# FAST: Fast Audio Signal-processing Technologies on FPGA

**Partners**
Grame – Centre National de Création Musicale (CNCM)
CITI – Institut National de Sciences Appliquées de Lyon (INSA)
LMFA – École Centrale de Lyon

**Coordinator**
Romain Michon, GRAME-CNCM, Lyon
michon@grame.fr https://ccrma.stanford.edu/~rmichon

PRC – Full Proposal
Domaines transversaux
CE38 (Axe 8.6.) Révolution numérique : rapports au savoir et à la culture

**Keywords:** Approche numérique à la création artistique, Traitement audio-numérique ultra-basse latence, Programmation de haut niveau, Acoustique virtuelle

**Persons Involved In the Project**

| Partner | Individual | | Role, Responsibility, and Effort (in P.m.) | |
|---|---|---|---|---|
| Grame | Romain Michon | Researcher | Project coordinator and leader of WP0, WP3, and WP4 | 10 |
| Grame | Yann Orlarey | Researcher | Contributor to WP1 and WP2 | 5 |
| Grame | Stéphane Letz | Researcher | Leader of WP1 | 8 |
| Grame | (*To be hired*) | Engineer | Contributor to WP3 | 12 |
| Grame | (*To be hired*) | Intern | Contributor to WP4 | 6 |
| CITI | Tanguy Risset | Professor | Leader of WP2 | 13 |
| CITI | Florent de Dinechin | Professor | Contributor to WP0, WP1, and WP2 | 11 |
| CITI | (*To be hired*) | PhD Student | Contributor to WP0, WP1, WP2, WP3, WP4, WP5, and WP6 | 36 |
| LMFA | Pierre Lecomte | Associate Professor | Leader of WP5 and WP6 | 11 |
| LMFA | Marie-Annick Galland | Professor | Contributor to WP0, WP5 and WP6 | 6 |
| LMFA | Sébastien Ollivier | Associate Professor | Contributor to WP0 and WP4 | 4 |
| LMFA | (*To be hired*) | Postdoc | Contributor to WP6 | 18 |
| LMFA | (*To be hired*) | Intern | Contributor to WP5 | 6 |

## Evolution Since the Pre-Proposal

The following modifications were made since the pre-proposal: (1) We reinforced the links between the project and SHS by highlighting in the context of the arts the applications of the tools that will be developed. In particular, a studio/rehearsal space with a modular acoustic featuring events open to the public will be implemented at LMFA. (2) We tried to make the essence of the proposal higher-level and less technical, taking the pre-proposal reviewer's comments into account. (3) Workpackages were reorganized to carry out all the hardware-related tasks before the work around artificial reverberation and active noise control. We also added some overlap between the two workpackages related to artificial reverberation and active noise control. (4) The overall budget of the project has been increased by 5% to take into account additional equipment costs related to new tasks in some workpackages which consequently resulted in a 5% increase of the requested funding. (5) The content of each workpackage is now thoroughly described and we provided additional technical details about the methodology that will be used for each task.

## 1 Context, Positioning, and Objectives

Embedded systems for audio and multimedia are increasingly used in the arts and culture (e.g., interactive systems, musical instruments, virtual and augmented reality, artistic creation tools, musical composition and performance, etc.). However, programming them can be out of reach to artists, creators, or non-specialized engineers. In parallel with the emergence of the "maker culture,"[1] progress have been made to make these types of systems more accessible, bringing more flexibility in digital approaches to artistic creation. For instance, the Arduino platform[2] greatly simplified the programming process of microcrontrollers. Similarly, Domain Specific programming Langages (DSL) such as FAUST[3] [48] facilitated the implementation of real-time audio Digital Signal Processing (DSP) algorithms.

[1] https://makerfaire.com/maker-movement (All URLs were verified on May 1, 2020).

[2] https://www.arduino.cc

[3] FAUST is a DSL for real-time audio signal processing primarily developed at GRAME-CNCM and by a worldwide community. FAUST is based on a compiler "translating" DSP specifications written in FAUST into a wide range of lower-level languages (e.g., C, C++, Rust, Java, WASM, LLVM bitcode, etc.). Thanks to its "architecture" system, 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.).

However, many limitations remain, especially for real-time applications where latency plays a crucial role (e.g., efficient active control of sound where audio processing should be faster than the propagation of sound [19], digital musical instruments playability [28], digital audio effects, etc.). While latency can be potentially reduced on "standard" computing platforms such as personal computers based on a CPU (Central Processing Unit), going under the "one millisecond threshold" is usually impossible because of buffering.

FPGAs (Field Programmable Gate Arrays) can help solve this problem as well as most of the limitations of traditional computing platforms used for musical and artistic applications. These chips are known for their high computational capabilities [14, 50] and very low-latency performances [69]. 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 fields capture using a very large number of digital microphones [56], active sound control over a large spatial region [72], etc.).

But FPGAs remain extremely complex to program, even with state-of-the-art high-level tools,[4] making them largely inaccessible to musicians, digital artists and makers communities. There are currently only a few examples of professional FPGA-based real-time audio DSP systems (i.e., Antelope Audio,[5] Korora Audio,[6] etc.) and in these applications, FPGAs are dedicated to a specific task, limiting creativity and flexibility.

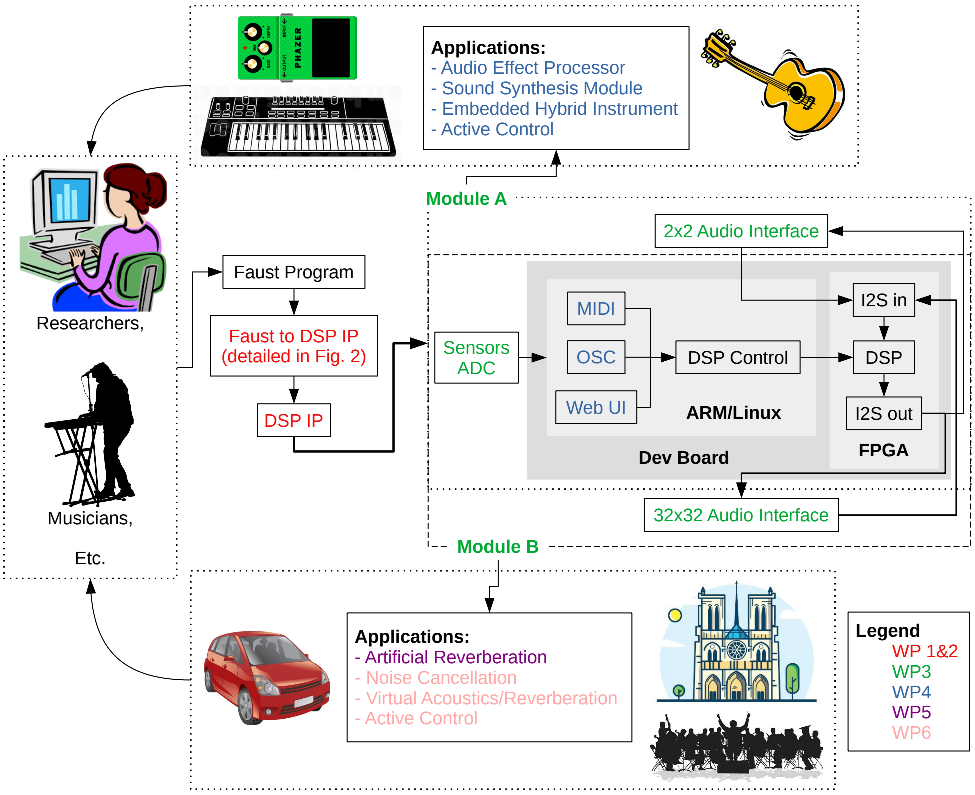

Figure 1: FAST overview.

## 1.1 Objectives and Research Hypothesis

The FAST project was thought to overcome the aforementioned difficulties (i.e., low latency, computing capacity, multi-channel, and ease of programming). **The goal of FAST is to design an FPGA-based platform for multichannel ultra-low-latency audio Digital Signal Processing (DSP) programmable at a high-level with FAUST** and usable for various applications ranging from sound synthesis and processing to active sound control and artificial sound field/room acoustics. In addition to that, we then plan to use this tool to:

- design a programmable sound processing/synthesizer module for musicians, artists, and makers;
- actively modify the acoustical properties of acoustic musical instruments and create "smart/hybrid instruments;"
- create a studio/rehearsing space with an adaptive acoustic.

FAST will gather the strengths of GRAME-CNCM[7] (FAUST, compilation, musical audio applications, dissemination/mediation, professional music production), CITI[8] at INSA Lyon (FPGA, DSP, fixed-point processing with FloPoCo[9]), and LMFA[10] at École Centrale de Lyon (acoustics,

[4] 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: https://www.ni.com/fr-fr/shop/labview.html, Vivado HLS: https://www.xilinx.com/HLS, etc.), but none of them is specifically dedicated nor adapted to real-time audio DSP.

[5] https://en.antelopeaudio.com

[6] https://www.kororaaudio.com

[7] http://grame.fr

[8] http://www.citi-lab.fr

[9] http://flopoco.gforge.inria.fr

[10] http://lmfa.ec-lyon.fr

DSP, artificial reverberation, active control).

The main objective of FAST is to **develop a new FAUST architecture backend for FPGA-based platforms**. 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 program 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. Eventually, we aim at generating VHDL code directly from the FAUST compiler.

This type of system has a wide range of applications in multiple domains. As mentioned earlier, music technology is in high demand for low latency because it helps increasing the playability of musical instruments on stage. Hence, the platform developed as part of FAST will be used to design programmable digital musical instruments and effect processors. In that context, the computational power of the FPGA will be exploited to run complex algorithms (e.g., physics-based models of musical instruments [8], modal reverbs [2], etc.) that are too costly to run on a traditional platform (i.e., laptop, etc.). GRAME-CNCM has decades of experience in that domain.

We plan to **implement two hardware/software audio DSP modules based on the platform** implemented in FAST. One will target the makers and music technology communities and will provide a stereo input and output as well as GPIOs for sensors. The other will target active acoustic control and spatial audio applications by providing 32 audio inputs and outputs.

One of the main field of application for the platform that we plan to develop as part of FAST is active control of acoustical spaces (e.g., noise cancellation in rooms, car passenger compartment, etc.) and virtual room acoustics (e.g., to apply the acoustical properties of a space to another, etc.). Sound field rendering systems are in high demand for low audio latency (i.e., to beat acoustical waves traveling at the speed of sound in the air) and high computational power (i.e., to implement Finite Difference Time Domain: FDTD schemes [8]). While FPGAs have been used in the past for this type of applications [62], there is a lack for a high-level tool to program and implement these types of algorithms. Similarly, FPGAs should allow to run in real-time room acoustics simulation algorithms such as modal reverbs [1] in the time domain, which is not currently possible on regular computers. LMFA École Centrale has a lot of experience in these domains and a wide range of equipment at its disposal (e.g., sound field array/listening room, etc.) that will be used/adapted to this purpose. We plan to **prototype real-world examples of room acoustics simulation/modifications** that could then be used in a concert setting. Such experiments have been attempted in the past at the Center for Computer Research in Music and Acoustics (CCRMA)[11] at Stanford University that is indirectly related to FAST thanks to Romain Michon's affiliation. In particular, the acoustics of the Hagia Sophia cathedral in Istanbul has been recreated in Stanford's Bing Concert Hall for a series of concerts centered around the idea of "archeoacoustics" [2]. The platform that we plan to implement as part of FAST will be used to take this type of system to a completely different level. Hence, we plan to organize events centered around these concepts with the help of GRAME-CNCM music production department. In particular, **we will design a system to recreate the acoustics of various landmarks from the Lyon area in LMFA's listening room to turn it into an immersive studios** where small events open to the public will be organized.

Finally, applications around active control are not limited to acoustical spaces. Musical instrument acoustics/digital lutherie are increasingly using these types of techniques [72] not to mention industrial applications outside of the field of audio (e.g., aircraft jet engine vibration control, etc.) which would be easily reachable thanks to LMFA's expertise in these domains [56].

