Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Cellular Automata Applied in Linear Acoustics and Their Implementation in Reconfigurable Hardware

Santiago Guzman Anaya

DISSERTAÇÃO DE MESTRADO

PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS MECATRÔNICOS

Brasília

2024

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Autômatos Celulares Aplicados em Acústica Linear e sua Implementação em Hardware Reconfigurável

Santiago Guzman Anaya

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Orientador: Prof. Dr. Daniel Muñoz Arboleda
Coorientador: Prof. Dr. Henrique Gomes de Moura

Brasília
2024

Anaya, Santiago Guzman.
A769c Cellular Automata Applied in Linear Acoustics and Their Implementation in Reconfigurable Hardware / Santiago Guzman Anaya; orientador Daniel Muñoz Arboleda; coorientador Henrique Gomes de Moura. -- Brasília, 2024.
104 p.

Dissertação de Mestrado (Programa de Pós-Graduação em Sistemas Mecatrônicos) -- Universidade de Brasília, 2024.

1. Autômatos celulares. 2. Acustica linear. 3. Hardware reconfigurável. 4. Guias digitais de onda. I. Muñoz Arboleda, Daniel, orient. II. Gomes de Moura, Henrique, coorient. III. Título

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Mecânica

Cellular Automata Applied in Linear Acoustics and Their Implementation in Reconfigurable Hardware

Santiago Guzman Anaya

Dissertação de Mestrado submetida ao Departamento de Engenharia Mecânica da Universidade Brasília como parte dos requisitos necessários para a obtenção do grau de Mestre

Trabalho aprovado. Brasília, 11 de julho de 2024:

Prof. Dr. Daniel Muñoz Arboleda, UnB/FGA
Orientador

Prof. Dr. Maria Alzira de Araújo Nunes, UnB/FGA
Examinador interno

Prof. Dr. Janier Arias Garcia, UFMG/ Department of Electronic Engineering
Examinador externo

Prof. Dr. Carlos Llanos Quintero, UnB/FT
Examinador suplente

Brasília
2024

Dedico este trabalho à minha mãe, ao meu pai e ao meu irmão,
Apesar da distância, eles sempre me apoiaram.

Acknowledgements

Agradeço ao Centro Internacional de Física Teórica Abdus Salam (ICTP) e à equipe que desenvolveu e disponibilizou os códigos do ComBlock IP Core.

Agradeço ao Prof. Dr. Daniel Muñoz Arboleda e ao Prof. Dr. Henrique Gomes de Moura pela orientação deste trabalho.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES)

Abstract

In acoustics, the field of physics that studies the behavior of sound waves, it is common to use simulations to approximate the behavior of sound waves under certain conditions. These simulations are important in the study and design of room acoustics such as theaters, auditoriums, musical instrumentals, vehicle cabins, among others, which can be described using linear acoustic behavior. However, such simulations are limited by the computational complexity of solving the differential equations describing the acoustic phenomena. To simplify these calculations, numerical methods are often used to reduce the computational cost of these simulations. These approaches reduce the computational complexity of the required calculations and allow the use of hardware accelerators. This research started with a study of acoustic wave propagation using the Digital Waveguide (DWG) model, which is commonly used to emulate sound wave propagation. As a result of this study, a step-by-step formulation for the DWG is presented, introducing a method that considers local impedance changes to accurately characterize reflection phenomena and transmission of sound waves. One of the disadvantages of the DWG model is that the sound is only propagated in one preferential direction between adjacent scattering joints. To overcome this disadvantage, in this thesis, a two-dimensional cellular automata model is developed to emulate acoustic wave propagation (CA2D), which is embedded in a hardware accelerator using Field Programmable Gate Arrays (FPGAs). The cellular automata model is based on simple rules defined by the user that make use of low-computational cost operators. For both the DWG and the CA2D models, a 64×64 elements system was implemented in software using the PYTHON language. This system was first stimulated with a 1 KHz sinusoidal signal and then with a G major guitar chord signal. The response of the system was characterized using Fast Fourier Transform (FFT) and Power Spectral Density (PSD) as evaluation and comparison criteria. Software execution time was estimated for both models. After evaluating the coherence of the CA2D, a hardware architecture was developed to implement it on an FPGA. This hardware architecture was developed in VHDL using Vivado 2018.3 software and embedded in the Pynq-Z2 and ZCU 104 System on Chip (SoC) FPGAs. This hardware architecture uses floating-point addition and multiplication IP cores to perform the necessary arithmetic operations, and the ComBlock third-party IP core that allows the SoC FPGA to communicate with a computer via an Ethernet connection. The hardware implementation consists of three different cells: the *source cell*, the *wall cell*, and the *mesh cells*. A VHDL code generator tool, called *vCA2Dgen*, was developed to facilitate the hardware implementation of the CA2D model. Due to the resources available on the chosen FPGA, only two systems were implemented, one with $10x10$ elements and the other one with $20x20$ elements. The two systems were stimulated with the two signals previously used in software, and the FFT and PSD were also used as validation and comparison criteria. In hardware, the execution time was estimated from the behavioral simulation and measured

using the ILA core tool. The on-board CA in hardware was shown to be approximately 6.12 times faster than the CA2D in software but was limited by the available hardware resources.

Keywords: Cellular automata. Linear acoustics. Reconfigurable hardware. Digital wave guides.

Abstract

In acoustics, the field of physics that studies the behavior of sound waves, it is common to use simulations to approximate the behavior of sound waves under certain conditions. These simulations are important in the study and design of room acoustics such as theaters, auditoriums, musical instrumentals, vehicle cabins, among others, which can be described using linear acoustic behavior. However, such simulations are limited by the computational complexity of solving the differential equations describing the acoustic phenomena. To simplify these calculations, numerical methods are often used to reduce the computational cost of these simulations. These approaches reduce the computational complexity of the required calculations and allow the use of hardware accelerators. This research started with a study of acoustic wave propagation using the Digital Waveguide (DWG) model, which is commonly used to emulate sound wave propagation. As a result of this study, a step-by-step formulation for the DWG is presented, introducing a method that considers local impedance changes to accurately characterize reflection phenomena and transmission of sound waves. One of the disadvantages of the DWG model is that the sound is only propagated in one preferential direction between adjacent scattering joints. To overcome this disadvantage, in this thesis, a two-dimensional cellular automata model is developed to emulate acoustic wave propagation (CA2D), which is embedded in a hardware accelerator using Field Programmable Gate Arrays (FPGAs). The cellular automata model is based on simple rules defined by the user that make use of low-computational cost operators. For both the DWG and the CA2D models, a 64×64 elements system was implemented in software using the PYTHON language. This system was first stimulated with a 1 KHz sinusoidal signal and then with a G major guitar chord signal. The response of the system was characterized using Fast Fourier Transform (FFT) and Power Spectral Density (PSD) as evaluation and comparison criteria. Software execution time was estimated for both models. After evaluating the coherence of the CA2D, a hardware architecture was developed to implement it on an FPGA. This hardware architecture was developed in VHDL using Vivado 2018.3 software and embedded in the Pynq-Z2 and ZCU 104 System on Chip (SoC) FPGAs. This hardware architecture uses floating-point addition and multiplication IP cores to perform the necessary arithmetic operations, and the ComBlock third-party IP core that allows the SoC FPGA to communicate with a computer via an Ethernet connection. The hardware implementation consists of three different cells: the *source cell*, the *wall cell*, and the *mesh cells*. A VHDL code generator tool, called *vCA2Dgen*, was developed to facilitate the hardware implementation of the CA2D model. Due to the resources available on the chosen FPGA, only two systems were implemented, one with $10x10$ elements and the other one with $20x20$ elements. The two systems were stimulated with the two signals previously used in software, and the FFT and PSD were also used as validation and comparison criteria. In hardware, the execution time was estimated from the behavioral simulation and measured

using the ILA core tool. The on-board CA in hardware was shown to be approximately 6.12 times faster than the CA2D in software but was limited by the available hardware resources.

Keywords: Cellular automata. Linear acoustics. Reconfigurable hardware. Digital wave guides.

List of Figures

Figure 1 – A schematic representation of a digital waveguide mesh. The symbols z^{-1} indicate a one-time delay. 26
Figure 2 – Schematic representation of a lossless digital waveguide with measurement points at $x = 0$, $x = cT$, $x = 2cT$ and $x = 3cT$. The symbols z^{-1} denotes a one-time delay, which is used to propagate the input waves in opposite directions along the delay line or waveguide (SMITH III, 1992). 26
Figure 3 – A schematic representation of a Cellular Automata mesh. 31
Figure 4 – Conway's Game of Life Example 31
Figure 5 – A schematic representation of a Cellular Automata Mesh. 32
Figure 6 – A schematic of the AMD Zynq 7000S SoC architecture. (Ref:Xilinx) . . . 33
Figure 7 – Methodology flow chart. 38
Figure 8 – Board layout of the PYNQ Z2 development kit. 39
Figure 9 – Board layout of the ZCU104 development kit. 39
Figure 10 – Equivalent system of spring-mass for acoustic impedance response. . . . 40
Figure 11 – Connection diagram between nodes using spring mass equivalence. . . 41
Figure 12 – A schematic representation of a square waveguide mesh. It can be thought of as an equidistant arrangement of nodes in the Cartesian domain. . . 45
Figure 13 – Some possible impedance arrangements for a dispersion joint. The black dots represent impedances inside a barrier (R_b) while the white dots represent impedances outside the barrier (air: R_m). 45
Figure 14 – Schematic of a real-world scenario with a square waveguide mesh and a barrier placed inside it. The walls of the waveguide also act as barriers with the same impedance. Additionally, two receptors and a sound source were included for certain simulations. 48
Figure 15 – Diagram of the Huygens-Fresnel principle. A spherical source produces spherical wavefronts that excite adjacent points which, once excited, behave like other sources creating new spherical wavefronts. The propagating wave is the product of the envelopes of these new wavefronts. 49
Figure 16 – Scheme of the discrete propagation of waves according to the Huygens-Fresnel principle proposed by (Y. KAGAWA T. TSUCHIYA; TAKEUCHIT, 1998) 49
Figure 17 – Scheme of the Von Neumann neighborhood. 50
Figure 18 – Neighborhood connections and information flow in the three cells. . . . 50
Figure 19 – 5×5 mesh showing the concessions for each type of cell and how the information is transmitted within the mesh. 51
Figure 20 – VHDL entity for the acoustic wall cell. 51

Figure 21 – Simulation results for the Wall Cell. 52
Figure 22 – VHDL Entity of the acoustic mesh cell. 52
Figure 23 – Scheme of the choice of the value of the coefficients and how it affects the behavior of the mesh cell in the transmission of the incident wave . 54
Figure 24 – Scheme of the finite state machine implemented for the operation of the mesh cell. Each state shows the corresponding stage for computing the Equations 3.30. 54
Figure 25 – Equations implemented by the mesh cell. The stages in which these equations are implemented by the finite state machine 24 are shown. . 55
Figure 26 – Behavioral simulation results for the mesh cell. 56
Figure 27 – VHDL Entity of the acoustic Source cell. 56
Figure 28 – Simulation results for the source cell. 57
Figure 29 – Frequency response of receivers 1 and 2 for a sinusoidal signal. Left: Frequency response in Receptor 2 (before the barrier); Right: Frequency response receptor in Receptor 1 (after the barrier). 58
Figure 30 – Simulation results for the wall cell. 60
Figure 31 – FFT results of the 64×64 DWG software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 62
Figure 32 – PSD results for the 64×64 DWG software implementation using the sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 63
Figure 33 – FFT results of the 64×64 DWG software implementation using the G major chord guitar: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 64
Figure 34 – PSD results of the 64×64 DWG software implementation using a G major chord guitar: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 64
Figure 35 – FFT results of the 64×64 CA2D software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 65
Figure 36 – PSD results of the 64×64 CA2D software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 66
Figure 37 – FFT results of the 64×64 CA2D software implementation using a G major Chord signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 66

Figure 38 – PSD results of the 64 × 64 CA2D software implementation using a G major Chord signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier). 67
Figure 39 – DWG and CA2D spectrum comparison for the G major chord guitar signal: a) Receiver 2 (before the barrier). b): Receiver 1 (after the barrier). . . . 68
Figure 40 – 10 × 10 implemented mesh. 69
Figure 41 – Mesh 10×10 27 bits. 69
Figure 42 – Mesh 10 × 10 16 bits. 70
Figure 43 – 20 × 20 implemented mesh. 71
Figure 44 – Mesh 20 × 20 16 bits. 71
Figure 45 – 10 × 10 mesh with a barrier. 72
Figure 46 – Mesh 10 × 10 16 bits and barrier. 72
Figure 47 – 20 × 20 mesh with obstacle. 73
Figure 48 – Mesh 20 × 20 with obstacle 16 bits. 74
Figure 49 – Simulation results of the hardware implementation of the CA2D using 27-bits and 10 × 10 mesh. a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major chord signals, respectively. 74
Figure 50 – Simulation results of the CA2D using 16-bits and 10 × 10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 75
Figure 51 – Simulation results of the CA2D using 16-bits and 20 × 20 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 76
Figure 52 – Simulation results of the CA2D using 16-bits and 10 × 10 mesh with osbtacle: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 77
Figure 53 – Simulation results of the CA2D using 16-bits and 20 × 20 mesh with osbtacle: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 78

Figure 54 – Hardware implementation results of the CA2D using 27-bits and 10×10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 79
Figure 55 – Hardware implementation results of the CA2D using 16-bits and 10×10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 80
Figure 56 – Hardware implementation results of the CA2D using 16-bits and 20×20 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively. 81
Figure 57 – Hardware implementation results of the CA2D using 16-bits and 10×10 mesh with obstacle: a) and b) FFT spectrum and PSD comparison for 1 KHz sinusoidal, respectively. 81
Figure 58 – Hardware implementation results of the CA2D using 16-bits and 20×20 mesh with obstacle: a) and b) FFT spectrum and PSD comparison for 1 KHz sinusoidal, respectively. 82

List of Tables

Table 1 – State of the art of the development of digital waveguides. 30
Table 2 – Related works regarding applications and hardware implementations of cellular automatas. 36
Table 3 – Specifications of the SoC Zynq 7020 used in the PYNQ-Z2 board. 39
Table 4 – Specification of the Zynq Ultrascale+ MPSoC XCZU7EV. 40
Table 5 – Software execution time for the meshes studied in Figure 14 and 29. . . . 68
Table 6 – 10×10 resources consumption. 69
Table 7 – 10×10 resources consumption. 70
Table 8 – 20×20 resources consumption. 71
Table 9 – 10×10 with barrier resources consumption. 72
Table 10 – 20×20 with barrier resources consumption. 74
Table 11 – Audio links with the hearing validation of the CA2D model using the free field receptor. 83
Table 12 – Software execution time and hardware latency for the meshes studied. . . 84
Table 14 – Link with the hearing validation of the CA2D model using the free field receptor. 100

Contents

1 INTRODUCTION . . . 17
1.1 Justification . . . 18
1.2 Research Questions . . . 19
1.3 Objectives . . . 20
1.3.1 General objective . . . 20
1.3.2 Specific objectives . . . 20
1.4 Methodological aspects . . . 20
1.5 Work contributions . . . 22
1.6 Document organization . . . 22

2 BACKGROUND . . . 24
2.1 Digital Waveguides . . . 24
2.1.1 State of Art Regarding Digital Waveguides . . . 28
2.2 Cellular Automata . . . 29
2.3 System on Chip FPGAs . . . 32
2.4 Applications and Implementations of Cellular Automatas . . . 33

3 METHODOLOGY . . . 37
3.1 Development tools and Hardware Setup . . . 37
3.2 Digital Waveguides based on the Impedance Method . . . 40
3.2.1 The *N*-dimensional Digital Waveguide Mesh Formulation . . . 42
3.2.2 Square Waveguide Mesh and Impedance Method . . . 44
3.2.2.1 Propagation in free-field environment . . . 46
3.2.2.2 Propagation with a vertical complete barrier . . . 46
3.2.2.3 Propagation with a vertical incomplete barrier . . . 47
3.2.3 DWG Validation Scenarios . . . 47
3.3 Modeling Acoustic Phenomena using Cellular Automata . . . 48
3.3.1 Wall Cell . . . 50
3.3.2 Mesh Cell . . . 52
3.3.3 Source Cell . . . 55
3.3.4 CA Validation Scenarios . . . 57
3.4 Automatic VHDL Code Generator for the CA2D Model . . . 58
3.5 Validation proposal for the FPGA Implementations . . . 58
3.5.1 Validation using Behavioral Simulation . . . 59
3.5.2 Hardware-in-the-Loop for Acoustic Simulation . . . 59
3.5.3 Validation Criteria . . . 60

