Flow for Ultra-Low Latency Audio Signal Processing’. In: *SMC-22 - Sound and Music Computing*. Saint-Étienne, France, June 2022. URL: <https://inria.hal.science/hal-03805199> (cit. on pp. 20, 21).
- [92] T. Reussner, C. Sladeczek, M. Rath, S. Brix, K. Preidl and H. Scheck. ‘Audio Network-Based Massive Multichannel Loudspeaker System for Flexible Use in Spatial Audio Research’. In: *Journal of the Audio Engineering Society* 61.4 (2013), pp. 235–245 (cit. on p. 21).
- [93] J. Revels, M. Lubin and T. Papamarkou. ‘Forward-Mode Automatic Differentiation in Julia’. In: *arXiv:1607.07892 [cs.MS]* (2016). URL: <https://arxiv.org/abs/1607.07892> (cit. on p. 28).
- [94] T. A. Rushton. ‘Differentiable DSP in Faust’. In: *IFC 2024 - 4th International Faust Conference*. Turin, Italy, Nov. 2024 (cit. on p. 28).
- [95] E. Salze, E. Jondeau, A. Pereira, S. L. Prigent and C. Bailly. ‘A New MEMS Microphone Array for the Wavenumber Analysis of Wall-Pressure Fluctuations: Application to the Modal Investigation of a Ducted Low-Mach Number Stage’. In: *Proceedings of the 25th AIAA/CEAS Aeroacoustics Conference*. Delft, Netherlands, 2019 (cit. on p. 9).
- [96] R. Schreiber, S. Aditya, S. A. Mahlke, V. Kathail, B. R. Rau, D. C. Cronquist and M. Sivaraman. ‘PICO-NPA: High-Level Synthesis of Nonprogrammable Hardware Accelerators’. In: *VLSI Signal Processing* 31.2 (2002), pp. 127–142 (cit. on p. 11).
- [97] S. Serafin, C. Erkut, J. Kojis, N. C. Nilsson and R. Nordahl. ‘Virtual reality musical instruments: State of the art, design principles, and future directions’. In: *Computer Music Journal* 40.3 (2016), pp. 22–40 (cit. on p. 15).
- [98] D. Shi, W.-S. Gan, J. He and B. Lam. ‘Practical Implementation of Multichannel Filtered-x Least Mean Square Algorithm Based on the Multiple-Parallel-Branch With Folding Architecture for Large-Scale Active Noise Control’. In: *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2019) (cit. on p. 15).
- [99] F. Thabet, P. Coussy, D. Heller and E. Martin. ‘Exploration and Rapid Prototyping of DSP Applications using SystemC Behavioral Simulation and High-level Synthesis’. In: *Signal Processing Systems* 56.2-3 (2009), pp. 167–186 (cit. on p. 11).
- [100] D. Theodoropoulos, C. B. Ciobanu and G. Kuzmanov. ‘Wave field synthesis for 3D audio: Architectural perspectives’. In: May 2009, pp. 127–136. doi: [10.1145/1531743.1531764](https://doi.org/10.1145/1531743.1531764) (cit. on p. 21).
- [101] R. Troian, D. Dagna, C. Bailly and M.-A. Galland. ‘Broadband Liner Impedance Education for Multimodal Acoustic Propagation in the Presence of a Mean Flow’. In: *Journal of Sound and Vibration* 392 (2017), pp. 200–216 (cit. on p. 16).
- [102] Y. Uguen, F. de Dinechin and S. Derrien. ‘Bridging High-Level Synthesis and Application-Specific Arithmetic: The Case Study of Floating-Point Summations’. In: *27th International Conference on Field-Programmable Logic and Applications (FPL)*. IEEE. Sept. 2017 (cit. on p. 12).
- [103] Y. Uguen, L. Forget and F. de Dinechin. ‘Evaluating the hardware cost of the posit number system’. In: *29th International Conference on Field-Programmable Logic and Applications (FPL)*. Barcelona, Spain, Sept. 2019. URL: <https://hal.inria.fr/hal-02130912> (cit. on p. 11).
- [104] T. C. Vannoy. ‘Enabling Rapid Prototyping of Audio Signal Processing Systems Using System-on-Chip Field Programmable Gate Arrays’. PhD thesis. Montana State University-Bozeman, Norm Asbjornson College of Engineering, 2020 (cit. on p. 21).
- [105] B. Verplank, M. Gurevich and M. V. Mathews. ‘THE PLANK: Designing a simple haptic controller.’ In: *Proceedings of the New Interfaces for Musical Expression Conference*. 2002, pp. 33–36 (cit. on p. 15).
- [106] M. Verstraelen, J. Kuper and G. J. M. Smit. ‘Declaratively Programmable Ultra-Low Latency Audio Effects Processing on FPGA’. In: *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx-14)*. Fraunhofer Institut, Sept. 2014, pp. 263–270. (Visited on 12/06/2024) (cit. on p. 21).

- [107] M. Verstraelen, J. Kuper and G. J. Smit. ‘Declaratively Programmable Ultra Low-Latency Audio Effects Processing on FPGA’. In: *Proceedings of the 17th International Conference on Digital Audio Effects (DAFx-14)*. Erlangen, Germany, 2014 (cit. on p. 8).
- [108] A. Volkova, R. Garcia, F. de Dinechin and M. Kumm. ‘Hardware-optimal digital FIR filters: one ILP to rule them all and in faithfulness bind them’. In: *Proceedings of the Asilomar conference*. Asilomar, United States, Oct. 2023. URL: <https://inria.hal.science/hal-04398268> (cit. on p. 13).
- [109] A. Volkova, M. Istoan, F. de Dinechin and T. Hilaire. ‘Towards Hardware IIR Filters Computing Just Right: Direct Form I Case Study’. In: *IEEE Transactions on Computers* 68.4 (Apr. 2019) (cit. on p. 12).
- [110] J. Zhang, T. D. Abhayapala, W. Zhang, P. N. Samarasinghe and S. Jiang. ‘Active Noise Control Over Space: A Wave Domain Approach’. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 26.4 (Apr. 2018), pp. 774–786 (cit. on p. 9).
- [111] T. G. Zieliński, M.-A. Galland and M. N. Ichchou. ‘Fully Coupled Finite-Element Modeling of Active Sandwich Panels with Poroelastic Core’. In: *Journal of vibration and acoustics* 134.2 (2012) (cit. on p. 16).