## 1.2 Positioning In Relation to the State of the Art

### 1.2.1 Domain Specific Languages for Real-Time Audio DSP

While there exists many high-level standard programming languages providing libraries/environments for asynchronous signal processing (e.g., MATLAB, Python, etc.), real-time DSP is usually programmed through C/C++ or even lower level paradigms. Nonetheless, several languages targeting real-time signal processing applications such as Kronos [47], Arrp [30], Stride [65], CSOUND [29], and more recently SOUL[12] have been available for many years. FAUST [48] is part of that category as well and can be regarded as the main alternative to C/C++ for real-time audio applications. By allowing us to target an unparalleled number of systems and platforms, it is now considered as a standard both in the industry and in academia. We plan to take advantage of FAUST's accessibility, versatility, and ease of use as part of FAST to facilitate the programming of FPGAs.

One important step is to enable fixed-point data computation in FAUST-generated code. Indeed, sound samples are essentially fixed-point data, and using floating-point for sound processing is inherently inefficient. It does make sense on CPUs which include expensive floating-point units anyway, but the main reason why floating-point is used for sound processing is that it is convenient, relieving the designer from having to think about the range and accuracy of data in a program. In FAST, we intend to provide an accessible programming paradigm while offering better efficiency, thanks to a compiler that automatically determines fixed-point formats.

### 1.2.2 Ultra-Low-Latency Signal Processing

Audio latency is a central concern when it comes to digital audio [28]. There are typically two main sources of latency on digital audio systems: buffering at the software/computation level and Digital to Audio Converters (DAC) themselves (mostly because of signal reconstruction filters). While ultra-low-latency ($50\mu s$) audio ADCs/DACs such as the Analog Devices ADAU177[13] can be found on the market, achieving ultra-low-latency can be relatively hard on the software and computational side.

Various systems both in the industry and in academia have been targeting low audio latency through the use of different hardware solutions. The most approachable ones are embedded Linux systems using dedicated hardware and running audio DSP outside of the operating system. The BELA [33] and the Elk[14] are the only two members of this family. They target the Beagle Bone,[15] and the Raspberry Pi[16] respectively. This type of system can achieve relatively low-latency with buffer sizes as low as 8 samples.

[11] https://ccmra.stanford.edu
[12] https://soul.dev/
[13] https://www.analog.com/en/products/adau1777.html
[14] https://elk.audio/
[15] https://beagleboard.org/
[16] https://www.raspberrypi.org/

Microcontrollers have been used more and more in recent years for sound synthesis and processing because of their increasing power. The Teensy[17] [38] and the ESP32[18] [39] are good examples of such systems. Thanks to their lightness (e.g., no OS), their latency performances are similar to that of dedicated/specialized embedded Linux systems (i.e., 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. They usually target engineers making them less approachable than the other types of systems mentioned in this section. Additionally, many of them don't provide native support for floating-points, further increasing the complexity of their programming. The Analog Devices SHARC Processor[19] is a leader on the market and it can be used as a prototyping system through the SHARC Audio Module.[20] It also provides an official FAUST support.[21] Cheaper/more lightweight DSP-based platforms such as the ESP32 LyraTD[22] are also starting to appear.

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

### 1.2.3 FPGAs Programming for Real-Time Audio Processing

FPGAs have been used in the field of audio processing for their high computational capabilities [14, 50, 44, 45] and very low-latency performances [69]. Their large number of GPIOs (General Purpose Inputs and Outputs) has also been exploited to implement modern real-time multi-channel processing algorithms (e.g., sound fields capture using a very large number of digital microphones [56, 64], various signal processors for audio [53, 74]). The resulting FPGA design are rarely open source. Recently, Vaca et al. presented an open audio processing platform based on the Zybo board [67] for dedicated purpose such as collecting analog frequency of a musical instrument. However, this project was not targeting ultra-low-latency applications.

Programming FPGA is usually done with a hardware description language (VHDL or Verilog). Developing a VHDL IP[23] is extremely time consuming. Hence, FPGA programmers have two possibilities: re-using existing IPs and assemble them to compose a circuit solving their problem (as proposed by LABVIEW[24]), or use High-Level Synthesis to *compile* a VHDL specification from a higher-level description.

High Level Synthesis [46] 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 no automatic tool is able to capture all the constraints and come up with the *optimal* solution. Many HLS tools have been proposed (i.e., Pico [58], CatapultC [9], Gaut [63], 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 (C, C++, or Python) which is *instrumented* using pragmas to guide the HLS process.

Using HLS today still requires very specific skills [26] to write a source description that is correctly processed by HLS tools. For instance, the use of optimized fixed-point arithmetic is mandatory because floating-point arithmetic on FPGA is much more expensive. This problem requires more than simple engineering to be tackled [70, 49, 59]. Using CITI expertise in that domain [17] will be a major technological breakthough that will enable efficient compilation of FAUST programs on FPGA. We believe that HLS technology has reached a certain maturity and can now be foreseen as a valuable tool for audio designers, DSP engineers and researchers, etc.

In FAST, we plan to take this kind of work to the next level by making FPGA platforms more accessible for ultra-low-latency multi-channel real-time signal processing applications.

### 1.2.4 Active Control of Musical Instruments

Since the 90's, different active control techniques have been successfully applied to the control of hybrid musical instruments in research laboratories, first for percussion and string instruments, and more recently for wind instruments [4, 10, 36, 37]. The use on stage of augmented instruments with integrated sensors and actuators and real-time control of the resonances using computers have shown new musical perspectives. Brass mutes based on active sound control such as Yamaha's Silent Brass[25] and actively controlled acoustic guitars (i.e., HyVibe[26]) are now available on the market. However, signal processing in these instruments is much simpler than one would expect from laboratory studies. Actually, the lack of efficient and fast computing capabilities has limited the complexity of control algorithms in smart musical instruments with embedded controllers. The development of hybrid instrument and the wider appropriation of active control techniques by musicians, musical instruments makers, and sound artists will now depend on the availability of easily programmable, affordable, and very fast electronic controllers.

### 1.2.5 Artificial Reverberation

Artificial reverberation can be implemented using a wide range of techniques such as feedback delay networks, waveguide meshes, finite difference scheme, simple combination of IIR filters, and Impulse Responses (IR) convolution [68]. The latter has been made more accessible by the increasing power of computers in recent years (it is relatively computationally intensive compared to some of the other methods). It allows us to apply the reverberation of an existing space onto a dry signal by convolving it with the IR of the space [54]. When recorded with an ambisonic microphone, the IR can be used to recreate the reverberation of a space in 3D using a speaker array. This method has been used in the framework of various projects around "archeoacoustics" which partly consists in recreating the acoustical environments of lost or forbidden spaces potentially towards real-time performances. The recreation in Bing Concert Hall at Stanford University of the acoustic of the Hagia Sophia cathedral in Istanbul [2] is a good example of that. As part of FAST, we plan to establish a catalog of 3D IRs of landmarks from the Lyon area to recreate their acoustics in LMFA's listening room.

Convolution reverbs have been taken to another step recently by introducing the concept of "modal reverb" [1] where convolution is carried out in the time domain instead of the frequency domain by using a bank of resonant bandpass filters taking advantage of the principle of modal

[17] https://www.pjrc.com/teensy/

[18] https://www.espressif.com/en/products/hardware/esp32/overview

[19] https://www.analog.com/en/products/processors-dsp/dsp/sharc.html

[20] https://www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/sharc-audio-module.html

[21] https://wiki.analog.com/resources/tools-software/sharc-audio-module

[22] https://www.espressif.com/en/products/hardware

[23] IP stands for Intellectual Property, it is the common denomination for *hardware library*, i.e., a circuit design that can be re-used as a software library

[24] https://www.ni.com/fr-fr/shop/labview.html

[25] https://fr.yamaha.com/fr/products/musical_instruments/winds/silent_brass/index.html

[26] https://www.hyvibe.audio/smart-guitar/

decomposition [3]. This approach is much more controllable since it gives access to the parameters of all the filters used in the bank. This comes at the cost of computation since hundreds of filters are required to achieve perceptually acceptable results. On the other hand, this type of algorithm is highly parallelizable, a property we plan to take advantage of since FPGAs offer a close-to-infinite level of parallelization. While similar experiments have been prototyped of GPUs [61], our system will take this to the next level by providing ultra-low latency and active control of the acoustic of the room where the sound will be rendered.

### 1.2.6 Active Control of Rooms Acoustics

Active control of room acoustics is a challenging topic with the first commercial applications dating back to the 90s [21]. The objective of this thematic is to vary at will the subjective and quantifiable acoustic parameters of a room (reverberation time, early reflections, loudness, etc.) in order to adapt it to the artist performance and the venue [51]. To achieve this, several loudspeakers microphones and DSP modules are used to create artificial reflections in a non-generative way [27]. Several commercial systems are available on the market such as the CSTB's CARMEN [57] or YAMAHA AFC system [27] [43] to name a few. Other examples can be found in [51]. In such systems, the feedback loops should be controlled, as well as the system stability, and the real-time DSP.

Although many quantifiable room acoustics criteria are optimized in these applications, the primary goal is to create "pleasant" acoustics for artists and listeners in a concert hall. It is still very difficult to physically control the entire sound field over a continuous spatial area, which could bring a significant improvement. Recent advances in active control and approaches to sound field decomposition [31] allow to partially overcome this limitation, by proposing "spatial active noise contol" algorithms [73, 23]. As one can imagine, such approaches require a very large number of control channels with very low latency. It is precisely on this point that the FAST project will provide the technological solution to implement such algorithms in a real situation.

This section first describes the methodology (i.e., work packages) that will be used to implement FAST. A detailed Gannt Chart can be found in Fig. 4 on p. 20. The second part of this section is about risk management.

### 1.2.7 WP0 – Management, Coordination, and Dissemination

| WP0 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| Grame, Michon | M0 | M41 | 4 P.m. | 4 P.m. | 4 P.m. | 12 P.m. |

T0.0 – Project Management and Scientific Coordination *T0.0 will start at M0 and end at M41. Total effort is 6 P.m: Grame 2, CITI 2, and LMFA 2.*

This task consists in: (1) Project management and planning (progress analysis and task coordination); (2) Consortium agreement, legal and intellectual property management; (3) Budget management; (4) Risk management (identification, assessment, and risk control) and potential project strategy updates; (5) Identification and resolution of potential conflicts; (6) Management of shared resources (e.g., web sites, wiki, code repositories, shared equipment); (7) Organization of internal meetings; (8) Monitoring of tasks deliverable, reports, cost statements, publications.

T0.1 – Dissemination *T0.1 will start at M0 and end at M41. Total effort is 6 P.m: Grame 2, CITI 2, and LMFA 2.*

- A website will present the project, the partners, the publications, and the tools developed as part of FAST.
- There is a potential for multiple "popular science" publications on this project (i.e., actively modifying the acoustics of rehearsal room, developing ultra-low latency audio effects and synthesizers, reproduction of lost acoustic spaces: archeoacoustics, etc.).
- Workshops will be organized on the technology/tools developed in the framework of FAST in French institutions interested in these topics and at Stanford University (where R. Michon is one of the instructors of the *CCRMA Summer Workshops Series*[28]).
- One of Grame's Music Production Department's missions is to host artist residencies year-round. In that context, artist residencies will be organized at Grame to exploit/abuse the tools, systems, and technologies developed as part of FAST.
- Another mission of Grame's Music Production Department is to organize concerts. Hence, we'll take advantage of Grame's experience and resources in this field to organize concerts leveraging the technology developed as part of FAST.
- The technology developed as part of FAST will be demonstrated to the public through events at LMFA's listening room/studio where live and fixed-media performances will be carried out (see WP4).
- The results of the project will be published in scientific journals, proceedings of international conferences and seminars, etc.

| Deliverable of WP0 | Due Date | Leader |
|---|---|---|
| **D0.0** – Consortium Agreement | M7 | Grame |
| **D0.1** – 1st Interim Management Report | M17 | Grame |
| **D0.2** – 2nd Interim Management Report | M29 | Grame |
| **D0.3** – Final Management Report | M41 | Grame |

### 1.2.8 WP1 – Fixed-Point Processing in FAUST

| WP1 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| (Grame, Letz) | M0 | M35 | 7 P.m. | 32 P.m. | 0 P.m. | 39 P.m. |