4 RESULTS . . . 62
4.1 Software Validation . . . 62
4.1.1 Results for the Digital Waveguide Model . . . 62
4.1.2 Results for the Cellular Automata Model . . . 63
4.2 Hardware Implementation of the CA2D model . . . 69
4.2.1 10×10 Mesh on the ZCU104 Board . . . 69
4.2.2 10×10 Mesh on the Pynq Z2 Board . . . 70
4.2.3 20×20 Mesh on the ZCU104 Board . . . 70
4.2.4 10×10 Mesh with a Barrier on the Pynq Z2 . . . 72
4.2.5 20×20 Mesh with a Barrier on the ZCU104 . . . 73
4.3 Hardware Validation using Behavioral Simulation . . . 73
4.3.1 Mesh 10×10 using 27-bits on the ZCU104 . . . 73
4.3.2 Mesh 10×10 16-bits on Pynq Z2 . . . 75
4.3.3 Mesh 20×20 using 16-bits on ZCU104 . . . 75
4.3.4 Mesh 10×10 using 16-bits with obstacle . . . 76
4.3.5 Mesh 20×20 using 16-bits with obstacle . . . 76
4.3.6 Discussion regarding the behavioral simulation . . . 76
4.4 Validation of the Physical Implementation . . . 79
4.4.1 Mesh 10×10 using 27-bits on the ZCU104 . . . 79
4.4.2 Mesh 10×10 using 16-bits on the Pynq Z2 . . . 79
4.4.3 Mesh 20×20 using 16-bits on the ZCU104 . . . 80
4.4.4 Mesh 10×10 using 16-bits on the Pynq Z2 with obstacle . . . 80
4.4.5 Mesh 20×20 using 16-bits on the ZCU104 with obstacle . . . 80
4.4.6 Discussion regarding the physical implementation . . . 82
4.5 Hearing Validation of the CA2D Model . . . 83
4.6 Computational Performance Results . . . 83

5 CONCLUSIONS . . . 85
5.1 Future Works . . . 86

REFERENCES . . . 88

6 RESUMO ESTENDIDO EM LÍNGUA PORTUGUESA . . . 95

1 Introduction

Natural phenomena have always been of interest in all fields of research, both in the natural sciences, such as physics and biology, and in the applied sciences, such as engineering. This interest is reflected in efforts to reproduce and apply natural behavior, which requires a mathematical modeling process. In this process, it is common to use Partial Differential Equations (PDEs) to describe natural phenomena. These equations contain one or more derivatives of an unknown function, typically dependent on one or more variables. The variables usually represent physical quantities, while their derivatives represent the change in those variables under the conditions imposed by the equation. Solving this type of equation allows these behaviors to be simulated computationally.

An example of this physical phenomenon is the propagation of waves. This phenomenon is described by the wave equation. This equation describes the behavior of stationary mechanical waves such as sound waves. In engineering there are several applications where sound propagation modeling can be used, for example in room acoustic problems (PIND et al., 2019), acoustic synthesis of sound (WOODHOUSE; POLITZER; MANSOUR, 2021), musical instrument simulation (CHATZIIOANNOU; VAN WALSTIJN, 2015), vocal tract simulations (ARNELA et al., 2016) or cabin sound studies (MORDILLAT et al., 2021).

PDEs, such as the wave equation, are typically represented in continuous space (time domain). This means that to compute an acoustic simulation, it is necessary to solve the wave equation at each instant of time throughout the simulation time. In addition, in many simulation cases, complex simulation conditions are considered, which increases the complexity of solving the wave equation (THIES, 2018), making this calculation a challenging numerical task and a time-consuming process (BOURNEZ, 2020). For this reason, simplifications and conditions are commonly applied to PDEs to constrain them to certain limits where solutions are feasible.

Due to the complexity, under certain conditions, of finding analytical solutions to the PDEs, it is common to resort to numerical methods to approximate a solution. Numerical methods are interactive mathematical methods that use discretizations to transform a continuous problem into a discrete one, dividing the problem into simpler parts to solve, allowing an approximation to solutions for the PDEs. Some of the most widely used numerical methods are the Finite Element Method (FEM), the Boundary Element Method (BEM), and the Finite Difference Method (FDM), which are general methods that can approximate solutions to any PDE. On the other hand, for the wave equation, there is a numerical method in the time domain, called Digital Waveguide (DWG), that was developed based on the concept of transmission and reflection of sound waves in a one-dimensional propagation medium.

This type of approach where a numerical model is developed based on the physical concept behind the behavior allows the use of a computational model that takes into account local interactions such as cellular automata (CA) to emulate the sound propagation.

1.1 Justification

Digital waveguides combines the physical principles of waves and numerical discretizations to emulate the behavior of acoustic waves. This model digitizes a physical waveguide in which a series of discrete elements or nodes are connected by a time delay line.

This model is ideal for describing phenomena in linear acoustics, which focuses on the propagation of acoustic waves under the assumption of small and linear perturbations in the medium. Specifically, this implies that variations in pressure, density, and velocity are sufficiently small to allow the equations governing the phenomenon to be linearized. Within this framework, the media are considered homogeneous and isotropic, with impedance variations representing obstacles assumed to have no thickness. Similarly, boundary conditions (such as walls) are modeled as thin and perfectly reflective, ensuring no changes to the propagation medium. Under these assumptions of linearity, the model captures wave propagation phenomena such as reflection and transmission. In this context, the DWG model does not take into account propagation effects that alter wave speed, such as diffraction or dispersion, nor does it include phenomena that generate new sound frequencies, such as multi-frequency interferences or wave-obstacle interactions, including radiation effects.

This linearity conditions allow the superposition principle to be applied to room acoustic systems. This principle states that when two or more independent disturbances act simultaneously on a system, the total response of the system is equal to the algebraic sum of the individual responses (SADD, 2021). For wave propagation phenomena, when two or more waves meet at the same point in space, the resulting wave is the sum of the amplitudes of the individual waves at that point in space.

Digital Waveguides were initially designed as a one-dimensional solution, employing the acoustic tube concept to approximate the propagation behavior of acoustic waves between two dispersion junctions. However, the accuracy of this approximation diminishes as the problem's dimensionality increases. For instance, in two-dimensional scenarios, acoustic waves propagate through concentric wavefronts that expand in all directions within the problem domain. Since DWG models account for wave propagation in only one direction (vertical or horizontal) they fail to fully capture the propagation behavior in higher-dimensional contexts. This limitation motivates the development of discrete models capable of accommodating wave propagation in non-preferential directions.

Cellular Automata (CA) is a computational model commonly used to simulate dynamical systems. It operates in discrete time steps and consists of a lattice of cells, each of

which can exist in one of several finite states. The state of each cell is updated at each time step according to predetermined rules that consider the current state of the cell and the states of its neighboring cells. These rules are typically derived from the equations governing the physical behavior of the phenomenon and rely on simple computational operations to update all cells simultaneously. Due to these features, CA is well-suited for developing two-dimensional models of acoustic wave propagation, as it maintains fidelity to the propagation phenomenon while significantly reducing the computational cost associated with solving the wave equation.

The reduction in computational complexity enables the use of hardware accelerators to enhance the performance of cellular automata. These hardware accelerators are devices specifically designed to perform certain tasks more efficiently than general-purpose processors. Commonly, Graphical Processing Units (GPUs) and Field-Programmable Gate Arrays (FPGAs) are employed as accelerators. GPUs specialize in graphics processing and are optimized to perform a variety of sequential tasks, significantly enhancing performance in certain applications. Conversely, FPGAs are high-performance devices that can be reprogrammed for various applications. This reprogrammability allows for the design of custom digital circuits that can be reconfigured as needed.

One of the main challenges of implementing CA models using FPGAs is the complexity of updating the hardware description of a system with large parallel structures. This implementation is commonly a time-consuming and error-prone process. In this context, developing automatic tools for generating the VHDL code could significantly accelerate the hardware implementation process.

1.2 Research Questions

The research questions that guide this work are the following:

1. Is it possible to develop a cellular automata model that takes into account the phenomena of reflection and transmission of acoustic waves to emulate the propagation of sound in bi-dimensional acoustic rooms?

2. Can the use of cellular automata model reduce the computational cost of the acoustic bi-dimensional rooms simulations based on traditional PDE-based numerical methods such as FEM, BEM, or FDM?

3. Does the implementation of this CA model in hardware (using System-on-Chip FPGA) increase the performance of the room acoustics simulations?

1.3 Objectives

1.3.1 General objective

To implement a cellular automata model for the simulation of the propagation of acoustic waves in two dimensions. This model must be based on simple rules implemented using low-computational cost operators and must be embedded in System-on-Chip FPGAs to explore its intrinsic parallelism. In addition, the proposed model must consider the non-preferential direction of the wave propagation, overcoming the disadvantages of the DWG models.

1.3.2 Specific objectives

- To develop an two-dimensional cellular automata model to emulate sound propagation in acoustic room applications.
- To implement the two-dimensional cellular automata model for room acoustic simulations using SoC-FPGAs.
- To develop an automatic VHDL code generator tool that allows designers to change the mesh size, the source, and receptor positions, and the positions of the internal obstacles.
- To develop a hardware-in-the-loop simulation that enables the integration of the hardware-based CA model to an acoustic simulator running in a desktop.

1.4 Methodological aspects

The entire research was divided into four stages:

1. In the first stage, the Digital Waveguide (DWG) model was employed to study the behavior of two-dimensional acoustic waves in acoustic rooms. These studies facilitated an understanding of the propagation and reflection phenomena of acoustic waves under these conditions. This understanding of acoustic phenomena served as a starting point for developing the acoustic cellular automaton reference model, known as CA2D. Using these two models, simulations of acoustic rooms were conducted in the Pycharm development environment, which specializes in the Python language. In these simulations, obstacles were included to evaluate the reflection behavior. In both cases, Fast Fourier Transform (FFT) and Power Spectral Density (PSD) were utilized to evaluate the simulation results. Additionally, the cProfile profiler was employed to estimate the runtime.

2. In the second stage, the necessary hardware architectures to replicate the CA2D reference model were developed. For this purpose, three types of cells were proposed:

   - Wall cells: These cells are responsible for producing reflections when acoustic waves interact with a wall. The architecture uses a negation of its input signal and routes it to its output.
   - Mesh cells: These cells propagate a signal through the CA2D. Their architecture is based on a finite state machine that performs the necessary calculations for the propagated pressure values.
   - Source Cells: These cells insert signals into the CA2D. Their architecture redirects an input value to their outputs, thereby transmitting the information of a given signal to the interior of the CA2D.

   All these hardware architectures were developed using the hardware description language VHDL (VHSIC Hardware Description Language). The integration of these three types of cells into a CA2D organism was accomplished using a generator called vCA2Dgen. This code generator facilitates the connection of all cells within the mesh and allows the creation of CA2D of different sizes, enabling alteration of the positions of the source cell and the receiver, as well as the insertion of obstacles into the mesh.

3. The third stage involved the implementation and validation of the CA2D model on a SoC-FPGA. Two mesh sizes, 10×10 cells and 20×20 cells (with and without obstacles), were implemented using the Pynq Z2 and ZCU104 FPGAs, respectively. For these mesh sizes, behavioral simulations with automatic reading and writing of text files were developed. The physical implementations employed a hardware-in-the-loop scheme using an IP core called Comblock to facilitate communication between a computer and the FPGA via an Ethernet connection. All physical implementations of the CA2D were evaluated in terms of resource consumption, power consumption, latency, thorughput, and timing reports.

   The acoustic performance results of the CA2D were evaluated using FFT and PSD for both the behavioral simulation and the physical implementation. The FFT is used to compare the frequency components between the source signal and the receptors. On the other hand, the PSD is used to evaluate the power of the signal aiming to verify the signal attenuation produced by the model. Acoustic performance validation was conducted using two types of signals: a 1KHz frequency sinusoidal signal and a G major guitar chord. Additionally, the latency of the physical implementations was measured.

4. Finally, for the $20 \times 20 Cells$ mesh, the CA2D runtime estimate in software and the embedded CA2D latency were used to calculate the speedup factor.

1.5 Work contributions

The main contributions of this work can be summarized as follows. a) A cellular automata model was proposed for simulating acoustic phenomena, extending the one-dimensional cellular automata model previously investigated (PEREIRA, 2022) to two dimensions. This model is called CA2D model; b) An automatic VHSIC Hardware Description Language (VHDL) code generator tool was developed, facilitating the implementation of the proposed CA2D model on FPGAs. This code generator makes it possible to change the size of the mesh and provides more flexibility in placing the source or obstacles throughout the mesh space; c) The third contribution is the development of a hardware-in-the-loop (HIL) system for acoustic simulation using the proposed CA2D model. This HIL scheme is based on the ComBlock IP-core (MELO, 2019), which enables intra-chip communication between the ARM processor and the FPGA, and facilitates communication with an external PC through the Ethernet port.

In addition to these contributions, two publications were addressed. The first one called *FPGA Implementation of Staggered Cellular Automata for Wave Propagation Simulation* was presented at the XI Southern Programmable Logic Conference (SPL) that was held in San Luis, Argentina, 2023 (PEREIRA et al., 2023). This staggered approach is based on previous unidimensional CA studies and allows large systems to be simulated in despite the limited amount of resources available on the FPGA.

The second publication, entitled *A Comprehensive Digital Waveguide Formulation using the Impedance Method for Acoustic Simulation*, was published in the Mechanical Systems and Signal Processing (MSSS) Journal (ANAYA et al., 2025). In this paper, a comprehensive mathematical formulation of the *impedance method* for DWG is reported and didactical examples are presented for obtaining the dispersion coefficients in one and two-dimensional cases.

1.6 Document organization

Chapter 2 provides a background on the topics pertinent to the development of this work. It introduces the model of digital waveguides, along with the underlying theoretical concepts. A review of the state-of-the-art applications of this model in acoustics is also presented. The chapter then discusses the concepts of cellular automata, including a historical context of their development. Following this, the concepts of SoC-FPGAs are introduced, and the use of cellular automata with these platforms is contextualized. The chapter concludes with a review of the state-of-the-art works that combine cellular automata and FPGAs.

Chapter 3 outlines the methodology of this work. It begins with an introduction to the hardware development and configuration tools. Next, the impedance method is

incorporated into the digital waveguide model, followed by a mathematical development of this inclusion. The chapter then details the development of the cellular automaton for acoustic wave propagation (CA2D), including a description of the implemented cells and their hardware architecture. For both the digital waveguide (DWG) and CA2D models, the case studies used for evaluation are presented. The chapter concludes with an overview of the SoC-FPGA scheme used for hardware implementation.

Chapter 4 presents the obtained results. Initially, the software simulation results for both the DWG and CA2D models are introduced. Time execution was used to compare the performance of these two implementations. Subsequently, the hardware implementations are discussed, with two sets of results presented: those corresponding to the testbench of the implemented meshes and those for their physical implementation. Finally, a performance comparison is made between the software implementation in CA2D and its hardware counterpart.

Chapter 5 presents the conclusions of this work and discusses potential future research directions based on the results of the two-dimensional CA2D model implemented in this study.

2 Background

This chapter presents some theoretical foundations of the topics discussed in this work. First, the digital waveguide model for acoustic problems is introduced, along with a portion of the mathematical foundation of this model and a summary of relevant research. Second, the cellular automata model is introduced, including its general concepts and some significant cellular automata developed from this model. Third, the concept of System on Chip (SoC) FPGA is presented, discussing how the use of these tools can impact the performance of cellular automata.Finally, a review of works that use cellular automata in conjunction with hardware accelerators is provided.

2.1 Digital Waveguides

Acoustics is the branch of physics that studies the behavior of mechanical waves in various media (gases, liquids, and solids). This branch includes topics such as vibration, sound, ultrasound, and infrasound. The study of these phenomena revolves around the wave equation (Equation 2.1), which is a partial differential equation (PDE) of second order that describes the propagation of stationary waves.

$$\nabla^2 P = \frac{1}{c_0^2}\frac{\partial^2 P}{\partial t^2} \tag{2.1}$$

where, $\nabla^2 P$ is the Laplacian operator, t, P, and c_0 represent the time, the pressure and propagation speed of the acoustic wave, respectively.

The solution of this equation is often conditioned on the boundary and initial conditions specified in the problem definition (SKEEL; BERZINS, 1990). Usually, analytical methods such as separation of variables, integral transformation, change of variables, are used to find a solution to this PDE (LAMOUREUX, 2006). However, due to the sensitivity of this equation with their initial conditions, in many problems only an approximate solution can be obtained. To perform this approximation numerical methods such as finite elements methods (OKUZONO et al., 2016) (LI et al., 2014), boundary elements methods (LIU, 2019) (FISCHER; GAUGER; GAUL, 2004), and finite differences methods (TREFETHEN, n.d.), (SUN; WU, 2007) are commonly used. Digital Waveguides (DWG) represent a specialized case of the finite differences method, and this section is dedicated to elucidating this concept.

A digital waveguide is a model that can be used to approximate the behavior of acoustic waves. This model is based on the solution of the wave equation proposed by Jean le Rond d'Alembert (ROND D'ALEMBERT, 1747), who studying mathematically the vibrating string problem, proposed an equation in which the wave propagation is described

by a superposition of two traveling functions (left and right traveling waves) in opposite directions, as show in the equation 2.2

$$p(x,t) = f\left(t - \frac{x}{c}\right) + g\left(t + \frac{x}{c}\right), \tag{2.2}$$

where t and x are the time and space variables, f and g are arbitrary functions that can be differentiated twice and c is the speed of sound.

This equation can be used to perform a digitization process. To do this, it is necessary to define a sampling variable X as the distance traveled by the wave during a time interval T, where $X = cT$. This sampling of space and time modifies the variables associated with the waveguide solution, i.e., $x \rightarrow x_m = mX$ and $t \rightarrow t_n = nT$. Substituting these variables into the equation 2.2, the digital waveguide is given by

$$p(x_m,t_n) = f\left(nT - m\frac{X}{c}\right) + g\left(nT + m\frac{X}{c}\right), \tag{2.3}$$

This equation (2.3) is commonly rewritten as

$$p(x_m,t_n) = p_r\left[(n-m)T\right] + p_l\left[(n+m)T\right], \tag{2.4}$$

where p_r and p_l refer to the right and left pressures, respectively. To obtain a notation that more accurately reflects the two components of the traveling wave that propagate in opposite directions, the variable T is dropped and, p_r and p_l are rewritten as p^+ and p^-, respectively

$$p(x_m,t_n) = p^+\left[(n-m)\right] + p^-\left[(n+m)\right], \tag{2.5}$$

where the expressions $p^+(n-m)$ and $p^-(n+m)$ denote the output of a delay line with a delay of m samples from the inputs $p^+(n)$ and $p^-(n)$, respectively. This equation is the starting point for the digital waveguide model because it describes the behavior of wave propagation at a given point in space.

This model was proposed and developed by Julius O. Smith III in 1992 (SMITH III, 1992) and was built by digitizing a larger region of space where a set of points, called scattering nodes or mesh nodes, are connected, as shown in Figure 1. These nodes record the wave properties as a function of time. Thus, the wave propagation is due to unidirectional waves propagating in a spatial array of ideal lossless waveguides.

The connection between the scattering nodes typically does not account for changes in the acoustic impedance of the medium. The challenge then is to connect the scattering nodes in a way that represents the impedance transitions that occur at the boundaries between two different nodes with different impedances. The digital waveguide solves this problem by using a bidirectional unitary delay line. A schematic of this bidirectional link is shown in Figure 2.

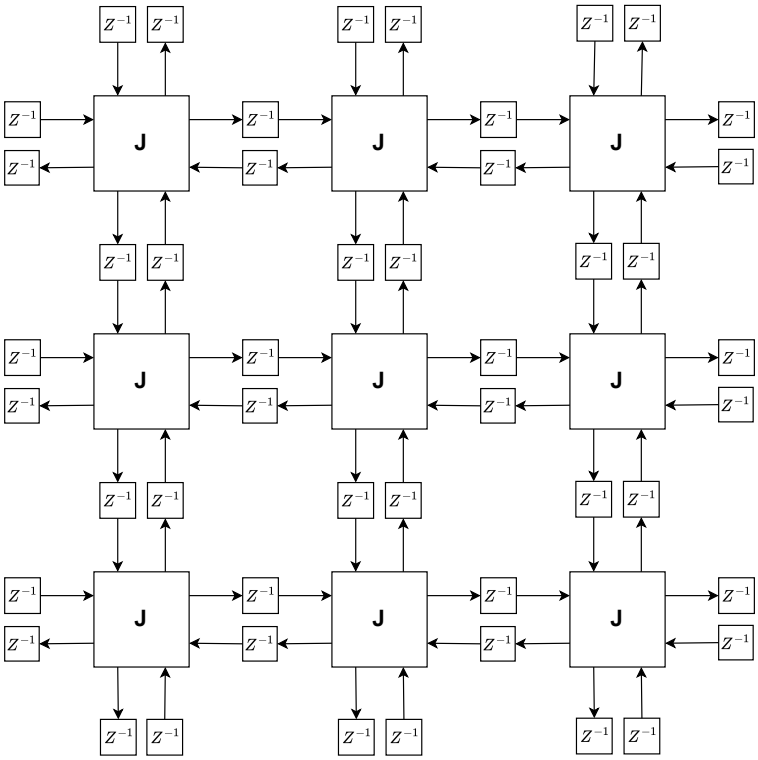


Figure 1 – A schematic representation of a digital waveguide mesh. The symbols z^{-1} indicate a one-time delay.

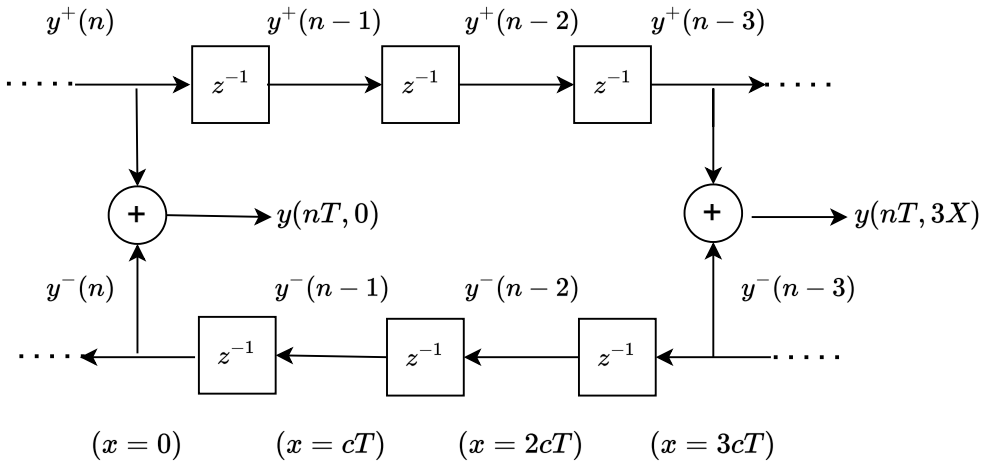


Figure 2 – Schematic representation of a lossless digital waveguide with measurement points at $x = 0$, $x = cT$, $x = 2cT$ and $x = 3cT$. The symbols z^{-1} denotes a one-time delay, which is used to propagate the input waves in opposite directions along the delay line or waveguide (SMITH III, 1992).

To account for this change in impedance, a method called Impedance Method is used in this work (described in Section 3.2) (MOURA, 2006). To do this, it is necessary to introduce a concept of acoustic impedance, which is discussed below.

The d'Alembert solution could be used to describe the motion of sound waves in an ideal fluid. In such a fluid, the motion of sound waves can be represented by the motion of transverse waves along an ideal vibrating spring. In accordance with the Equation 2.2, the velocity of a point inside the spring can be obtained by the sum of two motion packets, $\mathbf{V} = \mathbf{V}^+ + \mathbf{V}^-$. This kind of relation can be extended to elastic forces.

$$\mathbf{f} = \mathbf{f}^+ + \mathbf{f}^-. \tag{2.6}$$

The forces $\mathbf{f}^+$ and $\mathbf{f}^-$ are related to the velocities $\mathbf{V}^+$ and $\mathbf{V}^-$ by the following expressions (MORSE, 1949),

$$\mathbf{f}^+ = R_m \mathbf{V}^+ \tag{2.7}$$

$$\mathbf{f}^- = -R_m \mathbf{V}^-, \tag{2.8}$$

where R_m is the resistance to the particles movement inside the spring.

An analogy between the particles movements into a vibrating string and an acoustic tube with cross-section A can be made. In particular, the force $\mathbf{f}$ is related to the acoustic pressure p inside the waveguide, $\mathbf{V}$ is related to the sound velocity c of an infinitesimal volume of air with density ρ travelling in the tube, and R_m is related to the acoustic characteristic impedance R defined for the waveguide. There is a relationship between the characteristic impedance R and the acoustic wave impedance Z, given by

$$R = \frac{\rho c}{A} = \frac{Z}{A}. \tag{2.9}$$

An algorithm that evaluates the local impedance change between a node and its neighbors must consider the impendance of the node and the impedance of its neighbors. Considering a digital waveguide mesh as shown in Figure 1, it is necessary to adjust the characteristic impedance between node i and j ($R_{i,j}$) when the its acoustic impedance differs from its neighbor. This can be done in a very simple way using the following rules. For each neighborhood (i,j) with acoustic impedance (Z_i,Z_j), the following can be considered:

if $Z_i \neq Z_j$ **then**
    $R_{i,j} \leftarrow \alpha Z_m$
else
    $R_{i,j} \leftarrow \beta Z_i$
end if

where Z_m denotes the impedance of the medium, Z_i represents an impedance calculated from the neighbhood impedance. Thus, if the impedance between two nodes differs, the characteristic impedance of the current node is adjusted using the impedance of the medium; otherwise, the current node adopts the impedance of its neighbor. The coefficients α and β may be utilized for the purpose of quantitative validation correlations. However, for the purpose of qualitative validation, these coefficients may be set to unity. It is important to note that the acoustic impedance values, represented by $Z_i = [Z_m, Z_1, Z_2, ...]$, must be available at this point to represent the medium impedance transitions.

2.1.1 State of Art Regarding Digital Waveguides

This digital waveguide model has been employed in both theoretical and practical research related to the propagation of acoustic waves. Some of the research that has developed the theory of this numerical representation is described below.

- (MULLEN; HOWARD; MURPHY, 2006) presented an alternative development to the standard one-dimensional Kelly-Lochbaum vocal tract model, from which he proposed a two-dimensional vocal tract model using digital waveguides.
- (MURPHY; BEESON, 2007) investigated the terminations of a digital waveguide by studying the Kirchhoff boundary model and a dispersion-based wave model, ultimately proposing a hydronic termination implemented in a two-dimensional hybrid triangular digital waveguide mesh.
- (HACIHABIBOGLU; GUNEL; KONDOZ, 2008) presents the numerical derivation of a DWG for a general N-dimensional mesh that was used to derive the equations associated with different mesh topologies in both two and three dimensions. These topologies were triangular, rectilinear, hexagonal, cubic, tetrahedral, body-centered cubic, and compact cubic.
- (GERMAIN; EVANGELISTA, 2009) presents a guitar pick model that considers the interaction of the player with the guitar strings using a pick. In this model, the pick is simulated as a beam that bends as it interacts with the string. These considerations lead to a model that uses the DWG to simulate wave propagation through the strings. This model has been used in the synthesis of guitar sounds.
- (EVANGELISTA; ECKERHOLM, 2010) uses a digital waveguide to simulate the propagation of transverse waves along a guitar string. In addition, a model is developed to simulate collisions of inelastic and semi-elastic strings based on nonlinear dispersions. This approach explores some of the most common interactions in guitar playing, such as plucking, imperfect fingering, string tapping, and harmonic generation.

- (HACIHABIBOGLU; GUNEL; CVETKOVIC, 2010) presented a method for simulating directional microphones using the DWG. In this method, the directional microphone response takes into account the instantaneous intensity and the estimation of the direction of incidence. This study analyzes the calculation of instantaneous intensity and directional accuracy for different mesh topologies such as cubic, tetrahedral, body-centered cubic (BCC), and cubic close-packed (CCP). This model was used to simulate the behavior of a cardioid microphone in a cubic room using an asymmetric trivariate Gaussian pulse as the source.

- (SPEED; MURPHY; HOWARD, 2013) represents an analog model of the vocal tract using a cylindrical 3D DWG model. This mesh topology is derived from X-ray data of the vocal tract. The model has been used to measure and compare the frequency responses of three types of mechanical analogues. The first is an array of uniform quarter-wave cylindrical resonators, the second is a system of two concatenated cylindrical sections of different cross-sectional diameters, and the third is the vocal tract models themselves.

- (GULLY; DAFFERN; MURPHY, 2018) presented a model for the synthesis of dynamic speech sounds (diphthongs) using three-dimensional DWGs inspired by magnetic resonance data. This simulation model incorporated an admittance map into the production of these sounds. This model was compared to existing 2D and 3D DWM models for both monophthongs and diphthongs.

Table 1 summarizes he state of the art of the DWG and highlights some of the conditions of each of these papers.

Another computational model is called cellular automata. It consists of a dynamic system composed of a grid of cells, where each cell operates independently according to evolution rules based on the state of its neighborhood. This independence allows for parallel information processing throughout the cellular automaton. Consequently, this model can decompose complex problems into multiple simple subproblems that are solved locally and with parallel processing. This reduction optimizes computational resources and reduces processing time.

2.2 Cellular Automata

The Cellular Automata (CA) are computational models developed by John von Neumann in the late 1940s. The original idea was to propose systems that can self-reproduce to create complex systems. The development of this theory led him to contemplate concepts inherent to biology. This research culminated in the publication of the seminal paper "*Theory of Self-Reproducing Automata*" in 1966 by Arthur W. Bucks (NEUMANN; BURKS, 1966).

Table 1 – State of the art of the development of digital waveguides.

Authors	Year	Aplication	Topology	Dimensions
(MULLEN; HOWARD; MURPHY, 2006)	2006	Development a one and two-dimensional Kelly-Lochbaum vocal tract model using model using digital waveguides	Cylindrical	1D-2D
(MURPHY; BEESON, 2007)	2007	2D boundary termination using one-dimensional boundary nodes	Triangle	2D
(HACIHABIBOGLU; GUNEL; KONDOZ, 2008)	2008	Examine different DWG mesh topologies in two and three dimensions	Triangular, Rectilinear, Hexagonal, Cubic and Tetrahedral	2D-3D
(GERMAIN; EVANGELISTA, 2009)	2009	Model of the interaction of a guitar pick and the guitarist using DWG. Model used in the synthesis of guitar sounds	Lineal	1D
(EVANGELISTA; ECKERHOLM, 2010)	2010	Model of wave propagation in a guitar string. Model that also considers elastic and inelastic collision interactions to describe the most common phenomena associated with the behavior of these strings.	Lineal	1D
(HACIHABIBOGLU; GUNEL; CVETKOVIC, 2010)	2010	Simulation of directional microphones using DWG. Model tested in two- and three-dimensional mesh topologies	Cubic and Tetrahedral	2D-3D
(SPEED; MURPHY; HOWARD, 2013)	2013	A model of the vocal tract using a three-dimensional DWG based on X-ray data collected from the vocal tract. Three variations of the cylindrical topology were investigated.	Cylindrical	3D
(GULLY; DAFFERN; MURPHY, 2018)	2018	Vocal tract model using two- and three-dimensional DWG based on magnetic resonance imaging data of the vocal tract. Models used for synthesis of diphthong sounds	Cylindrical	2D-3D

Cellular Automata are discrete models consisting of an ordered lattice of elements (cells), each of which can be in a finite number of states. Figure 3 shows a schematic of a CA example. This system evolves in discrete time steps, where the states of the entire automata system are updated based on a set of predefined rules that take into account its current state and the state of its neighbors.