Up to now, the FAUST compiler has only been producing floating-point C++ code since it mostly targets CPUs. This work package focuses on integrating in the FAUST compiler the required technologies to use fixed-point arithmetic in the generated code. The key idea is to use the arithmetic expertise of CITI-Lab [70] to infer the bitwidth for each variable in a FAUST program in order to ensure high sound quality in the most

[27] https://fr.yamaha.com/fr/products/proaudio/afc/afc/index.html
[28] https://ccrma.stanford.edu/workshops

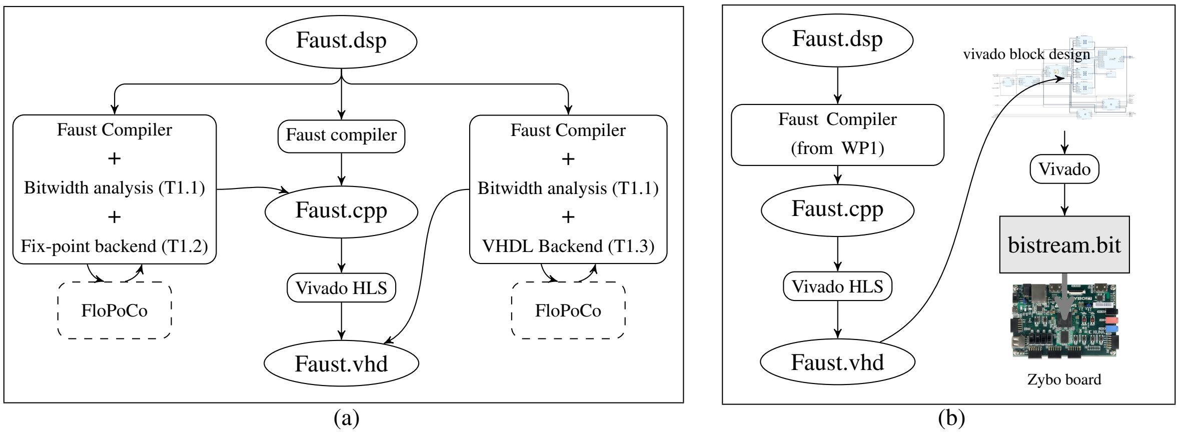

Figure 2: (a) Different possible compilation flows for generating VHDL from a FAUST program studied in WP1. Round boxes are files (i.e., format), squared boxes are tools. Dashed boxes are tools outside the compilation flow used to tune the architecture. (b) The complete faust2fpga flow engineered in WP2

efficient way. This will be carried out in task T1.1. Once that is done, we will be able to produce fixed-point DSP C++ code, extending the current C++ FAUST backend to target HLS tools (leftmost path in Fig. 2-(a)). Alternatively, we will directly generate VHDL using the existing FloPoCo VHDL backend (rightmost path in Fig. 2-(a)).

The middle path in Fig. 2-(a) is the easiest and currently only supported one: the FAUST compiler is "tuned" to output C++ code that fits into `vivado_hls` tools. Audio samples are coded on floats in that case. This flow will be studied in WP2 because many important optimizations are independent of the type chosen for the computation (i.e., memory access organization).

The left flow of Fig. 2-(a) adds fixed-point optimization. Here, the types used for computation are fixed-point of arbitrary width. First, an analysis of the FAUST data-flow graph should indicate what the required bit-width for each signal is, then the C++ code is generated accordingly and sent to `vivado_hls` using dedicated arithmetic operators. The type of analysis performed here is routinely performed in the `FloPoCo` tool [17], and for some well identified sub-blocks such as IIR filters [70] or trigonometric functions [16, 15], the bitwidth optimization code within FloPoCo can be used directly. This will be carried out in task T1.2.

Finally (right flow), for some FAUST program that are compute-bound, it might be interesting to think of a direct compiler from FAUST to VHDL. Indeed, the FAUST flow graph representation is close to a structural representation of the computations. The main difficulty here lies in the inference of memory accesses. For mundane tasks of VHDL generations, the FloPoCo VHDL backend [17] can be used. It also offers additional low-level arithmetic optimizations currently out of reach of HLS tools [12]. This will be studied in task T1.3.

**T1.0 – Survey and Bibliography of float2fixed Methodology** *T1.0 will start at M0 and end at M6. Total effort is 5 P.m.: Grame 1, CITI 4, and LMFA 0.*

This task will be responsible for choosing the strategies that will be explored for inferring fixed-point signal bitwidth in FAUST programs. This will imply a complete survey of existing techniques [13, 49, 59, 70] and industrial tools (e.g., Matlab, Xilinx [71], etc.). However we do not intend to develop a universal float-to-fix methodology as most of the existing ones work. Instead, we wish to exploit the domain knowledge available in the consortium. Some are trivial (e.g., inputs from ADCs and outputs to DACs are 16- to 24-bit fixed-point formats), some are more advanced (e.g., higher frequencies do not require the same resolution as lower ones, etc.). The role of this task will be to express constraints on the range and resolution of stream samples in a FAUST program, then define algorithms to propagate and solve these constraints until a fixed-point format can be associated to each variable. It will follow the *"computing just right"* paradigm successfully exploited at CITI in other domains.

**T1.1 – Integration of Fixed-Point in Faust Compiler** *T1.1 will start at M3 and end at M18. Total effort is 13 P.m.: Grame 2, CITI 11, and LMFA 0.*

The methodology and algorithms discussed in the previous task should then be implemented in the FAUST compiler. This means selecting a fixed-point library and implementing possibly complex algorithms [70] for choosing the bitwidth of each stream manipulated by FAUST. This task should finally deliver the first compilation of FAUST code to fixed-point C++ programs (Deliverable D1.0). We also expect this work to be usefull for other hardware targets, such as CPUs without floating-point unit.

**T1.2 – Generation of Fixed-Point C++ Code for HLS** *T1.2 will start at M6 and end at M35. Total effort is 6 P.m.: Grame 1, CITI 5, and LMFA 0.*

Interaction with the HLS tool (`vivado_hls`) has to be carefully studied: optimizations, in particular pipelining, provided by `vivado_hls` should not be blocked by the use of dedicated fixed-point operators. This task should optimize the faust2fpga design Flow delivered in T2.0: we except a complexity improvement of at least one order of magnitude for the generated designs.

**T1.3 – Generation of VHDL Directly from FAUST** *T1.3 will start at M12 and end at M35. Total effort is 15 P.m.: Grame 3, CITI 12, and LMFA 0.*

Another positive outcome of this task will be to make the FAST results independent from Xilinx tools. However it is probably too ambitious to target *all* possible FAUST programs. We except to issue (D1.1) the first faust2VDHL tool.

This task will attempt to identify FAUST programs where direct VHDL compilation would be more efficient. While HLS tools are now very capable, there is a wide range of optimization opportunities that are currently out of their reach, for instance the possible fusion of a fixed-point sum of products in a single sum of weighted bits [20, 12]: this happens a lot in digital signal processing [70].

| **Deliverable of WP1** | **Due Date** | **Leader** |
|---|---|---|
| **D1.0** – FAUST Compilation Towards Fixed-Point C++ Programs | M18 | Grame |
| This software deliverable will enable fixed-point designs to be compiled. Optimization of this tool-chain will be proceeded throughout the project. | | |
| **D1.1** – First faust2VDHL Direct Compilation Flow | M35 | Grame |
| This software module will allow to generate VHDL directly from FAUST without depending on Xilinx `vivado_hls`. It will most probably apply to a subset of FAUST programs to be defined. | | |

### 1.2.9 WP2 – FPGA Backends for FAUST

| **WP2 Leader** | **Start** | **End** | **Grame** | **CITI** | **LMFA** | **Total** |
|---|---|---|---|---|---|---|
| CITI, Risset | M0 | M35 | 5 P.m. | 20 P.m. | 0 P.m. | 25 P.m. |

Obtaining VHDL from a FAUST program is not sufficient to *compile* FAUST programs to FPGAs. Indeed, the real design space exploration will take place during the *high-level synthesis* of the C++ code generated by FAUST compiler (using an existing HLS tool) and during the *block design* tuning. As mentioned earlier, we have chosen to use Xilinx hardware board and software tools (`vivado_hls` and `vivado`). However, different types of boards can be targeted. In particular, a low cost board (e.g., Zybo,[29] the successor of Zedboard) and free webpack Xilinx software suite will be used for our compilation flow development. Conversely, a high performance FPGA (e.g., with Zynq UltrasScale[30]) and Vivado HL Design Editions will be used for the applications of WP3 and WP4, if needed.

Fig. 2-(b) illustrates the generation of a *bitstream*, i.e., an FPGA binary file. In the so-called *block design*, the FAUST IP generated from the VHDL code – `Faust.vhd` – should be interfaced with:

1. The on-board audio codecs using I2C and I2S IPs.
2. The memory of the board (for storing samples when echo/delay is used)
3. The ARM processor that will interact with the FAUST IP.

Important choices have to be made here which are usually referred to as *design space exploration*. These choices might depend on the target application and on the target board. They are related to the implementation of FAUST controllers, the memory architecture, and the schedule of the access to the different memories. These choices are the core of the different tasks of WP2.

T2.0 – First faust2fpga Design Flow With Fixed-Point *T2.0 will start at M3 and end at M12. Total effort is 5 P.m.: Grame 1, CITI 4, and LMFA 0.*

This task is responsible to produce the first automatic FPGA programming flow from FAUST. For that, many engineering issues have to be figured out: integration of fixed-point handling proposed in Task T1.1, interfacing with audio codec using I2S serial protocol and use of the processing subsystem (i.e., ARM processor) for initialization and control of the DSP treatment implemented. Following our recent proof of concept [55], we are confident in a rapid delivery of a functional design flow.

T2.1 – Design Space Exploration for Memory Organization *T2.1 will start at M9 and end at M24. Total effort is 10 P.m.: Grame 2, CITI 8, and LMFA 0.*

The compilation flow produced in the previous task (T2.0) will have to be optimized to produce more efficient results, either in terms of latency or in terms of computing power. There is a non-trivial design space exploration to perform in order to decide which audio samples should be stored on FPGA Block RAM and which should be stored in the external memory. External memory will imply slower treatment, but the treatment can be pipelined over successive samples (increasing latency, of course). This trade-off will be studied in order to produce a precise memory organization methodology depending on the constraint of the implemented DSP.

T2.2 – Design Flow Adaptation for High Performance FPGA *T2.2 will start at M18 and end at M35. Total effort is 5 P.m.: Grame 1, CITI 4, and LMFA 0.*

This task will provide the necessary adaptations of the compilation flow of Fig. 2-(b) to target the high-performance Zynq Ultrascale FPGA *Genesys ZU*. This target will be used for compute bound algorithms studied in WP5 (T5.1) and WP6 (T6.2). High computing power will need to optimize parallelization techniques used in the compilation flow (either by `vivado_hls` or by direct VHDL generation from FAUST).

T2.3 – Performance Study of the faust2fpga Compilation Flow *T2.3 will start at M24 and end at M35. Total effort is 5 P.m.: Grame 1, CITI 4, and LMFA 0.*

An important outcome of the FAST project is this new open-source compilation flow from FAUST to FPGA that will be useful in many contexts: for musicians, acoustic engineers or vibration's mechanics engineers. In order to convince these people to use it, we have to prototype a large number of audio treatments (e.g., filters, reverb effects, etc.) and study the obtained performances – in terms of latency and computing power again – depending of the configuration chosen for the flow (as explained in T2.1).

[29] https://www.xilinx.com/products/boards-and-kits/1-4azfte.html
[30] https://store.digilentinc.com/genesys-zu-zynq-ultrascale-mpsoc-development-board/

| Deliverable of WP2 | Due Date | Leader |
|---|---|---|
| **D2.0** – First faust2fpga Design Flow With Fixed-Point | M12 | Citi |
| This software deliverable will enable fixed-point designs to be compiled on FPGA. It will constitute the basis of the compilation flow release occuring throughout the project. | | |
| **D2.1** – faust2fpga Design Flow adapted for high performance FPGAFirst faust2VDHL Direct Compilation Flow | M35 | Citi |
| The software deliverable will consist in adapting the faust2fpga Design Flow to Ultrascale FPGA Genesys ZU for use in WP5 and WP6 | | |

### 1.2.10 WP3 – Hardware Platforms for Audio Processing

| WP3 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| Grame, Michon | M6 | M17 | 14 P.m. | 1 P.m. | 1 P.m. | 16 P.m. |

One of the main application of the technology that will be developed in WP1 & WP2 is ultra-low latency audio processing. While the use of an FPGA solves this problem at the computational level, ultra-low latency can only be achieved if the audio ADC/DAC used by the system allows it. Since no FPGA board available on the market offers such feature, we'll have to make our own FPGA-based audio processor hosting ultra-low latency ADC/DAC.