Due to the local nature of cellular automata, they become a feasible model for studying complex dynamical systems. Consequently, CA-based models are valuable in pure sciences such as physics (BRETON et al., 1995), biology (CHAUDHURI et al., 2018), computer science (D'AMBROSIO et al., 2018), medical applications (BRÜGGE et al., n.d.), cryptography (KUMARAVEL; MEETEI, 2013). A classic example of a CA model is the Conway's Game of Life, proposed by John Horton Conway in 1970 (GARDNER, 1970)(JOHNSTON; GREENE, n.d.).

Conway's Game of Life is a zero-player game whose evolution is determined by its initial state, from which the game evolves according to the rules of each cell. The rules are as follows:

- A living cell survives if it has two or three living neighbors.
- A new cell is born if it has three living neighbors.

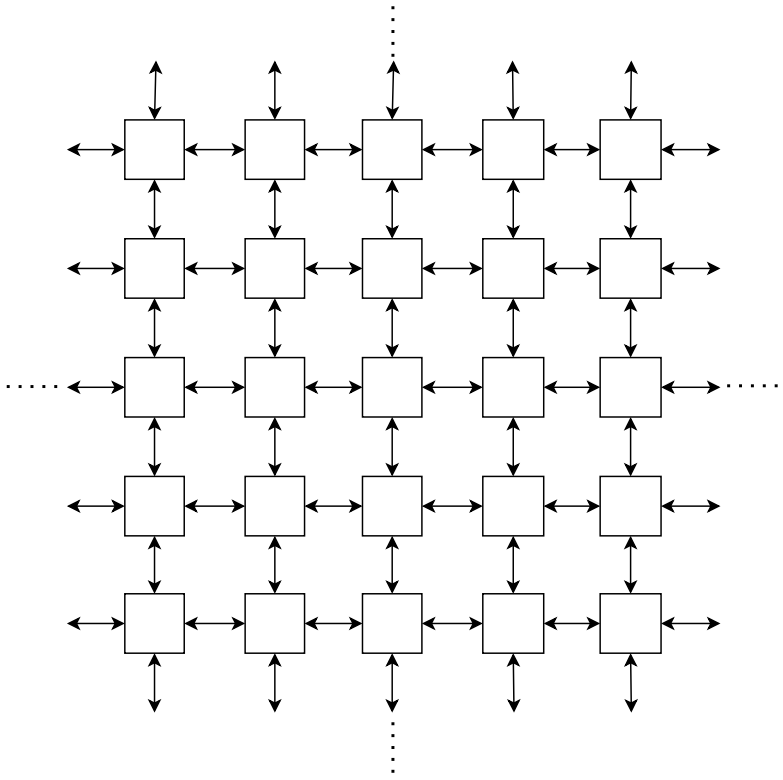
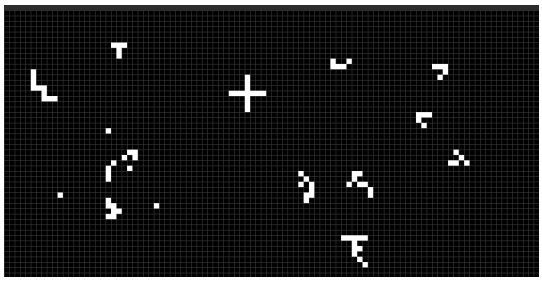


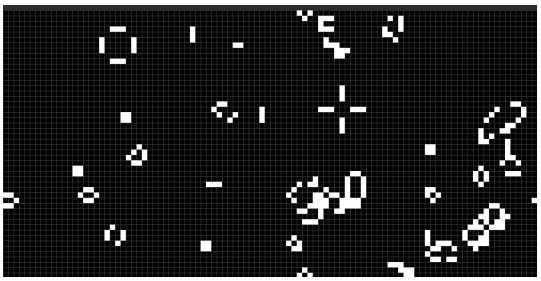
Figure 3 – A schematic representation of a Cellular Automata mesh.

- All other cells die or become inactive.

As shown in these rules, these automata have two possible states: *Alive* or *Dead*. Figure 4 shows an example of Conway's Game of Life, where the *Alive* state is represented in white and the *Dead* state is in black. Figure 4a is the initial state of the game and Figure 4b represents the final state. To see the complete evolution of this game readers are referred to the following video: `:https://youtu.be/PulNiRfykgo`



(a) Starting state of the Conway's Game of Life



(b) Final state of Conway's Game of Life

Figure 4 – Conway's Game of Life Example

Another famous CA was the one proposed by Wolfram in 1983 (WOLFRAM, n.d.) ,(WOLFRAM, 1984). This is a one-dimensional cellular automata with two possible states: zero or one. Its evolution in time depends on its states and the state of its right and left neighbors. This system has 256 possibles rules called "*Wolfram rules*" and they are usually

represented by a number between 0 and 255. Some of these rules can produce simple and predictable behaviors and others can produce complex and chaotic patterns. This automata begins with an initial cell in the state of 1 and the rest in 0 and then the system begins to evolve according to the established rules. An example of this is shown in the figure 5 using one of the most famous rules, rule 30.The investigation was summarized in "A New Kind of Science" (WOLFRAM, 2002).

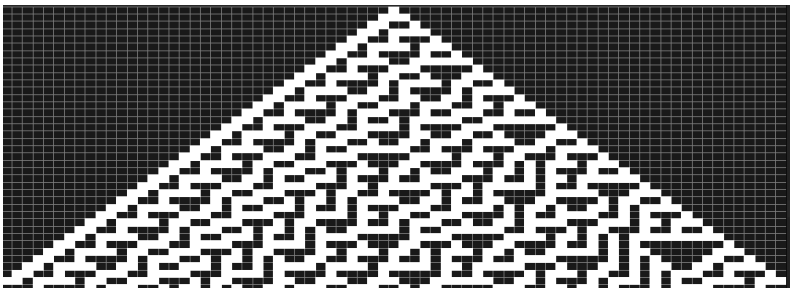


Figure 5 – A schematic representation of a Cellular Automata Mesh.

The FPGA has been included in studying cellular automata because these systems are typically used to perform specific computations in parallel. An example of this implementation is a study by (BAKHTERI; CHENG; SEMMELHACK, 2020) that implemented Conway's Game of Life on an FPGA. This study compared the performance of automation in a software application, a GPU, and the FPGA implementation and found that the FPGA was 36.7 times faster than the GPU and 2908 times faster than a common software applicatio showing the advantage of mixing FPGAs and cellular automata. FPGAs can be integrated together with a processor within the same platform, which is known as SoC-FPGA (System-on-Chip FPGA).

2.3 System on Chip FPGAs

A System on Chip (SoC) is an integrated device that combines all the necessary components of a computer (CPU, memory, I/O, peripheral controllers, etc.) on a single silicon chip.

On the other hand, an SoC-FPGA is known as a system that integrates an FPGA together with a SoC. In this type of system, the FPGA is typically programmed to perform specific tasks while the CPU handles general operations. In this way, the parallel processing offered by FPGAs is combined with the full functionality of an SoC. The figure 6 shows a schematic of the AMD Zynq 7000S SoC architecture on which some of the XILINX FPGAs such as the PYNQz2 are based.

The way SoC-FPGAs work is compatible with the parallel behavior characteristic of cellular automata, behavior in which it is necessary to update all the cells that make up the network simultaneously according to the rules with which they were programmed. In addition, SoC-FPGAs allow the combination of the programmed cellular automata with

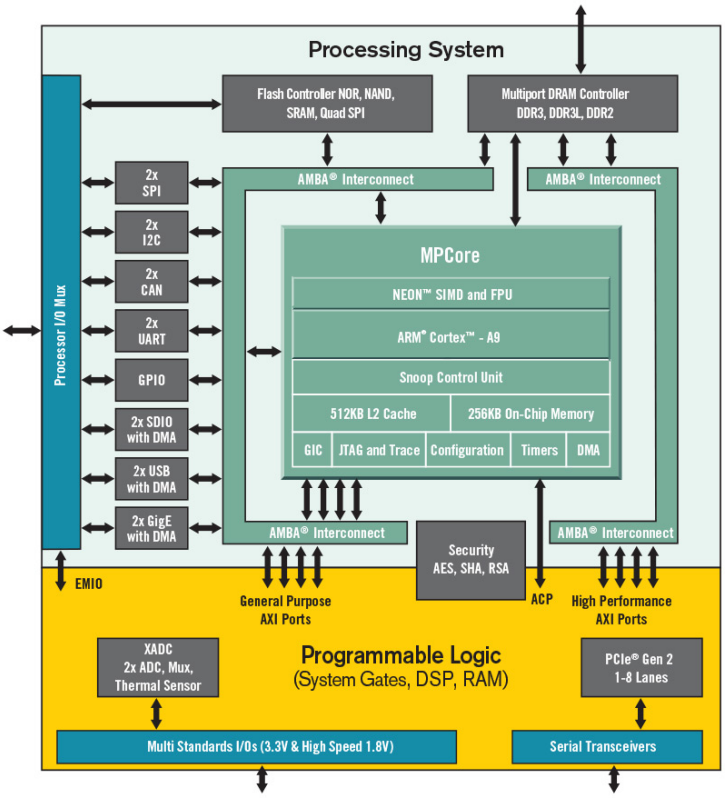


Figure 6 – A schematic of the AMD Zynq 7000S SoC architecture. (Ref:Xilinx)

other hardware and software components in the same system, allowing the development of complex systems that integrate multiple functionalities and capabilities. This type of solution is proposed to improve computational performance and reduce power consumption

In summary, the combination of cellular automata and SoC-FPGA makes it possible to take advantage of the parallel processing capacity and configuration flexibility of these devices to simulate complex dynamic systems. Below is a state of the art implementation of CA using these types of devices, whether GPUs or FPGAs.

2.4 Applications and Implementations of Cellular Automatas

To illustrate the versatility of cellular automata, the following are some works that use this computational model for different applications. In each case, the application for which cellular automata are developed is contextualized and the implementation platform is specified. The table 2 summarizes these examples.

- (YU et al., 2014) use the cellular automata to performed a human heart simulation using a GPU as hardware accelerator. The simulations were performed in a CPU with an Intel Dual Core i3-2100 CPU at 3.10 GHz with 16 GB of DDR3 memory was used and a graphics card Nvidia Telsa C2075 with 6 GB of global memory using a system

operation of Windows 7 64-bit. The GPU-assisted solution proved to be 30 times faster than the CPU-only solution.

- (TANASYUK; PEREPELITSYN; OSTAPOV, 2018) used one-dimensional cellular automata based on the 30, 86, 150 rules of Wolfram's cellular automata to implement cryptographic hash functions. The functions were implemented in software and hardware, and both were compared. These functions were implemented in FPGA (not specified) and in CPU Intel Core i5 4200U, 1.5 GHz and RAM 4GB.

- (NEDJAH; BEZERRA; MOURELLE, 2018) proposed a hardware architecture in which cellular automata use different neighborhoods, dimensionality, and rules to synthesize harmonious music. This research used a Virtex-5.

- (JIMÉNEZ-MORALES; GUISADO; GUERRA, 2018) proposes three variants of a model based on a cellular automata (CA) to simulate the behavior of general laser dynamics, pulsed pumped lasers, and lasers with antiphase dynamics. The simulations are computed on an unspecified CPU.

- (GUPTA; GÖZEN; TAYLOR, 2019) proposed a CA to simulate lipid membrane rupture processes in cells. The rules of CA behavior are proposed based on physical variables such as deformation, stress, and rupture. Simulations were performed on an iMac desktop computer using Matlab 2015b (MathWorks).

- (DOURVAS; SIRAKOULIS; ADAMATZKY, 2019) presents a cellular automata model to emulate the behavior of the slime mold Physarum Polycephalum. This model was tested on both an FPGA (Stratix V) and a GPU (unspecified). The FPGA implementation was approximately six times faster than the GPU solution. These studies demonstrate how FPGAs can enhance the performance of cellular automata.

- (SHAFIEI; KHAJI; ESKANDARI-GHADI, 2020)introduces an adaptive cellular automata method utilizing radial basis functions (RBFs) to simulate elastic wave propagation. This approach is divided into two main components. First, the problem is discretized using a triangular cellular automata mesh using a Vomnueman and Moore neighborhood, which is updated based on mass conservation and momentum balance principles, serving as a transfer function for mesh adjustment. Second, RBFs are applied to refine the mesh by identifying regions of redundancy or insufficiency of automatas. This iterative process continues until an optimal mesh configuration is achieved, ensuring the most accurate simulation results. Four different cases of wave propagation are analyzed and compared with other numerical methods such as FEM (finite elements) and SEM (spectral elements). The simulations are performed on a CPU with a 2 GHz T5870 processor.

- (LIN; ZHAO, 2020) proposes a cellular automata model based on variables such as force or stress to simulate seismic events. The simulations were performed on the TianHe-2 supercomputer at the China National Supercomputer Center in Guangzhou (32 Intel Xeon E5-2692 12C with 2,200 GHz 4,000 Xeon Phi 31S1P).

- (LUO; WANG; LEI, 2021) Used cellular automata to analyze the characteristics of sound radiation in structures of arbitrary shapes. First, considering the wave equation in spherical coordinates and using sound pressure as a variable, a 1D CA model is presented. This model has been used to analyze the superimposed sound field of multiple spherical sources in two-dimensional space (2D) and, in combination with the principle of wave superposition, the characteristics of the acoustic radiation of a rectangular piston in three-dimensional space (3D) .

- (RANGO et al., 2021) uses an extended cellular automata (XCA) to model subsurface flow processes (Alessio De Rango; Luca Furnari), especially in saturated flow. This research uses an Arch Linux CPU equipped with two Intel 8-core (16 threads) Xeon E5-2650 sockets and two Nvidia Titan Xp GPUs. As a result, the Nvidia Titan Xp achieves an average speedup of 19, while the CPU threads achieve an average speedup of 9. In addition, a dual Nvidia Titan Xp system achieves a speedup of 38.

- (MOURA; MUÑOZ, 2021) proposed a hardware architecture of a CA model to emulate the basic wave phenomena in one-dimensional systems and their interaction with obstacles. This model was developed and implemented on FPGA devices.

The adaptive approach proposed by (SHAFIEI; KHAJI; ESKANDARI-GHADI, 2020) attempts to improve the computational efficiency of elastic wave propagation simulation. However, the use of mass and momentum conservation principles to describe the propagation behavior involves matrix computations with sine functions, which increases the complexity of the operations. This problem becomes more pronounced when dealing with complex geometries. In addition, the process of updating the cellular automata (CA) and refining the radial basis functions (RBF) can result in extended computation times, with these times increasing as the size of the problem increases. The cellular automaton (CA) model developed by (MOURA; MUÑOZ, 2021) is restricted to the study of the propagation of one-dimensional waves. Furthermore, the size of the implemented system is constrained by the available hardware resources on the FPGA. (PEREIRA et al., 2023) addresses the issue of limited system size while still maintaining the restriction of unidimensional acoustic wave propagation. However, its solution necessitates a substantial number of registers and results in significant latency due to the need to configure the previous state of each block.

Table 2 – Related works regarding applications and hardware implementations of cellular automatas.

Authors	Year	Kind of solution	Dimension	Complexity	Platform
(YU et al., 2014)	2014	Human heart simulation using CA.	3D using a series of 2D images	Linear	GPU
(JIMÉNEZ-MORALES; GUISADO; GUERRA, 2018)	2018	Presentation of an alternative CA model applied to laser's dynamics.	2D	Non-linear	CPU
(TANASYUK; PEREPELITSYN; OSTAPOV, 2018)	2018	Cryptographic hash functions using cellular	1D	linear	GPU and FPGA
(GUPTA; GÖZEN; TAYLOR, 2019)	2019	CA system to emulate a ruptured biological diaphragm.	2D	Linear	CPU
(DOURVAS; SIRAKOULIS; ADAMATZKY, 2019)	2019	CA model with a multi-agent approach that mimics the behavior of the slime mold Physarum polycephalum.	2D	Lineal	FPGA and GPU
(SHAFIEI; KHAJI; ESKANDARI-GHADI, 2020)	2020	Adaptive CA system applied to sound wave propagation in an elastic lossless bi-dimensional case.	2D	Non-linear	CPU
(LIN; ZHAO, 2020)	2020	Seismic events modeling using CA systems.	2D	Non-Linear	Super-computer
(BAKHTERI; CHENG; SEMMELHACK, 2020)	2020	A SoC implementation of a CA system for the Conway's game of life.	2D	Linear	FPGA
(RANGO et al., 2021)	2021	Simulated subsurface flow process using CA systems.	2D	Linear	GPU
(LUO; WANG; LEI, 2021)	2021	A CA modeling applied to the emulation of structure-borne noises.	1D, 2D and 3D	Linear and Non-Linear cases	CPU
(MOURA; MUÑOZ, 2021)	2021	SoC-FGPA implementation of a CA model to emulate acoustic wave phenomenons in one-dimensional space.	1D	linear	FPGA
(PEREIRA et al., 2023)	2023	SoC-FGPA implementation of a staggered CA to emulate acoustic wave phenomenons in one-dimensional space.	1D	linear	FPGA

3 Methodology

In this work, the one-dimensional acoustic cellular automata model developed by Moura and Muñoz (MOURA, 2022) has been scaled to a two-dimensional model and embedded in a SoC-FPGA architecture. A three-step methodology was followed. The first step is the development of the CA2D model and its software implementation. This stage begins with a study of the acoustic wave propagation phenomenon using a DWG model, used to approximate the behavior of acoustic waves such as the study carried out by Moura in 2005 (MOURA, 2006), which is presented and complemented in this thesis. This study was subsequently used as a reference to formulate the CA2D model. The second step is the development of the hardware architecture of the three different cells that compose the CA2D model. Subsequently, an automatic VHDL code generator for the CA2D model was developed, allowing different mesh sizes to be implemented in a SoC FPGA device. Validation scenarios with barriers and walls were implemented. A 1 kHz sinusoidal signal and a G major guitar chord signal were used to excite the CA2D system and to evaluate its response using the Fourier Transform (FFT) and the Power Spectral Density (PSD). The third step is related to performance validation, in which the obtained digital circuits were characterized. The estimated execution time of the software implementation and the measured latency and throughput of the hardware implementation were used to estimate an acceleration factor. Figure 7 shows a flowchart of the implemented methodology.

3.1 Development tools and Hardware Setup

- **Reference models in software:** The DWG model and the CA2D model have been implemented using the PyCharm 2024.1.3 development environment in the Python 3.10 programming language.
- **Code Generator:** A VHDL code generator for the hardware implementation of the CA2D model was developed in GNU Octave 8.0.

  Both the reference models and the VHDL code generator tool run on an Intel Core i7 7700 at 3.60 GHz, with 16GB RAM, 916 GB HD, using Linux Ubuntu 20.04 operating system.
- **Hardware Implementation** The hardware implementation of the CA2D model was developed using the VHDL hardware description language, which is used to describe digital circuits and for electronic design automation. These hardware designs were synthesized and implemented in the Vivado Design Suite and the Software

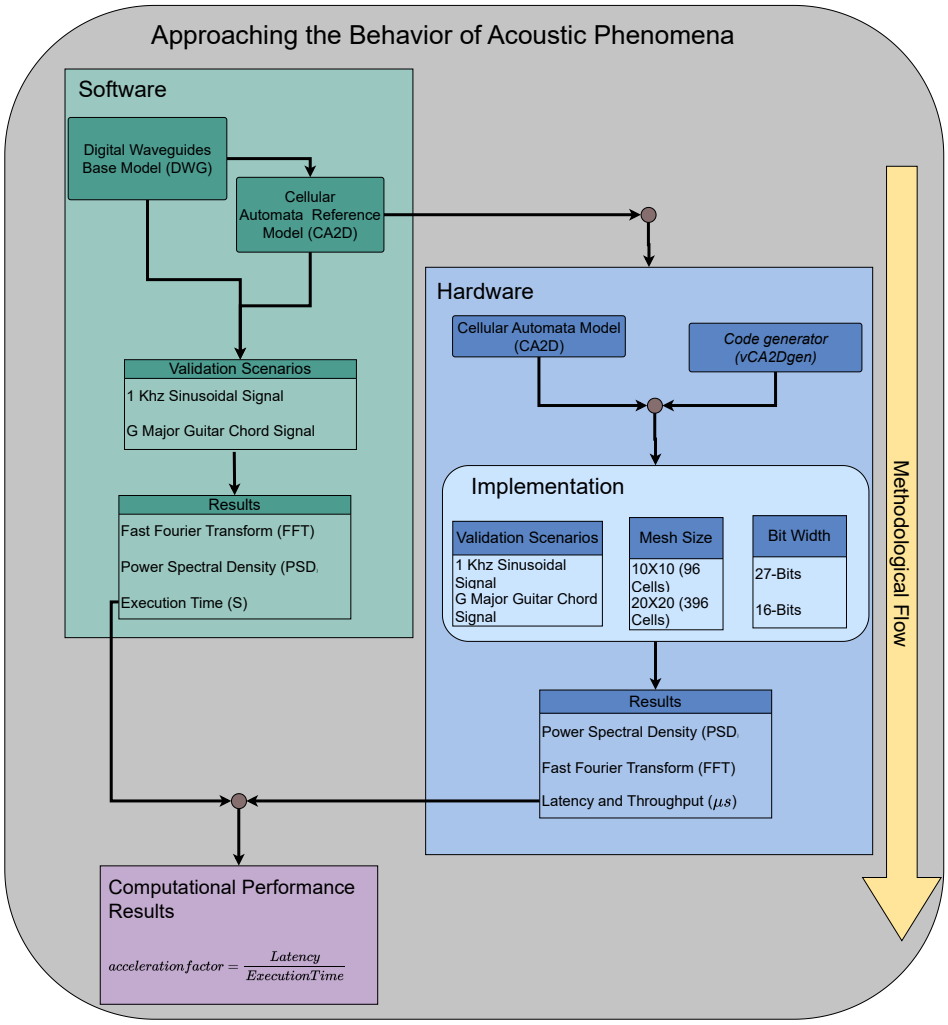


Figure 7 – Methodology flow chart.

Development Kit (SDK) version 2018.3 from AMD Xilinx. The SoC-FPGA development kits used in this work are presented below.

- PYNQ Z2: development kit based on the SoC Zynq-7020, which contains a dual-core ARM Cortex-A9 and an Artix7 FPGA. This development kit offers resources such as switches, LEDs, push buttons, Micro USB port, Arduino Shield connector, Raspberry Pi connector, USB (JTAG programming circuitry, OTG 2.0, UART bridge) a 10/100/1G Ethernet. Figure 8 depicts the board layout and all the ports and I/O pins. Table 3 summarizes the available resources offered in the Zynq-7020 SoC FPGA.
- ZCU104: This board uses a Zynq Ultrascale+ MPSoC equipped with a quad-core ARM Cortex-A53, a dual-core Cortex-R5 real-time processor, a Mali-400 MP2 graphics processing unit, 4KP60 capable H.264/H.265 video codec and a 16nm FinFET+ FPGA. This development kit offers resources such as switches, LEDs, push buttons, I2C bus, PMBUS, Micro USB slot, memory of 38Mb, USB UARTs with FT4232H JTAG/3xUART bridge, RJ-45 Ethernet connector. Figure 9

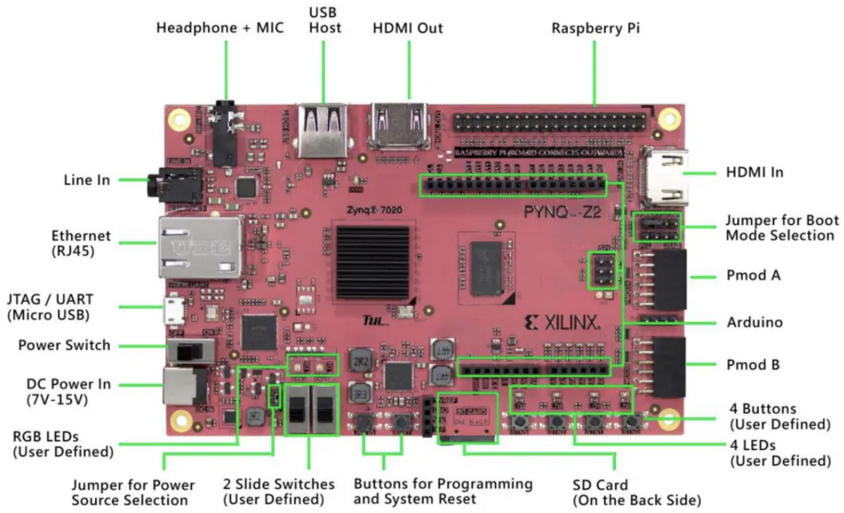


Figure 8 – Board layout of the PYNQ Z2 development kit.

SoC	Zynq-7000
Processor	Dual ARM Cortex A9
Flip-flops	106400
LUTs	53200
DSPs	220
BRAMS	4.9 MByte
I/O	68
DDR	512 Mbyte

Table 3 – Specifications of the SoC Zynq 7020 used in the PYNQ-Z2 board.

shows the board layout. Table 4 summarizes the available resources offered by the XCZU7EV MPSoC FPGA.

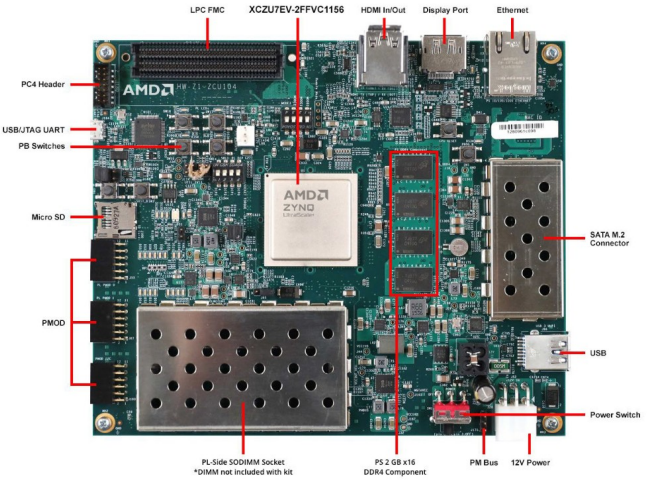


Figure 9 – Board layout of the ZCU104 development kit.

SoC	Zynq UltraScale+ MPSoC EV
Processor	quadcore ARM Cortex-A53 and dual-core Cortex R5
Flip-flops	460800
LUTs	230400
DSPs	1728
BRAMS	38 MByte
I/O	464
DDR	2GB

Table 4 – Specification of the Zynq Ultrascale+ MPSoC XCZU7EV.

3.2 Digital Waveguides based on the Impedance Method

For a better understanding of the behavior of an acoustic system, a comparison can be made with a mechanical spring-mass system, as shown in Figure 10, where the joint J is immersed in a system consisting of a group of flexible springs connected by digital waveguides, where the joint is influenced by the motion of the attached springs. In the acoustic system, the springs can be replaced by small volumes of air that undergo compression and expansion as a result of fluctuations in acoustic pressure at the dispersion joints.

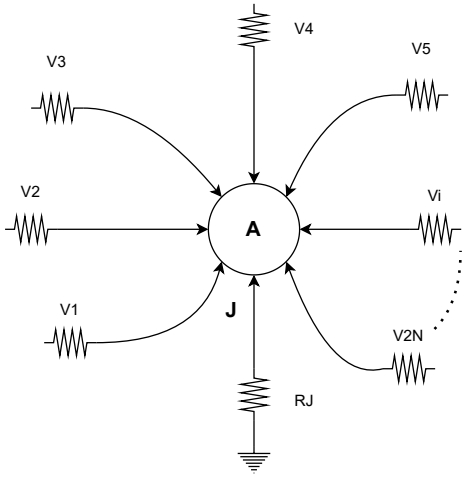


Figure 10 – Equivalent system of spring-mass for acoustic impedance response.

Equations 2.7 and 2.8 describing the force required to produce a transverse displacement at point J can be written as a function of the characteristic impedance R_J of the wave and the displacement velocity $\mathbf{V}_J$ at that point $\mathbf{J}$. If the joint is influenced by forces in more than one direction, as in Figure 11, the total force at this node can be obtained by summing the contributions from its neighboring nodes.

$$R_J \mathbf{V}_J = \sum_{i=1}^{2N} \mathbf{f}_i. \tag{3.1}$$

Using the relation shown in Equation 2.6 for f_i, the following expression can be derived,

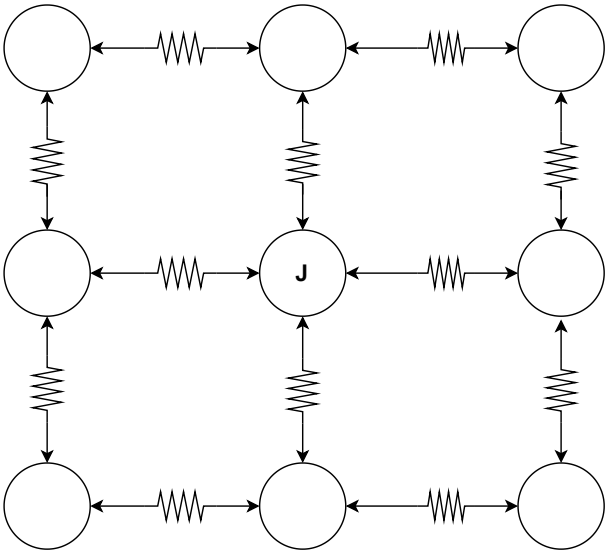


Figure 11 – Connection diagram between nodes using spring mass equivalence.

$$R_J \mathbf{V}_J = \sum_{i=1}^{2N} \left(\mathbf{f}_i^+ + \mathbf{f}_i^- \right), \tag{3.2}$$

$$R_J \mathbf{V}_J = \sum_{i=1}^{2N} \left(R_i \mathbf{V}_i^+ - R_i \mathbf{V}_i^- \right). \tag{3.3}$$

In the case of unidirectional equilibrium, the pressure in the right direction is equal to the pressure in the left direction. This means that $R_J \mathbf{V}_J = 0$. Hence,

$$0 = \sum_{i=1}^{2N} \left(R_i \mathbf{V}_i^+ - R_i \mathbf{V}_i^- \right). \tag{3.4}$$

This analysis is performed on a discrete system, where the velocity inside the spring can be expressed as the sum of the velocity moving to the right ($\mathbf{V}^+$) and the velocity moving to the left ($\mathbf{V}^-$). Then, replacing $\mathbf{V}_J = \mathbf{V}_i^+ + \mathbf{V}_i^-$ in Equation 3.4, it is obtained

$$0 = \sum_{i=1}^{2N} R_i \left(\mathbf{V}_i^+ + \mathbf{V}_i^+ - \mathbf{V}_J \right), \tag{3.5}$$

which can be expressed as,

$$\sum_{i=1}^{2N} R_i \mathbf{V}_J = \sum_{i=1}^{2N} 2 R_i \mathbf{V}_i^+, \tag{3.6}$$

allowing $\mathbf{V}_J$ to be expresses as a function of R_i and $\mathbf{V}_i^+$, such as

$$\mathbf{V}_J = \frac{2\sum_{i=1}^{2N} R_i \mathbf{V}_i^+}{\sum_{i=1}^{2N} R_i}. \tag{3.7}$$

This equation calculates the velocity at a scattering node of the spring mass system. Using the analogy between the spring-mass and an acoustic system, this equation can be rewritten to calculate the pressure at the scattering node J. Then the equation 3.7 is rewritten as

$$\mathbf{p}_J = \frac{\sum_{i=1}^{2N} 2R_i \mathbf{p}_i^+}{\sum_{i=1}^{2N} R_i}, \tag{3.8}$$

where $\mathbf{p}_J$ is the sound pressure at node J, $\mathbf{p}_i^+$ is the incident pressure and R_i is the characteristic impedance. The equation 3.8 can be rewritten as,

$$\mathbf{p}_J(n) = \sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n). \tag{3.9}$$

From this equation, it is possible to define a coefficient called the scattering coefficient a_i, whose value is determined by the impedance transitions along the mesh, taking into account the number of crossing waveguides N at the node. The mathematical expression for this coefficient is

$$a_i = \frac{2R_i}{\sum_{i=1}^{2N} R_i}, \tag{3.10}$$

where R_i is the waveguide impedance of the neighbor i to the junction $\mathbf{p}_J$, and $\sum_{i=1}^{2N} R_i$ is the sum of all wave impedances around the scattering junction $\mathbf{p}_J$. Using this method, an analytical expression for the scattering coefficients can also be obtained for a lossless medium, as follows.