Two systems of this kind will be developed as part of WP3. **Module A** will take the shape of a circuit board connectable to the host FPGA board and providing a stereo audio input and output (2x2) as well as GPIOs. It will target the makers/music technology communities to prototype ultra-low latency/high performance electronic musical instruments, sound effect processors, etc. **Module B** will take the shape of a rackable module wrapped with some casing. Basic parameters will be configurable through a physical interface accessible on the front panel of the device (e.g., pots, buttons, etc.). It will target speaker and microphone arrays for active control of rooms acoustics and will therefore provide a greater number of inputs and outputs (32x32) but no GPIOs. Both modules will be programmable in FAUST through USB. Advanced software parameters linked to the FAUST program (i.e., FAUST DSP parameters) will be accessible through a web interface running a built-in server (see WP4).

While designing Module A will be relatively straight forward, handling 32x32 ultra-low latency channels on Module B will be more challenging and imply the implementation of additional custom hardware. Adding a I2C sensor ADC for GPIOs on Module A and for potentiometers and buttons on Module B should be straight forward.

A research engineer will work at Grame on this project for one year under the guidance of Romain Michon and Pierre Lecomte.

T3.0 – Multi-Channel Ulra-Low Latency Audio ADC/DAC *T3.0 will start at M6 and end at M11. Total effort is 5 P.m.: Grame 5, CITI 0, and LMFA 0.*

This task will focus on designing a multi-channel (32x32) audio interface for ultra-low latency (below 100 $\mu s$). Audio PCM codecs will have to be selected and multiplexed to reach that goal. The deliverable for this task will be a PCB prototype ready-to-be-interfaced with the FPGA board that will be used as part of FAST.

T3.1 – FPGA-Based Audio Processors Design *T3.1 will start at M9 and end at M15. Total effort is 9 P.m.: Grame 7, CITI 1, and LMFA 1.*

This task will center on designing Module A and B. Module A will just be a circuit board connectable/mountable on the host FPGA board. Module B should be rackable and easy to interface with our speaker and microphone array (see WP5). This will involve finalizing circuit design (e.g., balancing inputs and outputs for TRS connections, adding the front panel components I/Os, etc.), designing casings and front panel interfaces, and finally assembling working prototypes of the system.

T3.2 – Final Testing, Production, and Fabrication *T3.2 will start at M15 and end at M17. Total effort is 2 P.m: Grame 2, CITI 0, and LMFA 0.*

The final task of WP3 will focus on polishing the work carried out as part of T2.0 and T2.1 to release the final design of the two audio processors, finalize their datasheets and schematics, and potentially outsource their production if needed. The ability to access the DSP parameters of the FAUST program should "come for free" from the developments conducted in WP4.

| Deliverable of WP3 | Due Date | Leader |
|---|---|---|
| **D3.0** – One 2x2 and One 32x32 Ultra-Low Latency FPGA-Based Audio Processor + Corresponding Report and Schematics | M17 | Grame |
| This hardware deliverable will consist of two ultra-low latency FPGA-based audio processors (one 2x2 channels and one 32x32 channels). It will come with a report describing the design of the system and the corresponding schematics. | | |
| **D3.1** – Report and Schematics on Module A and B | M17 | Grame |
| Report describing the design of the system and corresponding schematics. | | |

### 1.2.11 WP4 – DSP Processor Module

| WP4 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| Grame, Michon | M18 | M23 | 8 P.m. | 1 P.m. | 3 P.m. | 12 P.m. |

The stereo multi-channel ultra-low latency FPGA-based audio processor designed in WP3 (Module A) will be exploited in WP4 to turn into a flexible platform to implement sound effect processors/synthesizers, musical instruments, etc. This will involve the implementation of a software control interface including a GUI (Graphical User Interface) as well as MIDI and OSC support. Existing FAUST DSP libraries will be optimized for our platform and benchmarked in regard to other systems. Finally, demonstrators such as sound effect processors/synthesizers and musical instruments will be prototyped. This work will be carried by a masters-level intern hired as part of FAST.

**T4.0 – Software Control Infrastructure** *T4.0 will start at M18 and end at M20. Total effort is 4 P.m: Grame 3, CITI 1, and LMFA 0.*

FPGA development boards such as the Zybo[31] host an ARM CPU and run a full Linux distribution. The control portion of DSP algorithms with no ultra-low latency needs will run on the ARM processor (see WP2). T4.0 will focus on the development of a software interface running on the Linux system of the board to control the DSP portion running on its FPGA. This interface will include a web GUI (Graphical User Interface) running on an embedded web server accessible from outside of the board (e.g., WiFi, etc.) with a smart-device (e.g., smartphone, tablet, etc.) or a computer and will be controllable through MIDI and OSC.

**T4.1 – Adapt FAUST DSP Algorithms** *T4.1 will start at M19 and end at M22. Total effort is 2 P.m: Grame 2, CITI 0, and LMFA 0.*

The FAUST DSP libraries [40] implement hundreds of algorithms for sound synthesis and processing. None of them was implemented with FPGA applications in mind, hence it is likely that a wide range of optimizations can be carried out to run them on the Module A DSP processor developed in WP3. This task will consist of reviewing and adapting existing algorithms from the FAUST DSP libraries for FPGA applications. Of course, new algorithms could be implemented as well and added to the libraries. This will also be a good time to run benchmarks to compare the performances of our system with traditional platforms.

**T4.2 – Musical Instruments and Sound Effect Processors** *T4.2 will start at M19 and end at M23. Total effort is 6 P.m: Grame 3, CITI 0, and LMFA 3.*

This task will focus on making demonstrators using the work from the two previous tasks. A programmable guitar pedal effect or a modular synthesizer based on Module A will be designed. Physical models of brass instruments based on Finite Difference in Time Domain algorithms will also be implemented [8].In order to demonstrate that an FPGA controller is a component particularly suited for the design of hybrid instruments, the soundboards of a guitar and a piano will be instrumented with sensors and actuators to apply active control algorithms with Module A. The playability of instruments will be validated with musicians. Controlling wind instruments is more challenging [36], but a demonstrator with a pipe will also be designed to test sound control algorithms before the availability of Module B for WP6.

| Deliverable of WP4 | Due Date | Leader |
|---|---|---|
| **D4.0** – DSP Processor Module Software Control Infrastructure | M23 | Grame |
| This deliverable will consist in designing a software control infrastructure for our DSP processor module that will include: a web interface running on its own embedded web server and MIDI and OSC support. | | |
| **D4.1** – FAUST DSP Libraries Adapted to FPGA | M23 | Grame |
| This deliverable will consist in a collection of FAUST DSP libraries adapted for a use on our FPGA-based platform. | | |
| **D4.2** – Set of Sound Effect/Synthesizer Processors and Musical Instruments | M23 | Grame |
| This deliverable will consist in a set of demonstrators that will take the form of guitar pedal effects, modular synthesizer modules, musical instruments implementing active acoustic control, etc. | | |
| **D4.3** – Report on WP4 | M23 | Grame |
| This deliverable will consist of a report summarizing to work carried out in WP4. Its format will allow it to be transformed into a conference paper (e.g., SMC conference or NIME conference, etc.). | | |

### 1.2.12 WP5 – 3D Artificial Reverberation

| WP5 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| LMFA, Lecomte | M18 | M29 | 1 P.m. | 1 P.m. | 11 P.m. | 13 P.m. |

LMFA at École Centrale hosts a listening room with a 22.2 sound spatialization system with possibility for channel expansion. The goal of this work package is to use the 32x32 FPGA-based audio processor developed in WP3 to create a studio in LMFA's listening room where the reverberation of existing or modeled acoustical spaces will be experienced in three dimensions. We aim at organizing public events around this system towards the end of WP5 such as live performances and listening sessions. This work will be carried by a masters-level intern hired as part of FAST.

**T5.0 – Building the Studio** *T5.0 will start at M18 and end at M25. Total effort is 1 P.m: Grame 0, CITI 0, and LMFA 1.*

The goal of this task is to install the 32x32 FPGA-based audio processor developed in WP2 in the listening room of LMFA at École Centrale. This will involve selecting gears, cabling, testing and calibrating the system in order to prepare and keep it up to date for the following tasks of WP5 and WP6.

**T5.1 – Artificial Reverberation Algorithms** *T5.1 will start at M18 and end at M27. Total effort is 7 P.m: Grame 1, CITI 1, and LMFA 5.*

This task will leverage the work carried out during the first eighteen months of WP1 and WP2 to implement artificial reverberation algorithms in FAUST and run them in the setup implemented in T5.0. We'll target algorithms taking advantage of the inner properties of the FPGA (i.e., high-degree of parallelization). In particular, we believe that modal reverb [1] – which is hard/impossible to implement on standard computing platforms – will find a very natural use-case with our system. This type of algorithm works by modeling the frequency response of a real or artificial acoustic space using a bank of second-order IIR filters which can be implemented with biquads. Filtering by the bank is an operation that can be highly parallelized and therefore adapted to the use of an FPGA. Moreover, since the parameters of each biquad can be modified on the fly, the resulting impulse response becomes parametric, which is not possible with linear convolutions. The use of modal reverb will for example allow to "morph" between several rooms. For instance, take the low frequency response of room A and high frequency response of room B, etc. A specific user interface will be developed to allow for the easy control of the reverberation parameters. A possible extension of this technique could rely on spatial sound techniques. In fact, since IRs can be captured "in 3D" using an Ambisonics microphone, the convolution process can be carried out in 3D as well for the accurate rendering of spatial impression [35]. Eventually, it will be possible to experience the reverberation of any existing space in 3D in LMFA's listening room or at home (through binaural/transaural technology) while keeping a high level of control on the system.

[31] https://www.xilinx.com/products/boards-and-kits/1-4azfte.html

**T5.2 – Landmarks Impulse Responses** *T5.2 will start at M24 and end at M27. Total effort is 2 P.m: Grame 0, CITI 0, and LMFA 2.*

The most exciting thing about WP5 is that we'll be able to experience the reverberation of any existing acoustic space in LMFA's listening room. The goal of this work package is to go on the field and make impulse response measurements of landmarks in the Lyon area to establish a catalog of reverberations that will be usable in LMFA's listening room. This database will then be a testimony of the acoustics of the Lyon environment at the time of the project and will be made freely available to the public. As the FAST project seeks to democratize the use of FPGAs to the greatest number of people, it will be possible to immerse oneself at home in a real or virtual place with the help of this database, using binaural/transaural technology.

**T5.3 – Public Events** *T5.3 will start at M27 and end at M29. Total effort is 3 P.m: Grame 0, CITI 0, and LMFA 3.*

This task will take place during the last month of WP5 and will be about opening LMFA's listening room to the public in order for it to experience the reverberation of various landmarks from the Lyon area or from around the world from a single standpoint. Recordings of performances could be played back through the system at specific hours. As the hardware system of WP3 is low-latency, the demonstrations will be interactive and not playback-only: the ambient sound in the studio will be captured with microphones and processed as well. Hence, the public will experience the reverberation of the room they have chosen. These demonstrations will pave the way for WP6. Indeed, the immersion effect can be greatly enhanced if the resulting sound field is precisely and physically controlled, which is what WP6 proposes.

| Deliverable of WP5 | Due Date | Leader |
|---|---|---|
| **D5.0** – Listening Room/Studio For 3D Artificial Reverberation | M25 | LMFA |
| This deliverable will consist of preparing LMFA's listening room for 3D artificial reverberation applications using the system developed in WP3. | | |
| **D5.1** – Software Library in FAUST for 3D Artificial Reverberation | M27 | LMFA |
| This software deliverable will consist of a software library in FAUST for 3D artificial reverberation to be ran on the system implemented in D5.0. | | |
| **D5.2** – Database of 3D Impulse Responses | M27 | LMFA |
| This software deliverable will consist of a database of 3D impulse responses measured in landmarks of the Lyon area. | | |
| **D5.3** – Public Events Around the System Developed in WP3 | M29 | LMFA |
| A series of concerts and interactive listening sessions open to the public featuring the technology developed in WP5. | | |
| **D5.4** – Final Report on WP5 | M29 | LMFA |
| This repport will describe the work carried out as part of WP5 and should be ready to be turned into a conference paper (e.g., SMC conference). | | |

### 1.2.13 WP6 – Spatial Active Noise Control Applications

| WP6 Leader | Start | End | Grame | CITI | LMFA | Total |
|---|---|---|---|---|---|---|
| LMFA, Lecomte | M18 | M41 | 2 P.m. | 1 P.m. | 26 P.m. | 29 P.m. |

In WP6, the LMFA listening room will be turned into a "next generation rehearsal space" for artists and musicians. By this, we mean a room whose acoustic field is physically controllable over a large region. For instance, musicians will be able to interact with the acoustics of the room where they will give their performance (e.g., a singer projecting his voice in a concert hall, musicians rehearsing in concert conditions but do not need to physically go to the concert venue, etc.).