3.2.1 The N-dimensional Digital Waveguide Mesh Formulation

Before starting the mathematical development, it is necessary to note that in the discrete context of digital waveguides shown in Figure 1, the input values at a scattering joint at time n are equal to the output values at the neighboring joints one time step back $n-1$. This relation can be expressed mathematically as

$$\mathbf{p}_J^+(n) = \mathbf{p}_i^-(n-1). \tag{3.11}$$

Having established this relationship between input and output in time steps n and $n-1$ for a scattering joint. The next step in the DWG formulation is to establish the pressure relation (Equation 3.9), but for a scattering joint in time step $n-1$. Such as,

$$\mathbf{p}_J(n-1) = \sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n-1). \tag{3.12}$$

From Equations 3.11 and 3.9 one can state that the pressure in the joint J at the instant $n+1$ will be the output pressure from the neighbor's joints at the instant n. This physical concept can be expressed as,

$$\mathbf{p}_J(n+1) = \sum_{i=1}^{2N} a_i \mathbf{p}_i^-(n). \tag{3.13}$$

Based on the D'Alembert principle, the force at a node can be described by Equation 2.6. This concept can be extended to the pressure in a digital waveguide according to (Mullen2006), deriving the next expression $\mathbf{p}_J(n) = \mathbf{p}_i^+(n) + \mathbf{p}_i^-(n)$, which is valid at any time ($n+1$, n, $n-1$,.., etc). This relation can be included in the Equation 3.12, then

$$\mathbf{p}_J(n-1) = \sum_{i=1}^{2N} a_i \left\{ \mathbf{p}_J(n-1) - \mathbf{p}_i^-(n-1) \right\}. \tag{3.14}$$

According to Equation 3.13, the term $\sum_{i=1}^{2N} a_i \mathbf{p}_i^-(n-1)$ is equal to $\mathbf{p}_J(n)$, which, according to Equation 3.12, corresponds to $\sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n)$, then,

$$\mathbf{p}_J(n-1) = \sum_{i=1}^{2N} a_i \mathbf{p}_J(n-1) - \sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n). \tag{3.15}$$

The next step is to eliminate the traveling waves of type $\mathbf{p}^{\pm}(n)$. This can be done by subtracting Eq. 3.13 from eq. 3.15.

$$\mathbf{p}_J(n-1) - \mathbf{p}_J(n+1) = \sum_{i=1}^{2N} a_i \mathbf{p}_J(n-1) - \sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n) - \sum_{i=1}^{2N} a_i \mathbf{p}_i^-(n). \tag{3.16}$$

Remembering the Eq. 2.6 derived from the equilibrium state of the mass-spring system, it is possible to conclude that the sound pressure at a joint can be obtained by the sum of two pressure parcels $p_i = p_i^+ + p_i^-$. This relation that can be used for all dispersion joints in the mesh at any instant of time. Using this relation it is possible to replace $\sum_{i=1}^{2N} a_i \mathbf{p}_i^+(n) + \sum_{i=1}^{2N} a_i \mathbf{p}_i^-(n)$ by $\sum_{i=1}^{2N} a_i \mathbf{p}_i(n)$, then

$$\mathbf{p}_J(n-1) - \mathbf{p}_J(n+1) = \sum_{i=1}^{2N} a_i \mathbf{p}_J(n-1) - \sum_{i=1}^{2N} a_i \mathbf{p}_i(n). \tag{3.17}$$

Isolating the term $\sum_{i=1}^{2N} a_i \mathbf{p}_i(n)$,

$$\sum_{i=1}^{2N} a_i \mathbf{p}_i(n) = \mathbf{p}_J(n+1) - \mathbf{p}_J(n-1) + \sum_{i=1}^{2N} a_i \mathbf{p}_J(n-1), \tag{3.18}$$

$$\sum_{i=1}^{2N} a_i \mathbf{p}_i(n) = \mathbf{p}_J(n+1) + \left\{ \sum_{i=1}^{2N} a_i - 1 \right\} \mathbf{p}_J(n-1), \tag{3.19}$$

the term $\mathbf{p}_J(n+1)$ can be isolated and adjusted to the time-step n to obtain the final scattering joint equation,

$$\mathbf{p}_J(n) = \sum_{i=1}^{2N} a_i \mathbf{p}_i(n-1) - \left\{ \sum_{i=1}^{2N} a_i - 1 \right\} \mathbf{p}_J(n-2). \tag{3.20}$$

This equation describes a model for an N-dimensional digital waveguide mesh, where the barriers or changes in the impedance within the mesh (boundary conditions) are introduced into the mesh by the scattering coefficients a_i, and acoustic sources can be applied to a particular scattering node.

Although the Equation 3.20 represents an N-dimensional model of acoustic wave propagation, it will be used in this thesis only for the two-dimensional case. The next section demonstrates the use of Equations 3.10 and 3.20 for a rectangular DWG mesh in a room acoustic application.

3.2.2 Square Waveguide Mesh and Impedance Method

To use the equation 3.20, firstly it is necessary to sum the four coefficients a_i around each scattering junction (four waveguides around a scattering junction inside the square mesh). The waveguide impedances are denoted by $R_{i,j}$, while the impedances to each scattering junction are denoted by $Z_J : R_{North}, R_{South}, R_{East}, R_{West}$. Figure 12 shows a schematic of this impedance configuration.

To use the DWG model for linear acoustic simulations, it is necessary to introduce some practical considerations. First, at each time step, denoted by n, the entire mesh must be updated using Eq. 3.20. Second, at each time step it is also necessary to simulate a delay line, which is a digital emulation of the medium representing the wave motion. This simulation model has a limitation in the sound velocity, which in the presented scheme should be constant over the entire mesh, which means that it can not change because of the interaction between two different media. To overcome this problem, some physical approximations can

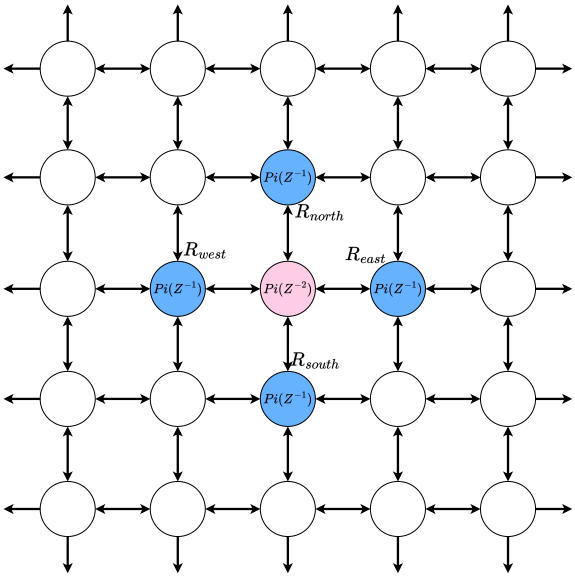


Figure 12 – A schematic representation of a square waveguide mesh. It can be thought of as an equidistant arrangement of nodes in the Cartesian domain.

be taken into account. For example, assuming that the barriers are thin enough to cover only a single point or a single line of scattering junctions, it is possible to create a mesh that has only one total impedance, Z_m, and several obstacle impedances, Z_{bi}. In this way, direct connections between two obstacles are eliminated. However, it is important to highlight that this numerical approximation is not a problem for linear acoustics.

This physical approximation limits the possible configurations for barriers in terms of the impedance values of the scattering junctions. Figure 13 shows three of the twelve possible scenarios where the physical impedance value at a scattering junction is different from the four impedances around it. In all of these scenarios, the cardinal notation is used for convenience. In the next, it is demonstrated how Equations 3.10 and 3.20 can be used for these three case studies. For the scattering coefficients computation, it is important to take into acoo

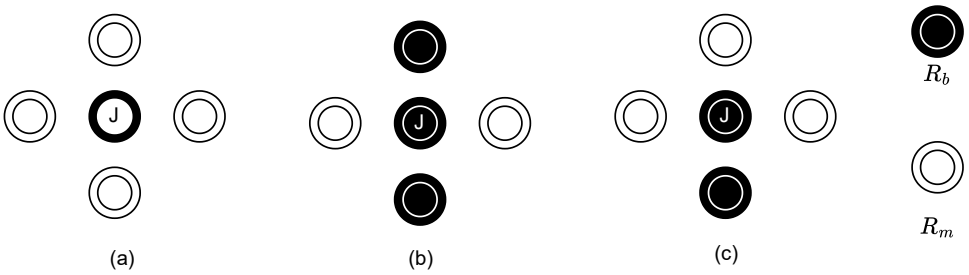


Figure 13 – Some possible impedance arrangements for a dispersion joint. The black dots represent impedances inside a barrier (R_b) while the white dots represent impedances outside the barrier (air: R_m).

3.2.2.1 Propagation in free-field environment

The first case chosen is the free field dispersion around a junction J, this means the junction is surrounded by equal impedances, as shown in Figure 13a. The scattering coefficient can be determined by applying Equation 3.10, where the numerator represents the waveguide impedance that directs the waves to the scattering junction J. The denominator is the sum of all waveguide impedances around the scattering junction J. In the case of the free field, the north, south, east, and west impedances are as follows,

$$a_N, a_S, a_W, a_E = \frac{2R_m}{R_m + R_m + R_m + R_m} = \frac{2R_m}{4R_m} = \frac{1}{2}. \tag{3.21}$$

This result shows that all scattering coefficients are equal to 1/2, since the calculations are always $2R_m$ divided by $4R_m$. Using these coefficient values in the equation 3.20 gives the following expression for the sound pressure,

$$\mathbf{p}_J(n) = \frac{1}{2}\mathbf{p}_i(n-1) - \mathbf{p}_J(n-2), \tag{3.22}$$

which describes the free-field propagation of acoustic waves. For visual validation, several small-scale simulations were conducted to verify the fidelity of the DWG model in replicating the behavior of acoustic waves. These simulations are presented in the following video : https://www.youtube.com/watch?v=Z3quu2eLJ34.

3.2.2.2 Propagation with a vertical complete barrier

The second case study is the case of normal incidence on an obstacle, as shown in figure 13b. Due to the impedance configuration, the equation 3.10 was used to calculate two coefficients. One for the east and west directions (outside the barrier) and one for the north and south directions (inside the barrier).

Considering the west or east positions, the Equation 3.10 is

$$a_W, a_E = \frac{2R_m}{R_m + 3R_b}, \tag{3.23}$$

while for the north and south directions (impedance of the central node J), the Equation 3.10 is

$$a_N, a_S = \frac{2R_b}{2R_m + 2R_b}. \tag{3.24}$$

In the case where $R_b >> R_m$, the scattering coefficients at the west and east tend to be zero $a_W, a_E \to 0$, indicating that no wave from normal incidence is transmitted to the barrier, resulting in perfect reflection. On the other hand, the scattering coefficients in the south and north tend to be unity $a_N, a_S \to 1$, indicating that traveling waves at the north are

perfectly transmitted only inside the barrier, not outside. Substituting these coefficients into the Equation 3.20 yields,

$$\mathbf{p}_J(n) = \mathbf{p}_N(n-1) + \mathbf{p}_S(n-1) - \mathbf{p}_J(n-2). \tag{3.25}$$

3.2.2.3 Propagation with a vertical incomplete barrier

The third case study is the barrier at the bottom. Only the impedance at the south is inside the barrier, as shown in Figure 13c. In this case, the Equation 3.10 was used to calculate two coefficients, one for the east, west, and north directions (outside the barrier), and one for the south direction (inside the barrier).

Considering the west, east, or north position, the Equation 3.10 is

$$a_W, a_E = \frac{2R_m}{2R_m + 2R_b}, a_N = \frac{2R_m}{3R_m + R_b}, \tag{3.26}$$

while for the south direction (impedance of the central node J.) the equation is

$$a_S = \frac{2R_b}{2R_m + 2R_b}. \tag{3.27}$$

Analyzing the same case where $R_b >> R_m$, the scattering coefficients in the west, east, and north tend to be zero $a_W, a_E, a_N \to 0$, while the scattering coefficients in the south tend to be unity $a_S \to 1$. Substituting these coefficients into the Equation 3.20 yields,

$$\mathbf{p}_J(n) = \mathbf{p}_S(n-1). \tag{3.28}$$

3.2.3 DWG Validation Scenarios

To validate the formulated model for the DWG, the scenario depicted in Figure 14 was designed. This scenario is composed of a total of 3844 dispersion joints arranged in a rectangular room with a medium impedance of $Z_m = 400$ (shown as blue joints) and a high impedance barrier of $Z_{bi} = 1 \times 10^{40}$ (shown as orange joints), where two receptors were placed at positions (1.0, 2.5) and (3.5, 2.5) and the source is the center at position (1.0, 2.5). The update rate was set to 6 KHz.

Since this research focuses on CA2D for simulations in acoustic rooms, where the study of sound propagation is important, two audio signals will be used to validate this behavior: a 1 KHz sinusoidal signal and a G major guitar chord signal sampled at 6 KHz. The choice of these low-frequency signals is due to the fact that these numerical methods based on acoustic wave propagation are able to accurately simulate these phenomena in this low-frequency range (THOMAS, 2016), but for signals with high sampling frequencies the computational cost of performing the simulation is increased (SOUTHERN et al., 2011). For both cases, the simulation was run for one second, resulting in 6000 time-frames of simulation data collected from both receptors.

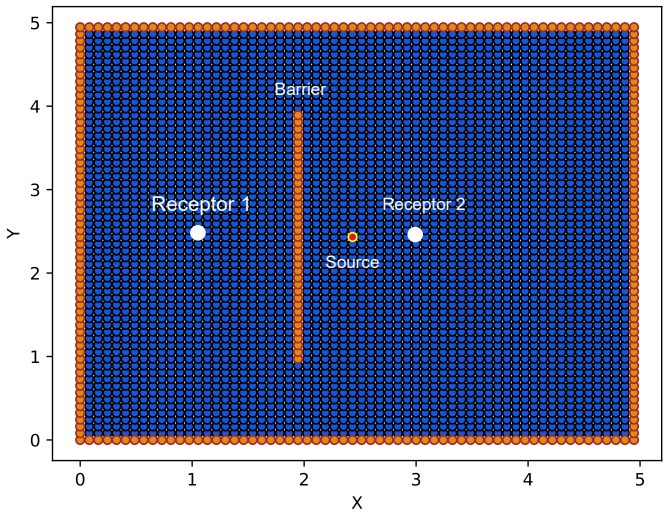


Figure 14 – Schematic of a real-world scenario with a square waveguide mesh and a barrier placed inside it. The walls of the waveguide also act as barriers with the same impedance. Additionally, two receptors and a sound source were included for certain simulations.

This digital waveguide model is the starting point for the development of the cellular automata model. In the next section, the Huygens-Fresne principle is applied to develop the two-dimensional acoustic cellular automata (CA2D) model.

3.3 Modeling Acoustic Phenomena using Cellular Automata

The Huygens-Fresnel principle was used to develop the cellular automata organs (MILLER, 1991). This principle states that each point on a wavefront is itself a source of spherical wavefronts and that the secondary wavefronts emanating from different points interfere with each other. A diagram of this principle is shown in Figure 15.

To solve the problem of the differential equation with its boundary conditions (it becomes numerically impossible to find a solution using analytical solutions), previous works proposed a discrete Huygens-Fresnel approach to describe the propagation of acoustic waves (Y. KAGAWA T. TSUCHIYA; TAKEUCHIT, 1998), (Y. KAGAWA T. TSUCHIYA; TAKEUCHIS, 1999). In a bi-dimensional case, this approach considers a set of nodes separated by a certain periodic distance Δl, ordered by Cartesian coordinates, as depicted in Figure 16. Acoustic propagation occurs between isolated nodes in which an incident disturbance is transmitted to the four neighbors surrounding it. This propagation is characterized by half of the incident energy and an opposite magnitude in the direction of reflection.