In contrast with the techniques presented in §1.2.6, we wish to follow the approach of spatial active noise control here, whose recent algorithms are promising for controlling a continuous region of space [31, 73]. Thus, the minimization criteria will not be motivated by the listener's auditory system, but by a physical description of the target acoustic field. In contrast with WP5, an additional microphone array will be used here to continuously monitor the reconstructed sound field and feed the spatial active noise control algorithms to converge to the target. This physical approach is much more demanding in terms of sound field control, but we are convinced that it will bring greater realism and also open the way to many more applications: assistance in instrument making, archaeology, virtual reality, acoustic radiation from complex structures (e.g., aircraft engine, etc.).

A typical use case of what we are aiming to achieve in WP6 is shown in Fig. 3. Artists stand in the room within a multi-radius open spherical microphone array. This array allows us to: (1) Decompose the total acoustic field on Spherical Harmonics basis [24] and capture its complexity including directivity [52]; (2) Separate the incoming and outgoing sound fields [11].

When artists produce a primary acoustic field (in blue), the microphone array picks up the radiated field and decomposes it on the spherical harmonics basis. This spatial information is sent to the 3D rendering system, which must generate the appropriate reflections (in green) to reproduce the reverberation of the desired location. Landmarks spatial impulse responses provided by T5.2 will be used here. However, parasitic reflections (in red) occur in the listening room which is not anechoic. In addition, possible noise sources may be present outside the microphone array (in purple). The challenge then is to be able to cancel the reflections and parasitic noise within the microphone volume, where the artists are located and keep the virtual reflections within this volume. To achieve this, a powerful spatial active control algorithm must be used as well as a powerful ultra-low-latency multi-channel system. The 32x32 FPGA-based audio processor developed in WP3 will be used to implement WP6.

**T6.0 – Strategy and Algorithm Selection** *T6.0 will start at M18 and end at M30. Total effort is 8 P.m: Grame 0, CITI 0, and LMFA 8.*

This first task consists of a literature review on classical, multi-channel active noise control [19] and on recent approaches involving sound field decomposition [31, 73]. Based on this knowledge, the chosen strategy and development of a spatial active control algorithm will be carried out. Preliminary simulation results will be provided. Then, the algorithms will be adapted to target an implementation on the 32x32 system of WP3.

**T6.1 – Spatial Active Noise Control Over a Continuous Fixed Volume** *T6.1 will start at M30 and end at M36. Total effort is 11 P.m: Grame 1, CITI 1, and LMFA 9.*

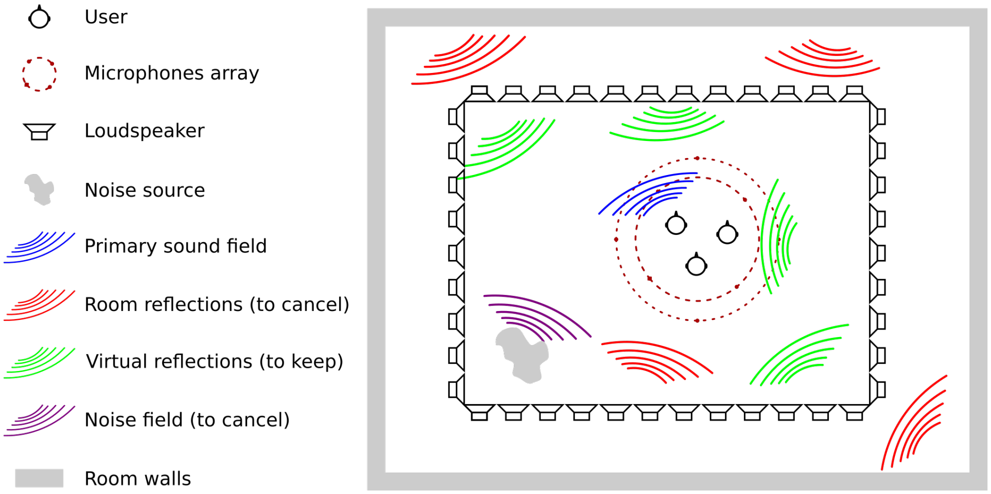

Figure 3: A typical use-case of spatial active noise control: the primary sound field (in blue) is emitted by the artist inside the microphone array (user). The loudspeaker array is used as a secondary source to generate virtual reflections (green) and cancel any room reflections and noise fields (red, purple).

The objective of this task is to validate experimentally, in the LMFA listening room, the spatial active control on a fixed volume. The volume, bounded by a spherical microphone array, is fixed and cannot move. Indeed, to initialize the correction filters required by the spatial active noise control algorithm, an estimation of the acoustic transfer functions between the loudspeakers and the microphone array must be done [31]. As these functions depend on the location of the measurements, they should be remeasured if the correction volume were to move.

This task will be both experimental and theoretical. Indeed, the algorithms developed during the task T6.0 will most likely require some adaptations.

T6.2 – Spatial Active Noise Control Over a Continuous Mobile Volume *T6.2 will start at M36 and end at M41. Total effort is 10 P.m: Grame 1, CITI 0, and LMFA 9.*

In a second step, a more exploratory phase will propose to study the possibility of moving the controled volume. This requires an estimation in-situ of acoustic transfer functions. As this approach can be very demanding, a less constraining approach to the measurement volume can be taken, such as the transaural 3D restitution technique [7] (binaural restitution on a loudspeaker network with cancellation of crossed paths). Indeed, this technique requires to precisely reconstruct the pressure only at two points near the user's eardrums.

| **Deliverable of WP6** | **Due Date** | **Leader** |
|---|---|---|
| **D6.0** – Spatial Active Noise Control Working in Simulated Room | M30 | LMFA |
| This software deliverable will consist of spatial active noise control algorithms running on simulated rooms. | | |
| **D6.1** – Implementation of Spatial Active Field Control Algorithm and Validation Measurements Carried Out in LMFA Listening Room | M41 | LMFA |
| This deliverable will consist of an experimental set-up with spatial active noise control algorithms running on a fixed or mobile volume | | |
| **D6.2** – Final Report on WP6 | M41 | LMFA |
| This report will describe the work carried out as part of WP6 and should be ready to be turned into several journal and conference papers. | | |

## 1.3 Risk Management

FAST is deliberately ambitious, in particular in its scope that covers music, acoustics, DSP, FPGA programming, compilation, etc. However, this ambition is well within the frame of the complementary expertise of the project members (see §2.1)

The first risk is related to the feasibility of WP1 and WP2. Luckily, FAST builds upon the results of a previous project (SyFaLa [55]) where FAUST-generated objects can already be ran on FPGAs using Vivado HLS. The system is currently highly unoptimized so most of the work carried out as part of WP1 and WP2 will be related to improving this existing system to make it more efficient. In practice, this is linked to the main risk of FAST which is computational power. While we know that we'll be able to run a wide range of algorithms on our FPGA-based system, we also know that there will be limits to what can be done. These limits have yet to be determined.

While designing the 2x2 audio DSP processor (Module A) of WP3 should be very straight forward, implementing Module B presents much greater challenges related to its extended number of audio channels (32x32). When having a system with 32 input and output channels at our disposal would be ideal, we could scale it down in case of an issue. In practice, this shouldn't be a problem thanks to use of ambisonics which will make our system very scalable.

WP4 presents very few risks and its implementation should be very straight forward.

WP5 will build upon the success of WP1, WP2, and WP3. Since the algorithms which will be used as part of WP5 (i.e., modal reverb) are quite expensive from a computational standpoint, this will be a good test case for the system implemented in WP1 and WP2. As we know how to implement these algorithms "by hand" (i.e., without going through FAUST) [70], we have the guaranty that in the unlikely scenario where FAUST-generated objects would be too slow, we'd be able to make this system work. Similarly, using a smaller number of audio channels will not be a problem as well and will not compromise the success of WP5.

The topic of spatial active noise control is quite recent and few experimental results are available in the literature. This makes WP6 an ambitious work with a moderate risk that is commensurate with the ambition. However, the approaches towards which we are moving (decomposition of the acoustic field using spherical harmonics) are scalable. This makes it possible to limit the number of channels and the size of the control volume, without changing the theory. The realization of WP6 without the use of the system that will be implemented in the framework of the FAST remains possible, which limits the risk. The contribution of the 32x32 platform will only make WP6 more efficient and will offer new possibilities.

# 2 Organization and Implementation

## 2.1 Scientific Coordinator and His Consortium

Romain Michon is a researcher at GRAME-CNCM and a researcher and lecturer at the Center for Computer Research in Music and Acoustics at Stanford University (USA). His research focuses on embedded systems for real-time audio signal processing, Human Computer Interaction (HCI), New Interfaces for Musical Expression (NIME), physical modeling of musical instruments, digital fabrication techniques for music, and the FAUST programming language.

Since 2018, one of his goal has been to make more accessible the programming of low-level embedded platforms such as microcontrollers [38, 39] for real-time audio signal processing applications through FAUST. These developments were used at the heart of the Amstramgrame project[32] of which he is the main scientific coordinator. Amstramgrame aims at facilitating the teaching of maths and physics through the programming of embedded instruments called the Gramophones. The core idea of Amstramgrame is to make abstract scientific concepts more tangible through sound. Michon's knowledge in embedded systems will be very useful in the context of FAST.

Romain Michon has been invested in the development of FAUST for the past ten years. In particular, he implemented various targets of the language (e.g., Android, microcontrollers, etc.) and he developed an environment for physical modeling of musical instruments based on FAUST.

As an active member of the music HCI community, he developed a wide range of interfaces and musical controllers such as Nuance [41] and the BladeAxe [42]. This year, he's the paper chair of NIME (New Interfaces for Musical Expression)[33] which is the largest and most prestigious conference in this field.

Yann Orlarey is a researcher in Computer Music specializing in programming languages for music creation, sound synthesis, and audio signal processing. He is particularly interested in the creative and artistic use of functional programming, in particular, languages derived from $\lambda$-Calculus and Combinatory Logic. He is the designer, with his colleagues at Grame, of the FAUST programming language and its compiler.

Stéphane Letz is a member of the Audio Working Group of the W3C, and a specialist in real-time audio architectures and music protocols. He is the designer of Jack2, the popular low latency audio server. He is also one of the main developer and maintainer of the FAUST compiler and ecosystem.

Romain Michon, Yann Orlarey, and Stéphane Letz all work at Grame which is a "Centre National de Création Musicale" (CNCM) organized in three departments: music production, transmission/mediation, and computer music research. Grame's research department has been leading the development of the FAUST programming language since its creation in 2004. It has expertise in computer science (compilation), audio DSP, digital lutherie, and HCI (Human Computer Interaction) in general. Thanks to the proximity of all three departments at Grame, research outcomes can easily find their way to professional musical productions and be shared with a general public through the organization of workshops and tight collaborations with the French ministry of education. Hence, the planning of concerts and outreach events around scientific culture leveraging the technology and research outcomes of FAST will be facilitated by Grame's general structure and strike force.

The two other academic partners of the project will be INSA-Lyon through the CITI laboratory (Center of Innovation in Telecommunications and Integration of Service) and Ecole-Centrale-Lyon through LMFA laboratory (Laboratoire de Mécanique des Fluides et d'Acoustique).

CITI lab will provide important skills about compilation, embedded systems, FPGA, high level synthesis and arithmetics operator implementations. Tanguy Risset (professor at INSA-Lyon) has been working for many years on HLS for FPGA (developer of the MMAlpha platform) before successively launching the creation of two Inria team-projects related to embedded systems (Compsys and Socrate). Florent de Dinechin (professor at INSA-Lyon) is one of the world specialist of arithmetic operators implementation on FPGA, he is also the creator of the FloPoCo tool that will be used in the project. His expertise will be a unique opportunity to provide optimal FPGA designs.

LMFA (Laboratoire de Mécanique des Fluides et d'Acoustique) is a JRU (Joint Research Unit – UMR 5509). Its legal representatives are the Ecole Centrale de Lyon (ECL) and its linked third parties the Centre National de la Recherche Scientifique (CNRS), the Institut National des Sciences Appliquées de Lyon (INSA), and the Université Claude Bernard – Lyon I (UCBL). LMFA has leading expertise in acoustics, fluid mechanics, and active sound control. It hosts unique experimental facilities (i.e., anechoic rooms, wind tunnels, sound field synthesis rooms) which will be used for the experimental work of this proposal. Pierre Lecomte (associate professor at UCBL) has been working on sound field capture, control, and synthesis using loudspeakers and digital microphone arrays. He authored a plugin-suite (Ambitools) developed in FAUST for real-time sound field synthesis.[34] Marie-Annick Galland (professor at École-Centrale Lyon) has been working on active noise/sound controls and supervised several PhD on the topic. Sébastien Ollivier (associate professor at UCBL) has been working on sound propagation and physical modeling of musical instruments.

## 2.2 Means for Achieving the Objectives

### 2.2.1 Partner 0: Grame

Staff Expenses

[32] https://www.amstragrame.fr
[33] https://nime2020.bcu.ac.uk/
[34] https://www.sekisushai.net/ambitools

| Item of Expenditure | Grame | CITI | LMFA | Total |
|---|---|---|---|---|
| **Staff on ANR** | 162 620 € | 110 988 € | 73 440 € | 347 048 € |
| **Staff NOT on ANR** | – | 186 672 € | 89 120 € | 275 792 € |
| **Instruments and Material**[a] | 15 300 € | 10 806 € | 20 000 € | 46 106 € |
| **Building and Ground** | – | – | – | – |
| **Service Delivery and Intellectual Property** | 8 400 € | 7 800 € | 3 000 € | 19 200 € |
| **Travel** | 20 000 € | 20 000 € | 20 000 € | 60 000 € |
| **Management and Structure**[b] | 84 369 € | 11 968 € | 9 315 € | 105 652 € |
| **Full Cost** | 290 689 € | 348 234 € | 214 875 € | 853 798 € |
| **Total Requested** | 145 344 € | 161 562 € | 125 755 € | 432 661 € |

a: Including scientific consumables. – b: See the overhead costs sections for each partner below.

Table 1: Requested Means by Items of Expenditure and by Partners

| Kind of Personnel | Grame | CITI | LMFA | Total |
|---|---|---|---|---|
| **Permanent Staff on FAST** | 23 | 0 | 0 | 23 |
| **Permanent Staff NOT on FAST**[a] | 0 | 24 | 21 | 45 |
| **Engineers** | 12 | 0 | 0 | 12 |
| **Postdoc** | 0 | 0 | 18 | 18 |
| **PhD Student** | 0 | 36 | 0 | 36 |
| **Interns** | 6 | 0 | 6 | 12 |
| **Total Effort** | 41 | 60 | 45 | 146 |

a: Salary and other costs not charged on (i.e., not requested in) FAST.

Table 2: Overall Effort by Partners in P.m.

| Kind of Personnel | Effort | Justification |
|---|---|---|
| Permanent Staff | 23 P.m. | Grame's permanent staff will be invested in all the work packages of the project |
| Engineer | 12 P.m. | An engineer will be recruited by Grame to carry out the work linked to WP3 |
| Intern | 6 P.m. | An intern will be hired by Grame to carry out the work linked to WP4 |

### Instruments and Equipment Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| PW[a] | 2 Small FPGA Dev Boards | 600 € | Standard FPGA development boards to be used for R&D by Grame's permanent staff throughout the project |
| PW | 1 Large/Powerful FPGA Deb Board | 1 200 € | Powerful FPGA development board to be used for specialized applications throughout the project |
| PW | 2 Laptops | 2 000 € | Laptops to be used by Grame's permanent staff throughout the project |
| PW | Small Array of Speakers | 6 000 € | Small speaker array (8 speakers) to prototype locally at Grame the systems implemented in WP3 and WP5 |
| PW | Consumable Electronics | 1 000 € | E.g., sensors, ADC, etc. to be used throughout the project |
| WP3 | 1 Small FPGA Dev Board | 300 € | To be used for the 2x2 audio processor (Module A) |
| WP3 | 1 Large FPGA Dev Board | 1 200 € | To be used for the 32x32 audio processor (Module B) |
| WP3 | Electronic Parts | 3 000 € | For the audio processor modules |
| **TOTAL** | 15 300 € | | |

a: Project Wide.

### Service Delivery and Intellectual Property Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| PW[a] | Software Licences | 4 800 € | Software licenses for FPGA programming |
| WP3 | Casing Fabrication | 600 € | For the audio processor modules |
| WP3 | Circuit Board Printing & Assembling | 3 000 € | For the audio processor modules |
| **TOTAL** | 8 400 € | | |

a: Project Wide.

### Travel Expenses

| Item | Unitary Cost | Total |
|---|---|---|
| 2 Journal Papers | 1 000 € | 2 000 € |
| 6 International Conference Trips | 3000 € | 18 000 € |
| **TOTAL** | 20 000 € | |

### Overhead Costs

Overhead costs at Grame consists in 50% for staff expenses ($162,620 \times 0.5 = 81,310$ €) and 7% for other expenses ($43,700 \times 0.07 = 3,059$ €) which corresponds to a total of 84,369 € (see Table 1).

## 2.2.2 Partner 1: CITI

### Staff Expenses

| Kind of Personnel | Effort | Justification |
|---|---|---|
| Permanent Staff | 24 P.m. | CITI's permanent staff will be invested in all the work packages of the project |
| PhD Student | 36 P.m. | A PhD student will be recruited by CITI to work on WP1 and WP2 |

### Instruments and Equipment Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| PW[a] | 2 Small FPGA Dev Boards | 600 € | Standard FPGA development boards to be used for CITI permanent staff throughout the project |
| WP2 | 1 Large/Powerful FPGA Deb Board | 1 200 € | Powerful FPGA development board for adapting faust2fpga workflow by PhD |
| WP2 | 1 Large/Powerful FPGA Deb Board | 1 200 € | Powerful FPGA development board for adapting faust2fpga workflow by PhD |
| PW | 3 Laptops | 6 000 € | one Laptop fot the PhD, one laptop for each permanent member |
| PW | Consumable Electronics | 3 000 € | E.g., sensors, ADC, etc. to be used throughout the project |
| **TOTAL** | 10 806 € | | |

a: Project Wide.

### Service Delivery and Intellectual Property Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| PW[a] | Software Licences | 4 800 € | Software licenses for FPGA programming |
| WP2 | Circuit Board Printing & Assembling | 3 000 € | For the audio processor modules |
| **TOTAL** | 7 800 € | | |

a: Project Wide.

### Travel Expenses

| Item | Unitary Cost | Total |
|---|---|---|
| 2 Journal Papers | 1 000 € | 2 000 € |
| 6 International Conference Trips | 3000 € | 18 000 € |
| **TOTAL** | 20 000 € | |

### Overhead Costs

Overhead costs at INSA-Lyon consists in 8% of the requested amount: $149,594 \times 0.08 = 11,968$ € (see Table 1).

## 2.2.3 Partner 2: LMFA

### Staff Expenses

| Kind of Personnel | Effort | Justification |
|---|---|---|
| Permanent Staff | 21 P.m. | LMFA's permanent staff will be invested in all the work packages of the project |
| Postdoc | 18 P.m. | A postdoc will be recruited by LMFA to work on WP6 |
| Intern | 6 P.m. | An intern will be recruited by LMFA to work on WP5 |

### Instruments and Equipment Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| WP4 | An acoustic guitar, vibration actuators & sensors, microphones, loudspeakers, amplifier, 2x48V supply, cables, mechanical parts | 2000 | To built active control demonstrators: a hybrid guitar, a piano soundboard control setup, a setup to test acoustic active control of a pipe |
| WP5-WP6 | Loudspeakers | 3200 | Loudspeakers to complement or enhance the LMFA listening room |
| WP5 | 5x Spherical Microphone Arrays | 6000 | 3D impulses responses capture and sound field analysis |
| WP5 | 5x Proximity Microphones | 500 | Microphone for capturing "dry" signals on users for demonstrations in the LMFA listening room |
| WP5-WP6 | 2x 12 Channels Amplifier | 1000 | Amplifier for complementary loudspeakers |
| WP6 | 2x Multichannel DSP Boards | 2000 | Real-time DSP boards for the preliminary development of active control algorithms |
| WP6 | 500x Digital MEMS Microphones | 500 | Microphones for the microphone array used in WP6 |
| WP5-WP6 | 2x Powerful Desktop/Laptops | 4000 | Computers for the intern and post-doc |
| WP5-WP6 | Cables, loudspeaker mounts, spare parts | 800 | Miscellaneous material for the assembly, wiring of the installations |
| | TOTAL | 20 000 € | |

a: Project Wide.

### Service Delivery and Intellectual Property Expenses

| Scope | Item | Cost | Justification |
|---|---|---|---|
| WP5-WP6 | Microphone arrays mounts structure | 3000 | Construction of a structure for a double-layer microphone array |
| | TOTAL | 3 000 € | |

### Travel Expenses

| Item | Unitary Cost | Total |
|---|---|---|
| 2 Journal Papers | 1 000 € | 2 000 € |
| 6 International Conference Trips | 3000 € | 18 000 € |
| TOTAL | 20 000 € | |

### Overhead Costs

Overhead costs at Centrale Lyon consists in 8% of the requested amount: $115,440 \times 0.08 = 9.235$ € (see Table 1).

# 3 Project Impact and Benefits

Audio DSP is used everywhere is nowadays life (e.g., smartphones, TVs, cars, tablets, computers, musical instruments, speakers, etc.). Computational power and low-latency are probably the most important technical factors when it comes to these types of systems. FPGAs can help take this field to a new level. This is only possible if this type of platform is easily accessible to non-FPGA engineers. We believe that FAST has the potential to revolutionize the field of real-time audio DSP in the same way than Arduinos disrupted the world of prototyping and engineering. Possibilities unveiled by these developments will impact a wide range of fields and could lead to plethora of technological and industrial applications to design concert halls of a new kind, suppress noise in a car's passenger compartment, diminish vibrations in airplane jet engines, improve existing digital musical instrument technologies, etc. FAST will also give birth to many scientific publications in all these fields.

## 3.1 Expected Impact on the Music Technology and Makers Communities

Just like the BELA [33] brought low-latency to the music technology and makers communities for real-time audio DSP applications a couple of years ago, we want to make ultra-low-latency a reality for everyone! FPGA platforms are currently out of reach to these communities. One of the main ambition of FAST is to facilitate their programming using a high-level DSL (i.e., FAUST) in order for engineers, musicians, and makers to seize this technology and place it at the heart of their work. This will only be possible if we make open source developments a core element of FAST by targeting open source hardware as much as possible and by completely opening the software tools which will be implemented as part of this project. The Module A described in WP3 will target this type of applications by providing a platform for prototyping digital musical instruments, art installations, audio effect processors, etc. We also believe that it could easily find its way in start-up projects and open the door to a wide range of new developments in the music technology industry – a market whose worldwide sales exceed €15 billion.