In this work, this discrete approach to the Huygens-Fresnel principle was used to develop a cellular automata organism. This cellular automata consists of three kinds of cells:

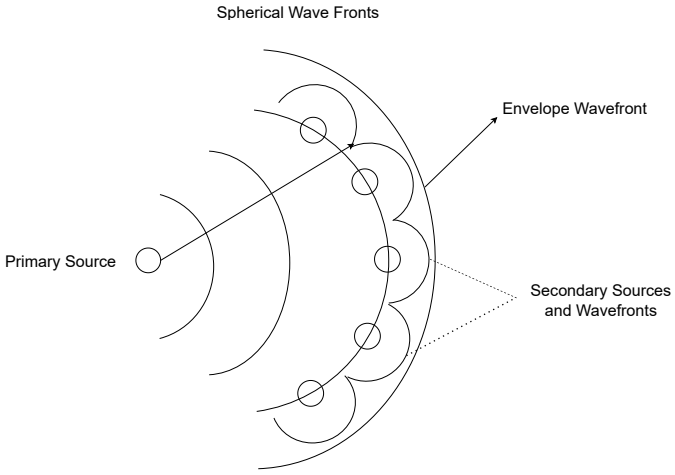


Figure 15 – Diagram of the Huygens-Fresnel principle. A spherical source produces spherical wavefronts that excite adjacent points which, once excited, behave like other sources creating new spherical wavefronts. The propagating wave is the product of the envelopes of these new wavefronts.

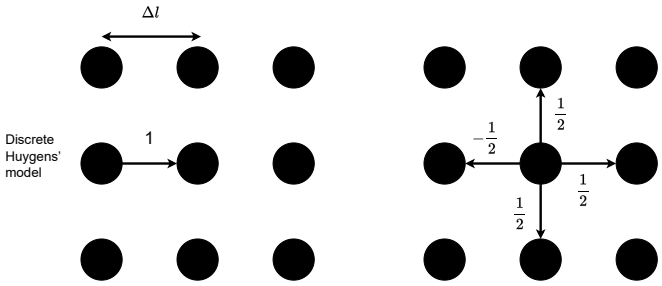


Figure 16 – Scheme of the discrete propagation of waves according to the Huygens-Fresnel principle proposed by (Y. KAGAWA T. TSUCHIYA; TAKEUCHIT, 1998)

(a) a cell for the walls in the environment space (called wall cells), (b) cells inside the space (called mesh cells), and (c) a source cell where the inputs are injected into the organism. The proposed cellular automata organism considers a Von Neumann neighborhood as shown in Figure 17. In a two-dimensional case, it means that the source cells and mesh cells consider the north, south, east, and west directions for their inputs and outputs (four inputs and four outputs), while the wall cells have only one input and one output. In the latter case, the direction depends on the position of the cell. In this way, the sound wave is reflected in the opposite direction of the incident wave.

The connections and information flow of the three types of cells are shown in Figure 18. The mesh can be thought of as a set of elements ordered in space (a matrix), where each element inside this matrix is a mesh cell. Figure 19 shows an example with a 5×5 cellular automata mesh. In this example, the source is placed at the center of the network (position 3,3), the first and last columns and rows are wall cells, and the rest are mesh cells. The scheme exemplifies the connections and information flow between cells within the mesh. The details of the hardware architecture and the behavior of these three cells are described below.

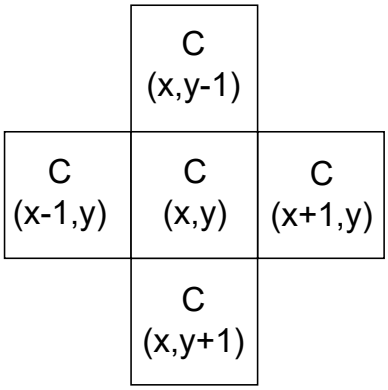
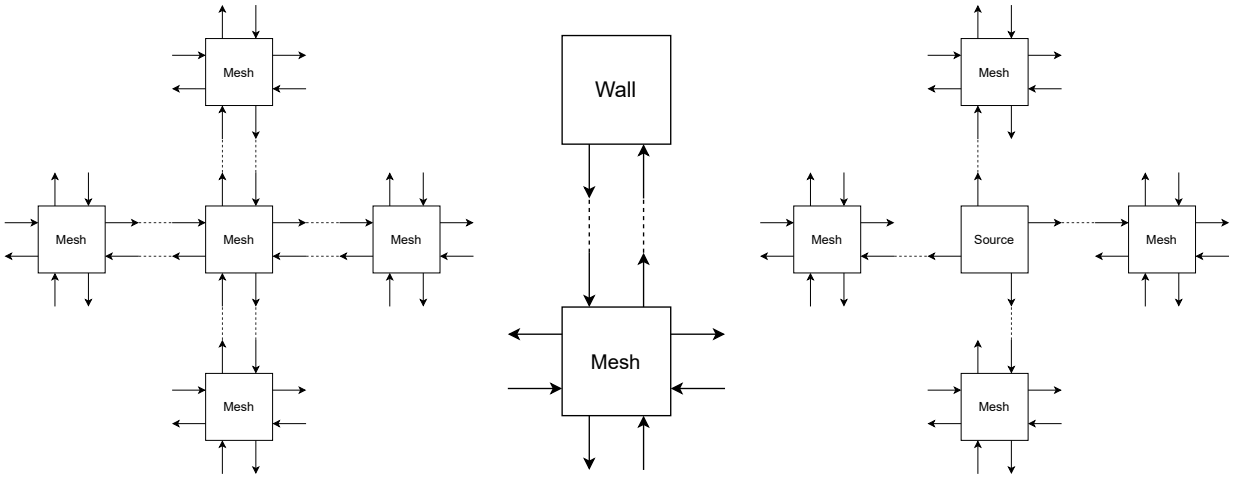


Figure 17 – Scheme of the Von Neumann neighborhood.

It is important to highlight that all the digital circuits were implemented using floating-point numerical representation, taking advantage of previously implemented IP cores for arithmetic operations (MUÑOZ et al., 2010b) (MUÑOZ et al., 2010a)[1]. These IPs can be customized in terms of bit-width. In particular, two different representations were used. The first one, uses a 27-bits representation (1 bit for sign, 8 bits for the exponent word, and 18 bits for the mantissa word). This is done because some Xilinx FPGAs devices make use of 18 × 18-bits DSP (Digital Signal Processing blocks). The second one, uses a 16-bits representation (1 signal bit, 8 bits for the exponent, and 7 bits for the mantissa), aiming to minimize the hardware consumption.



(a) Mesh connections (b) Wall connections (c) Source connections

Figure 18 – Neighborhood connections and information flow in the three cells.

3.3.1 Wall Cell

Figure 20 shows a schematic of the wall cell. This type of cell has four inputs and two outputs. Inputs: *clock* (clk), *reset*, *start*, P_{in} (input pressure). Outputs: ready, P_{out} (output pressure).

[1] These IP cores do not have a rounding technique.

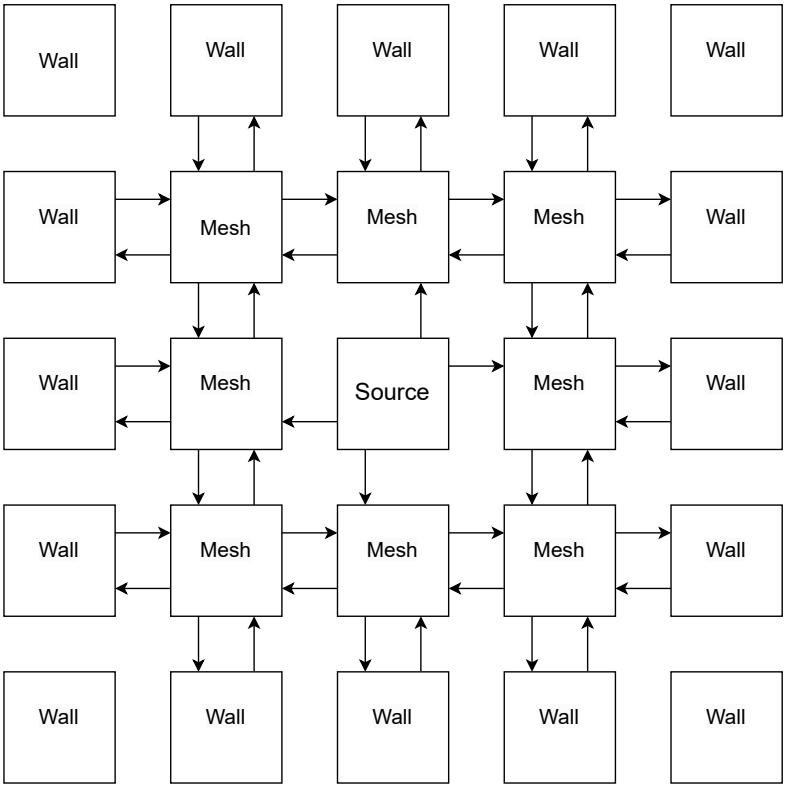


Figure 19 – 5×5 mesh showing the concessions for each type of cell and how the information is transmitted within the mesh.

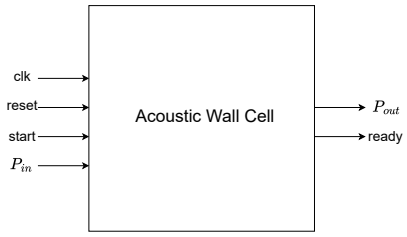


Figure 20 – VHDL entity for the acoustic wall cell.

This cell determines the behavior of the walls in a cellular automata organism, i.e. they define the boundary conditions of the network. In this work, the wall cell was designed to emulate perfectly reflective boundary conditions, which means all sound pressure acting on the walls is reflected, but the magnitude and its direction are inverted. Mathematically, the behavior of this cell is given by the Equation 3.29.

$$P_{out} = -P_{in} \tag{3.29}$$

The wall cell was implemented in hardware as a sequential circuit, waiting for the start input and inverting the signal bit of the P_{in} input and copying it to the P_{out} output. Thus,

this cell has a latency of 1 clock cycle. Figure 21 shows a testbench of the behavior of this cell for a 27-bit floating-point numeric representation.

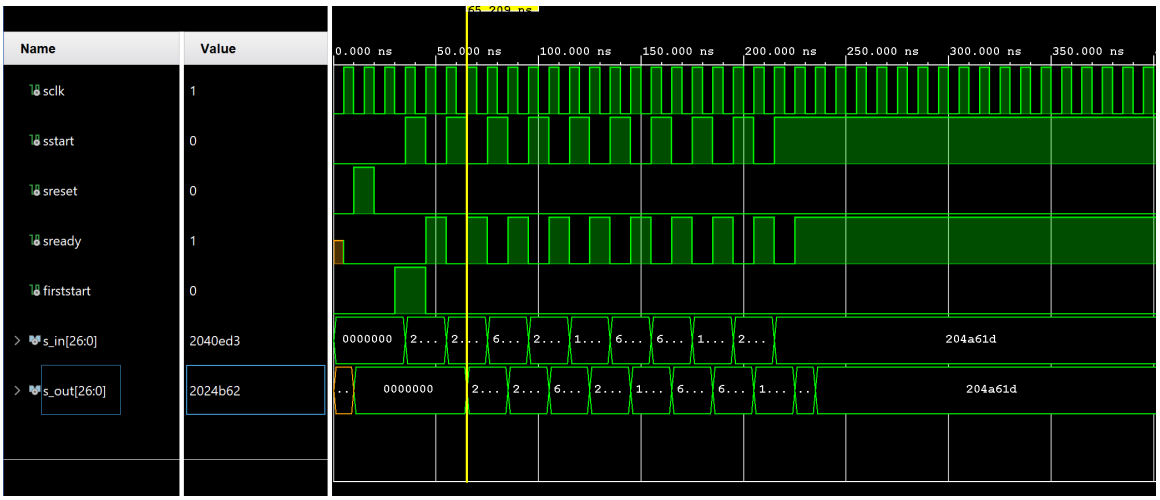


Figure 21 – Simulation results for the Wall Cell.

3.3.2 Mesh Cell

This cell is used to represent the cells used to propagate the sound pressure along the mesh. Figure 22 shows a schematic of this cell, which has eight inputs and five outputs. Inputs: *clock* (clk), *reset*, *start*, *mode*, Pn_{in}, Ps_{in}, Pe_{in}, Pw_{in}. Outputs: *ready*, Pn_{out}, Ps_{out}, Pe_{out}, Pw_{out}.

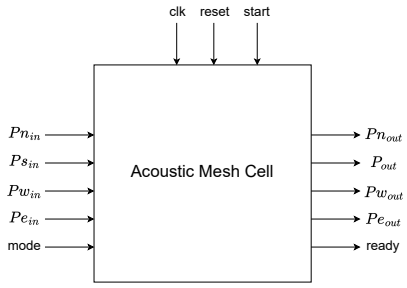


Figure 22 – VHDL Entity of the acoustic mesh cell.

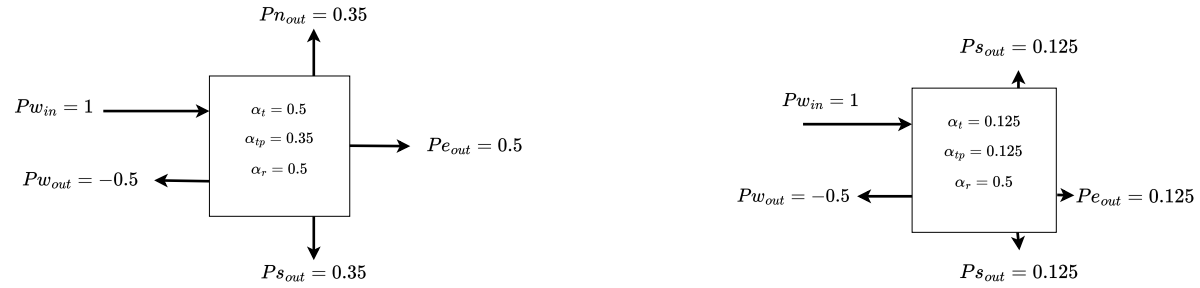
According to the discrete Huygens-Fresnel principle, all mesh cells behave as sources as soon as one of their inputs has a pressure value. Then, if a mesh cell receives an stimulus different from zero in one of its four inputs, the cell computes the four expressions in Equation 3.30 and attributes them to the outputs. In this way, the mesh cell will be a new source to its neighbors at the next time step.

$$
\begin{aligned}
Pn_{out} &= -\alpha_r Pn_{in} + \alpha_t Ps_{in} + \alpha_{tp}(Pe_{in} + Pw_{in}) \\
Ps_{out} &= -\alpha_r Ps_{in} + \alpha_t Pn_{in} + \alpha_{tp}(Pe_{in} + Pw_{in}) \\
Pe_{out} &= -\alpha_r Pe_{in} + \alpha_t Pw_{in} + \alpha_{tp}(Ps_{in} + Pn_{in}) \\
Pw_{out} &= -\alpha_r Pw_{in} + \alpha_t Pe_{in} + \alpha_{tp}(Ps_{in} + Pn_{in})
\end{aligned}
\tag{3.30}
$$

It can be observed that the neighborhood inputs are related to the cell outputs using linear acoustics, where α_r and α_t are the reflection and transmission coefficients, respectively. These coefficients characterize the transmission and reflection behavior of the cells. The reflection is the information that is returned in the input direction, while the transmission is the information that is transmitted to the remaining three outputs. On the other hand, the α_{tp} coefficient represents a proportional transmission coefficient used to control the amount of energy transmitted in the non-preferential direction, i.e. the direction in which the incoming wave does not strike, preventing the cellular automata from diverging over time.

It is important to note that the values of α_r, α_t, and α_{tp} depend on the type of behavior desired for the mesh cell. In this CA model, mesh cells have two behaviors: mesh mode and obstacle mode. In mesh mode, the value of the coefficients α_r and α_t are set to 0.5, values consistent with the discrete Huygens-Fresnel model proposed by Kawaga (Y. KAGAWA T. TSUCHIYA; TAKEUCHIT, 1998). On the other hand, the value of α_{tp} is usually set to a lower value than 0.5. This set of values causes the mesh cell to transmit half of the acoustic pressure it receives in the direction of wave incidence to its neighbors, while the percentage of acoustic pressure in the non-preferred direction will be lower. In the obstacle mode, $\alpha_r = 0.5$, while α_t and α_{tp} are set to a very small number, i.e. in the direction in which the wave hits the obstacle, the reflected wave will be half the incident pressure, while in the remaining three directions this output will be approximately zero. It should be highlighted that the choice of values for α_r, α_t and α_{tp} in both the mesh and obstacle modes was made according to the calculation performed in the digital waveguide model by the impedance method (see section 3.2).