## 3.2 Expected Impact on the Audio DSP Community

The computational and ultra-low latency performances as well as the accessibility provided by the platform that will be developed as part of FAST will open the door to new research avenues for the audio DSP research community. On the computational plan, the high level of parallelization and modular architecture of FPGAs will provide new opportunities to run specific DSP algorithms in real-time. In particular, physical models of musical instruments using the Finite Difference Scheme Method [8] could greatly benefit from this type of developments. Similarly, new types of artificial reverberation algorithms could be explored/implemented as this is already done as part of WP5. Ultra-low audio latency will make possible all sorts of research developments around active acoustic control of musical instruments [36] and of rooms [72]. Similarly, research around audio spatialization and sound field rendering could be pushed forward by our developments. All these research fields are craving for an easily programmable ultra-low latency multi-channel audio processing system that we plan to develop as part of FAST (Module B – see WP4). In particular, Stanford's CCRMA which is one of the world's leader in the field audio spatialization is extremely interested in such platform to use it to implement new generations of Wave Field Synthesis (WFS) systems, etc.

All the research fields mentioned in this section are among the "hotest" in audio DSP nowadays which is why truly bieleve that our developments will be used by a lot of people.

## 3.3 Expected Impact on the Active Control Community

As mentioned in WP6, FPGA-based solutions for active control applications are the most suitable. The accessibility of this type of systems that will be provided as part of FAST will likely have a significant impact on the active control community.

For instance, FPGAs could be easily used in the context of single-channel controllers: a theme already developed for active headsets [5], for example. Ultra-low-latency allows for a fully digital feedback control to be implemented in this case. More generally, feedback control which was 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 could be the implementation of multichannel controllers. Experiments have already been performed for the design of multichannel feedforward FPGA controllers with the development of architectures adapted from the FXLMS reference algorithm [60]. This could lead the way to real/practical scale applications outside of the lab. In recent advances, some algorithms provide continuous noise control over large areas of space, (i.e., “spatial noise control”) [31, 73].

## 3.4 Expected Impact on the FPGA Community

High Level Synthesis is a technology that enables the high-level programming of FPGAs. However, although this technology is quite mature, it is not really used by non-specialist because of the lack of high-level tools such as the one that will result from FAST. As mentioned before, the HLS tools proposed earlier [58, 9, 63] start from a programming language (C, C++, or Python) which is *instrumented* using pragmas, making them quite hard to use for non-specialists. Our project will reveal that the use of domain specific languages, such as FAUST in our case, opens the doors to new generation FPGA compilers.

We believe that the result of FAST are likely to have an important impact on FPGA research community as it will propose innovative compilation techniques to provide better FPGA compilers.

## 3.5 Other Industrial Applications

Beside the aforementioned applications in the music and makers/prototyping industries, we believe that the work carried out as part of FAST will find its way in the fields of automotive, acoustics, etc. We think that the ultra-low latency audio processor modules (see WP4) developed as part of FAST have an important role to play in that context by providing extended computational power for noise cancellation algorithms and lower latency performances than any other available system. Similarly, a few companies have been trying to develop systems for the active control of room acoustics. Meyer Sound Constellation System[35] is a good example of that. We believe that the platform developed as part of FAST has the potential to take this type of system to the next level.

Industrial developments of active noise control systems have so far been limited either to small spaces (active headsets, low-frequency ducts for aeraulic systems) or to noises of a particular nature (e.g., periodic noise from propeller aircraft, land vehicle engines, etc.). FPGAs based solutions, which offer low-latency and high computational capabilities, will enable the extension of controlled volumes, and possibility of moving it. For instance in the automotive sectors with the reduction of road noise inside the passenger compartment [25].

Another application would be the active treatment of boundary conditions with the realisation of “smart surfaces” for absorption [32, 6], or vibro-acoustic isolation [34, 22, 75]: Active material developments are based on multi-channel control systems combining global control and decentralised feedback systems. The use of FPGAs would enable them to be applied on a large scale, in buildings and also in transport systems (aircraft, turbojet nacelles). The LMFA is developing both the experimental means (MATISSE and CAIMAN test benches, ECL-B3 test bench from Equipex PHARE), and the numerical codes of acoustic propagation [18, 66], within the framework of a strong partnership with Safran Aircraft Engines (ANR ADOPSYS and ARENA industrial chairs). The development of a high-level compiler dedicated to Acoustic Digital Signal processing on FPGA is therefore in high demand by many acoustic researcher for numerous industrial applications.

## 3.6 Expected Impact Towards the Public

The extremely applied type of research that will be carried out as part of FAST has countless of high-level and easily approachable applications which are all very tangible. In that context, we plan to organize a series of events (see WP0) demonstrating and promoting our technology. In particular, concerts and listening sessions open to the public at LMFA’s listening room will allow the audience to experience the acoustics of various landmarks from the Lyon area within the same room. We will prototype a concept of “rehearsing studio of the future” where musicians can come and play in a space whose acoustic can be modified on demand. We will leverage Grame’s experience to organize artist residencies around this system. We also plan to open this studio to local musical professional music institutions (e.g., opera, conservatoire, etc.). We also hope to be able to organize concerts taking advantage of these developments. For example, a concept we’d like to explore would be to have an immersive opera where the acoustic of the concert hall evolves in function of the scenes.

FAST’s research outcomes also have a lot of applications towards digital lutherie/musical instrument making, guitar pedal effect design, modular synthesizers, etc. which are all very approachable by the general public. Grame organizes events yearround (e.g., concerts, exhibitions, etc.) and we plan to showcase the instruments that will be developed as part of FAST in this context.

## References

[1] Jonathan S Abel. Method and system for artificial reverberation using modal decomposition, April 16 2019. US Patent App. 10/262,645.

[2] Jonathan S Abel, Wieslaw Woszczyk, Doyuen Ko, Scott Levine, Jonathan Hong, Travis Skare, Michael J Wilson, Sean Coffin, and Fernando Lopez-Lezcano. Recreation of the acoustics of hagia sophia in stanford’s bing concert hall for the concert performance and recording of cappella romana. In *Proceedings of the International Symposium on Room Acoustics*, Toronto, Canada, 2013.

[3] Jean-Marie Adrien. *The Missing Link: Modal Synthesis*, page 269–298. MIT Press, Cambridge, MA, USA, 1991.

[4] Simon Benacchio, Adrien Mamou-Mani, Baptiste Chomette, and Victor Finel. Active control and sound synthesis – two different ways to investigate the influence of the modal parameters of a guitar on its sound. *The Journal of the Acoustical Society of America*, 139(3):1411–1419, 2016.

[35] https://meyersound.com/product/constellation/

[5] Rivera Benois, Patrick Nowak, and Udo 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.

[6] Benjamin Betgen and Marie-Annick Galland. **A New Hybrid Active/Passive Sound Absorber with Variable Surface Impedance**. *Mechanical Systems and Signal Processing*, 25(5):1715–1726, 2011.

[7] Sunil Bharitkar and Chris Kyriakakis. *Immersive Audio Signal Processing*, volume 121. 2007. Publication Title: The Journal of the Acoustical Society of America.

[8] Stefan Bilbao. *Numerical Sound Synthesis: Finite Difference Schemes and Simulation in Musical Acoustics*. John Wiley and Sons, Chichester, UK, 2009.

[9] Thomas Bollaert. *Catapult Synthesis: A Practical Introduction to Interactive C Synthesis*, pages 29–52. Springer Netherlands, Dordrecht, 2008.

[10] Henri Boutin. *Méthodes de contrôle actif d'instruments de musique - Cas de la lame de xylophone et du violon.* PhD thesis, Université Pierre et Marie Curie, 11 2001.

[11] Yacine Braikia, Manuel Melon, Christophe Langrenne, Éric Bavu, and Alexandre Garcia. Evaluation of a separation method for source identification in small spaces. *The Journal of the Acoustical Society of America*, 134(1):323–331, 2013.

[12] Nicolas Brunie, Florent de Dinechin, Matei Istoan, Guillaume Sergent, Kinga Illyes, and Bogdan Popa. **Arithmetic Core Generation Using Bit Heaps**. In *Field-Programmable Logic and Applications*, September 2013.

[13] Changchun Shi and R. W. Brodersen. An automated floating-point to fixed-point conversion methodology. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, volume 2, pages II–529, April 2003.

[14] Jiwon Choi, Myeongsu Kang, Yongmin Kim, Cheol-Hong Kim, and Jong-Myon Kim. Design space exploration in many-core processors for sound synthesis of plucked string instruments. *Journal of Parallel and Distributed Computing*, 73(11):1506–1522, 2013.

[15] Florent de Dinechin and Matei Istoan. **Hardware implementations of fixed-point Atan2**. In *22nd IEEE Symposium of Computer Arithmetic (ARITH-22)*, pages 34–41, 2015.

[16] Florent de Dinechin, Matei Istoan, and Guillaume Sergent. **Fixed-Point Trigonometric Functions on FPGAs**. *SIGARCH Computer Architecture News*, 41(5):83–88, 2013.

[17] Florent de Dinechin and Bogdan Pasca. **Designing Custom Arithmetic Data Paths with FloPoCo**. *IEEE Design & Test of Computers*, 28(4):18–27, jul 2011.

[18] Yuanyuan Deng, Didier Dragna, Marie-Annick Galland, and Antoni Alomar. **Comparison of Three Numerical Methods for Acoustic Propagation in a Lined Duct with Flow**. In *25th AIAA/CEAS Aeroacoustics Conference*, page 2658, 2019.

[19] Stephen Elliott. *Signal Processing for Active Control*. Elsevier, 2000.

[20] K A Feiste and E.E. Jr Swartzlander. Merged Arithmetic Revisited. *Signal Processing Systems, 1997. SIPS 97 - Design and Implementation., 1997 IEEE Workshop on*, pages 212–221, 1997.

[21] David Griesinger. Improving room acoustics through time-variant synthetic reverberation. In *Audio Engineering Society Convention 90*. Audio Engineering Society, 1991.

[22] Y. Hu, Marie-Annick Galland, and Kean Chen. **Acoustic Transmission Performance of Double-Wall Active Sound Packages in a Tube: Numerical/Experimental Validations**. *Applied Acoustics*, 73(4):323–337, 2012.

[23] Hayato ITO, Shoichi KOYAMA, Natsuki UENO, and Hiroshi SARUWATARI. Three-dimensional spatial active noise control based on kernel-induced sound field interpolation.

[24] Craig T. Jin, Nicolas Epain, and Abhaya Parthy. Design, optimization and evaluation of a dual-radius spherical microphone array. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 22(1):193–204, 2014.

[25] Woomin Jung, Stephen J. Elliott, and Jordan Cheer. Local active control of road noise inside a vehicle. *Mechanical Systems and Signal Processing*, 121:144–157, 2019.

[26] R. Kastner, J. Matai, and S. Neuendorffer. Parallel Programming for FPGAs. *ArXiv e-prints*, May 2018.

[27] Mendel Kleiner and Peter Svensson. Review of active systems in room acoustics and electroacoustics. In *INTER-NOISE and NOISE-CON Congress and Conference Proceedings*, volume 1995, pages 39–56. Institute of Noise Control Engineering, 1995.

[28] Nelson Lago and Fabio Kon. The quest for low latency. In *Proceedings of the International Computer Music Conference (ICMC-04)*, Miami, USA, 2004.

[29] Victor Lazzarini, Steven Yi, Joachim Heintz, Øyvind Brandtsegg, Iain McCurdy, et al. *Csound: a sound and music computing system*. Springer, 2016.

[30] Jakob Leben. Arrp: A functional language with multi-dimensional signals and recurrence equations. In *Proceedings of the 4th International Workshop on Functional Art, Music, Modelling, and Design*, Nara, Japan, 2016.

[31] Pierre Lecomte, Philippe-Aubert Gauthier, Christophe Langrenne, Alain Berry, and Alexandre Garcia. **Cancellation of Room Reflections over an Extended Area Using Ambisonics**. *Journal of the Acoustical Society of America*, 143(2):811–828, 2018.

[32] B. Mazeaud and M.-A. Galland. **A Multi-Channel Feedback Algorithm for the Development of Active Liners to Reduce Noise in Flow Duct Applications**. 21:2880–2899, 2007.

[33] Andrew McPherson and Victor Zappi. An environment for submillisecond-latency audio and sensor processing on beaglebone black. In *Proceedings of the Audio Engineering Society Convention*, Warsaw, Poland, 2015.

[34] Manuel Melon, Philippe Herzog, Azzedine Sitel, and Marie-Annick Galland. **One Dimensional Study of a Module for Active/Passive Control of Both Absorption and Transmission**. *Applied Acoustics*, 73(3):234–242, 2012.

[35] Juha Merimaa and Ville Pulkki. Spatial impulse response rendering I: Analysis and synthesis. *Journal of the Audio Engineering Society*, 53(12):1115–1127, 2005.

[36] Thibaut Meurisse, Adrien Mamou-Mani, Simon Benacchio, Baptiste Chomette, Victor Finel, David B Sharp, and René Caussé. Experimental demonstration of the modification of the resonances of a simplified self-sustained wind instrument through modal active control. *Acta Acustica united with Acustica*, 101(3):581–593, 2015.

[37] Thibaut Meurisse, Adrien Mamou-Mani, René Caussé, Benny Sluchin, and David B. Sharp. An active mute for the trombone. *The Journal of the Acoustical Society of America*, 138(6):3539–3548, 2015.

[38] Romain Michon, Yann Orlarey, Stéphane Letz, and Dominique 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.

[39] Romain Michon, Daniel Overholt, Stéphane Letz, Yann Orlarey, Dominique Fober, and Catinca Dumitrascu. **A Faust Architecture for the ESP32 Microcontroller**. In *Accepted to the Sound and Music Computing Conference (SMC-20)*, Turin, Italy, 2020.

[40] Romain Michon, Julius Smith, and Yann Orlarey. **New Signal Processing Libraries for Faust**. In *Proceedings of the Linux Audio Conference (LAC-17)*, Saint-Etienne, France, 2017.

[41] Romain Michon, Julius O. Smith, Chris Chafe, Matthew Wright, and Ge Wang. **Nuance: Adding Multi-Touch Force Detection to the iPad**. In *Proceedings of the Sound and Music Computing Conference (SMC-16)*, Hamburg, Germany, 2016.

[42] Romain Michon, Julius O. Smith, Matthew Wright, and Chris Chafe. **Augmenting the iPad: the BladeAxe**. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Brisbane, Australia, 2016.

[43] Hideo Miyazaki, Takayuki Watanabe, Shinji Kishinaga, and Fukushi Kawakami. Active field control (AFC)-electro-acoustic enhancement system using acoustical feedback control. *The Journal of the Acoustical Society of America*, 114(4):2342–2342, 2003.

[44] Erdem Motuk, Roger Woods, and Stefan Bilbao. Implementation of finite difference schemes for the wave equation on fpga. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, volume 3, pages iii–237. IEEE, 2005.

[45] Erdem Motuk, Roger Woods, Stefan Bilbao, and John McAllister. Design methodology for real-time fpga-based sound synthesis. *IEEE Transactions on signal processing*, 55(12):5833–5845, 2007.

[46] 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. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 35(10):1591–1604, Oct 2016.

[47] Vesa Norilo. Kronos: a declarative metaprogramming language for digital signal processing. *Computer Music Journal*, 39(4):30–48, 2015.

[48] Yann Orlarey, Stéphane Letz, and Dominique Fobe. *New Computational Paradigms for Computer Music*.

[49] Karthick Parashar, Daniel Menard, and Olivier Sentieys. A polynomial time algorithm for solving the word-length optimization problem. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Diego, United States, November 2013.

[50] Florian Pfeifle and Rolf 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.

[51] Mark A. Poletti. Active acoustic systems for the control of room acoustics. *Building acoustics*, 18(3-4):237–258, 2011.

[52] Martin Pollow, Gottfried Behler, and Bruno Masiero. Measuring directivities of natural sound sources with a spherical microphone array. In *Ambisonics Symposium 2009, June 25-27, Graz*, pages 1–6, 2009.

[53] Mihalis Psarakis, Aggelos Pikrakis, and Giannis Dendrinos. Fpga-based acceleration for tracking audio effects in movies. In *Proceedings of the IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, Toronto, Canada, 2012.

[54] Andrew Reilly and David McGrath. Convolution processing for realistic reverberation. In *Proceedings of the Audio Engineering Society Convention*, New York, USA, 1995.

[55] Tanguy Risset, Romain Michon, Yann Orlarey, Stéphane Letz, Gero Müller, Adeyemi Gbadamosi, Luc Forget, and Florent de Dinechin. **Faust2FPGA for Ultra-Low Audio Latency: Preliminary work in the Syfala project**. In *International Faust Conference (IFC)*, 2020.

[56] Edouard Salze, Emmanuel Jondeau, Antonio Pereira, Simon L. Prigent, and Christophe 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.

[57] Isabelle Schmich and Jean-Paul Vian. CARMEN: A physical approach for room acoustic enhancement system. *CFA/DAGA Strasbourg*, 2004.

[58] Robert Schreiber, Shail Aditya, Scott A. Mahlke, Vinod Kathail, B. Ramakrishna Rau, Darren C. Cronquist, and Mukund Sivaraman. PICO-NPA: high-level synthesis of nonprogrammable hardware accelerators. *VLSI Signal Processing*, 31(2):127–142, 2002.

[59] O Sentieys, D Menard, and N Simon. Id. fix: an eda tool for fixed-point refinement of embedded systems. Design Automation and Test in Europe (DATE) 2014 University booth.

[60] Dongyuan Shi, Woon-Seng Gan, Jianjun He, and Bhan 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. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2019.

[61] Travis Skare and Jonathan Abel. Gpu-accelerated modal processors and digital waveguides. In *Proceedings of the Linux Audio Conference (LAC-19)*, Stanford, USA, 2019.

[62] Yiyu Tan and Toshiyuki Imamura. An fpga-based accelerator for sound field rendering. In *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx-19)*, Birmingham, UK, 2019.

[63] Farhat Thabet, Philippe Coussy, Dominique Heller, and Eric Martin. Exploration and rapid prototyping of DSP applications using systemc behavioral simulation and high-level synthesis. *Signal Processing Systems*, 56(2-3):167–186, 2009.

[64] Dimitris Theodoropoulos, Catalin Bogdan Ciobanu, and Georgi Kuzmanov. Wave field synthesis for 3d audio: Architectural prospectives. In *Proceedings of the 6th ACM Conference on Computing Frontiers*, New York, USA, 2009.

[65] Joseph Tilbian and Andrés Cabrera. Stride for interactive musical instrument design. In *Proceedings of the New Interfaces for Musical Expression Conference (NIME-17)*, Copenhagen, Denmark, 2017.

[66] Renata Troian, Didier Dragna, Christophe Bailly, and Marie-Annick Galland. **Broadband Liner Impedance Eduction for Multimodal Acoustic Propagation in the Presence of a Mean Flow**. *Journal of Sound and Vibration*, 392:200–216, 2017.

[67] Kevin Vaca, Mitchell Jefferies, and Xiaokun Yang. An open audio processing platform with zync fpga. In *Proceedings of the IEEE International Symposium on Measurement and Control in Robotics (ISMCR-19)*, 2019.

[68] Vesa Valimaki, Julian D Parker, Lauri Savioja, Julius O Smith, and Jonathan S Abel. Fifty years of artificial reverberation. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(5):1421–1448, 2012.

[69] Math Verstraelen, Jan Kuper, and Gerard 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)*, Erlangen, Germany, 2014.

[70] Anastasia Volkova, Matei Istoan, Florent de Dinechin, and Thibault Hilaire. **Towards Hardware IIR Filters Computing Just Right: Direct Form I Case Study**. *IEEE Transactions on Computers*, 68(4), 2019.

[71] Xilinx. Reduce power and cost by converting from floating point to fixed point. White paper, WP491 (v1.0), 2017.

[72] Jihui Zhang, Thushara D. Abhayapala, Wen Zhang, Prasanga N. Samarasinghe, and Shouda Jiang. Active noise control over space: A wave domain approach. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(4):774–786, April 2018.

[73] Jihui Zhang, Thushara D. Abhayapala, Wen Zhang, Prasanga N. Samarasinghe, and Shouda Jiang. Active noise control over space: A wave domain approach. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 26(4):774–786, 2018.

[74] Jingbo Zhang, Ganggang Ning, and Shufang Zhang. Design of audio signal processing and display system based on soc. In *Proceedings of the 4th International Conference on Computer Science and Network Technology (ICCSNT)*, Harbin, China, 2015.

[75] Tomasz G. Zieliński, Marie-Annick Galland, and Mohamed N. Ichchou. **Fully Coupled Finite-Element Modeling of Active Sandwich Panels with Poroelastic Core**. *Journal of vibration and acoustics*, 134(2), 2012.

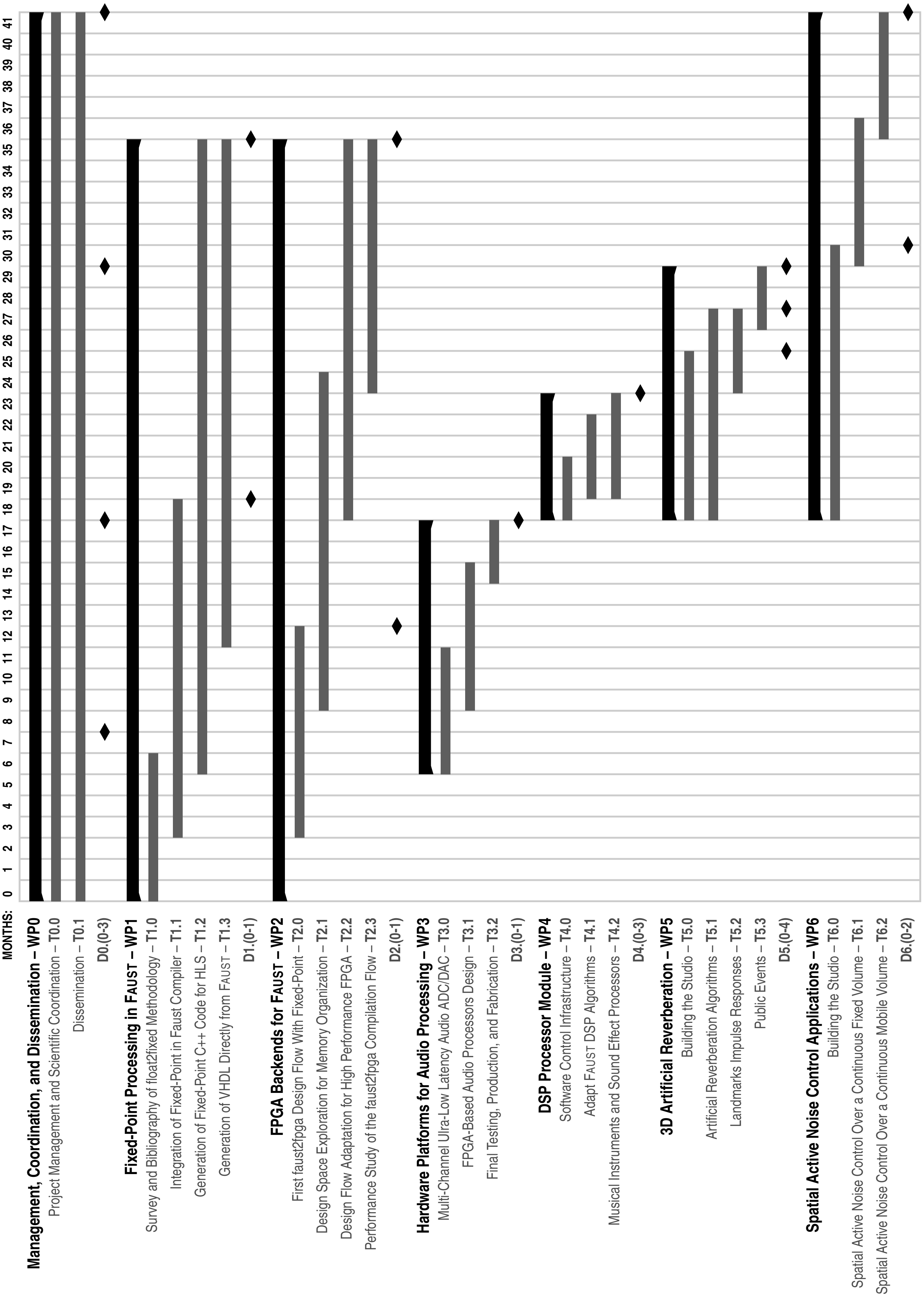

Figure 4: FAST Task Scheduling – Gantt Chart