Figure 23 shows an example of the behavior of the mesh cell for both modes. The mesh mode has coefficients α_r, α_t equal to 0.5 and α_{tp} equal to 0.35, while the obstacle mode has α_r equal to 0.5 and α_t, α_{tp} equal to 0.125. In this example, an incident wave of unit magnitude interacts with the west input of the mesh cell. For the mesh mode (Figure 23a), the cell transmits half (0.5) of the input pressure to the Pe_{out}, Pw_{out} outputs, corresponding to the direction of incidence of the wave and 0.35 to the Pn_{out}, Ps_{out} outputs corresponding to the non-preferential direction of propagation, it can be seen that Pw_{out} has a negative magnitude. For the obstacle mode (see Figure 23b) only the west output (Pw_{out}) is half the value of the input value and the remaining three outputs ($Pn_{out}, Ps_{out}, Pe_{out}$) are one-eighth of the input (0.125).



(a) Mesh cell behavior in mesh mode

(b) Mesh cell behavior in obstacle mode

Figure 23 – Scheme of the choice of the value of the coefficients and how it affects the behavior of the mesh cell in the transmission of the incident wave

In the hardware architecture, the cell selects the respective reflection, transmission, and proportional transmission coefficients (α_r, α_t, and α_{tp}, respectively) according to the *mode* input. A Finite State Machine (FSM), depicted in Figure 24, was used to implement in hardware the four expressions of Equation 3.30.

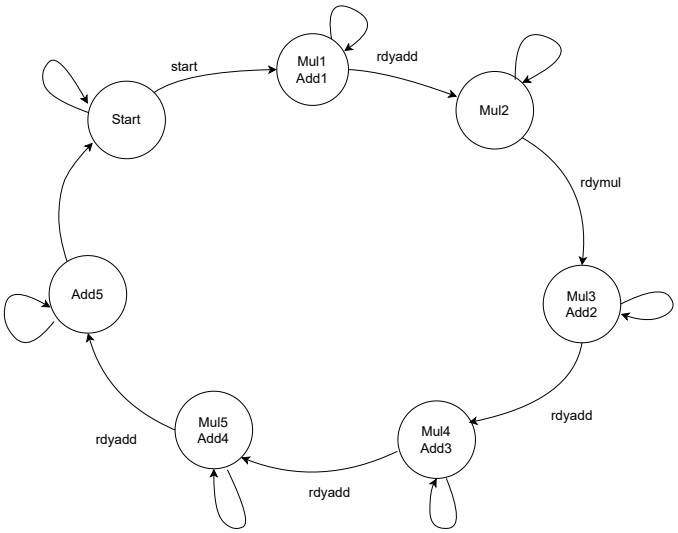


Figure 24 – Scheme of the finite state machine implemented for the operation of the mesh cell. Each state shows the corresponding stage for computing the Equations 3.30.

This FSM controls two parallel multipliers and two parallel adders, one for each propagation direction (north-south or east-west). Those operators are reused at different times to perform the calculations in both directions simultaneously. Figure 25 shows the different stages performed by the FSM, which are also explained below.

- Start: This is the start phase of the FSM. Without outputs. Change the estate with the *start* signal of the Source.
- Mul1 Add1: The steps marked with the circumscribed 1 in Figure 25 are performed. Without outputs. Change the estate with the *ready* signal of the adder.

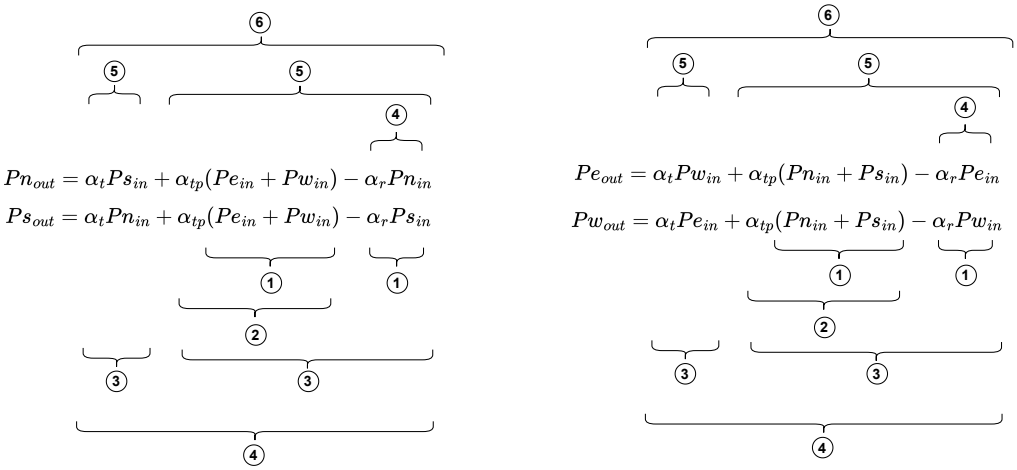


Figure 25 – Equations implemented by the mesh cell. The stages in which these equations are implemented by the finite state machine 24 are shown.

- Mul2: The steps marked with the circumscribed 2 in Figure 25 are performed. Without outputs. Change the estate with the *ready* signal of the adder.
- Mul3 Add2: The steps marked with the circumscribed 3 in Figure 25 are performed. Without outputs. Change the estate with the *ready* signal of the adder.
- Mul4 Add3: The steps marked with the circumscribed 4 in Figure 25 are performed. Outputs: South and East pressure. Change the estate with the *ready* signal of the adder.
- Mul5 Add4: The steps labeled with the circumscribed 5 in Figure 25 are performed. Without outputs.
- Add5: The steps labeled with the circumscribed 6 in Figure 25 are performed. Outputs: North and West pressure.

This cell starts its calculations by asserting the *start* port and when the calculations of the four outputs are finished, the cell asserts the *ready* signal. The latency for this cell is nineteen clock cycles, as demonstrated by the behavioral simulation depicted in Figure 26, in which a 27-bits floating-point numerical representation was used.

3.3.3 Source Cell

Figure 27 shows the schematic of this cell. Inputs: *clock* (clk), *reset*, *start*, *source*, *medium*. Outputs: *ready*, Pn_{out}, Ps_{out}, Pe_{out}, Pw_{out}.

This cell injects the sound pressure values into the mesh. To do this, this cell uses the *source* input port, which reads the sound pressure values to be injected into the cellular automata organism. This value is routed to the North (Pn_{out}), South (Ps_{out}), East (Pe_{out})

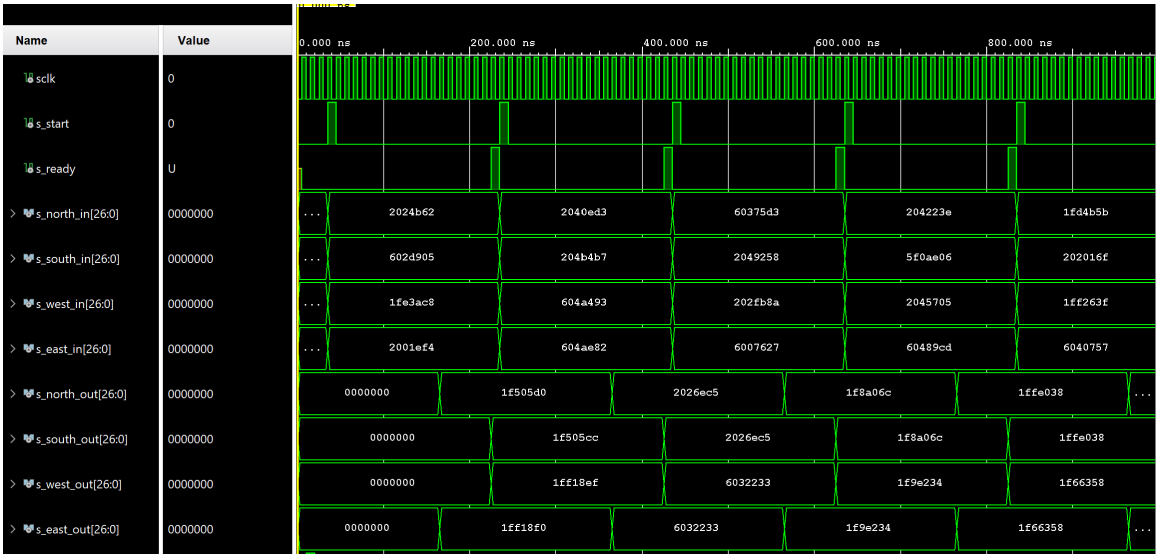


Figure 26 – Behavioral simulation results for the mesh cell.

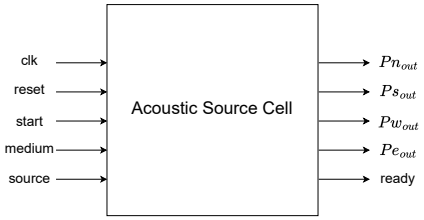


Figure 27 – VHDL Entity of the acoustic Source cell.

and West (Pw_{out}). These four outputs are assumed to respect the proposed von Neumann neighborhood.

In addition, this cell acts as the *heart* of the cellular automata organism, controlling the timing behavior of the CA. Thus, all other cells (the wall cell and the mesh cell) are synchronized by the *ready* signal of the source cell.

In order to respect the latency of the mesh cells, the source cell uses the *medium* input port. This input selects a preset that is used by a synchronous counter to respect the latency of nineteen clock cycles. When this counter reaches the preset value, the source cell sends new data to the cellular automata. It should be notice that by modifying this time delay it is possible to emulate propagation media with different sound velocities. However, this aspect is out the scope of this work.

In the same way as the previous cells, the *start* port starts the operation of this cell and the output is valid by asserting the *ready* signal. Figure 28 shows a behavioral simulation of this cell.

It is important to highlight that, through the behavior of these three cell types, the 2D Cellular Automata (CA2D) model adheres to both the Huygens-Fresnel principle and linear acoustic phenomena. The source cell and the mesh cell in mesh mode emulate reflection

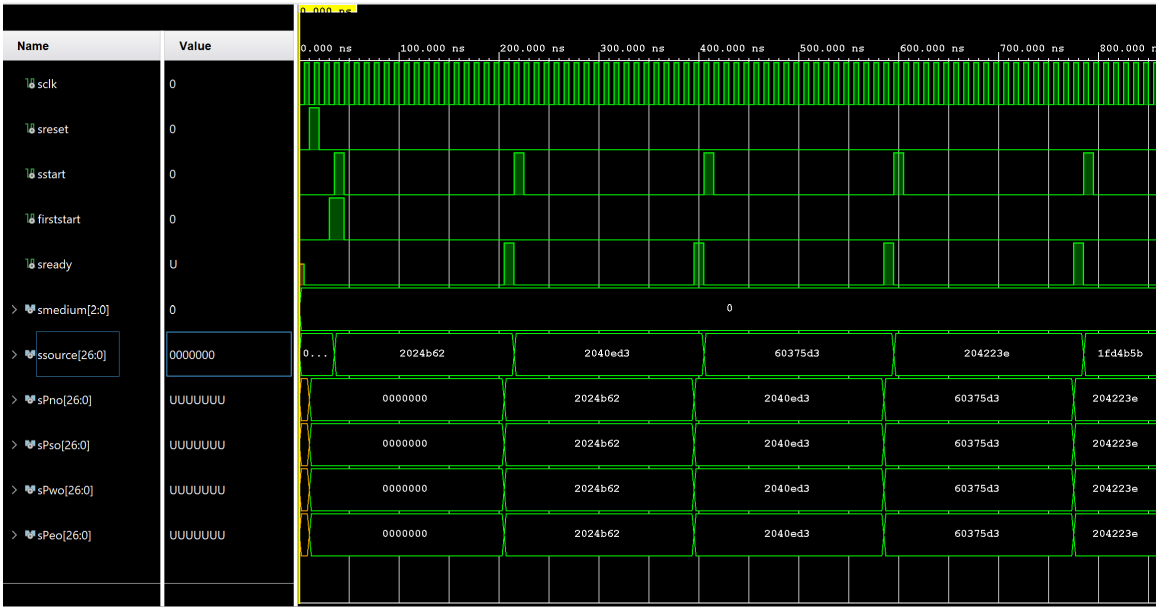


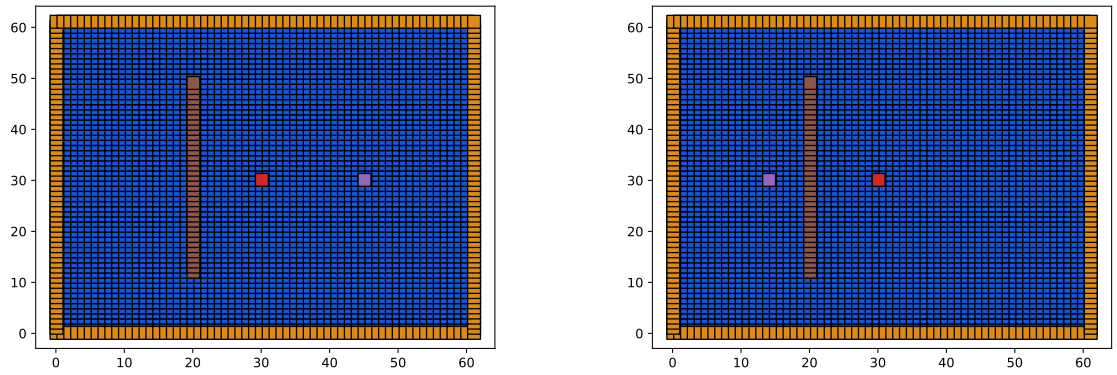
Figure 28 – Simulation results for the source cell.

and transmission phenomena. The reflection phenomena caused by obstacles and walls are captured by the mesh cell in Obstacle mode and the wall cell, respectively. Importantly, none of these cells store pressure values or perform calculations beyond those defined in Equations 3.29 and 3.30, thereby restricting the modeled phenomena to those directly affecting the propagation of acoustic waves.

3.3.4 CA Validation Scenarios

The same scenario used to validate the DWG model was employed to validate the CA model. This CA representation was implemented in Python using a list to store the pressure output values. Due to the use of a Python list, the CA is limited to a single receptor. This limitation arises from the programming scheme used to update the state of all cells, which evaluates the type of cell within the mesh by a *tag*. Depending on this tag, the cell behavior (source, mesh, or wall) is determined. Consequently, the output list cannot differentiate between receivers that store the output data.

Figure 29 shows two different cases for this scenario. The walls with a high reflection condition are shown in orange, the obstacles in brown, the source in red, the mesh cells in blue, and the receiver in purple. The receiver was placed at positions (14,30) and (45,30) for the first and second cases, respectively. For both cases, the source is at the center of the mesh (30,30). The refresh rate of the environment was set to 6 KHz.



(a) FFT Spectrum Receptor 2

(b) FFT Spectrum Receptor 1

Figure 29 – Frequency response of receivers 1 and 2 for a sinusoidal signal. Left: Frequency response in Receptor 2 (before the barrier); Right: Frequency response receptor in Receptor 1 (after the barrier).

3.4 Automatic VHDL Code Generator for the CA2D Model

Writing the VHDL code for a CA mesh is not straightforward, because the connections need a certain order to achieve an optimal connection within the mesh. For example, the 5×5 mesh shown in Figure 19 needs forty-four signals for the internal connection. This number of signals only increases as the mesh size increases. To solve this problem and to speed up the process of writing VHDL code, a code generator tool, called *vCA2Dgen*, was developed in Octave.

The *vCA2Dgen* tool enables users to create meshes of various sizes, allowing adjustments to the positions of the receiver and source, as well as the creation of both vertical and horizontal obstacles with customizable lengths. To generate the VHDL code, the *vCA2Dgen* tool requires the input of the mesh dimensions (height and width) and the positions of the source and receiver. To incorporate obstacles into the mesh, it is necessary to specify the initial position (height and width) and the length of the obstacle. Currently, the *vCA2Dgen* tool is limited to supporting a single source and a single receiver.

3.5 Validation proposal for the FPGA Implementations

Two different mesh sizes were used for validation purposes: 10×10, 20×20 using the Pynq Z2 and the ZCU104 development boards for 16-bits and 27-bits floating-point representations.

3.5.1 Validation using Behavioral Simulation

Several testbenches were created for all the possible configurations. Each testbench automatically reads the input from text files and writes the outputs to text files that can be decoded and interpreted in the PC.

This organization results will be also presented for validating the physical implementation of the CA2D model using the hardware-in-the-loop scheme described in the next section.

3.5.2 Hardware-in-the-Loop for Acoustic Simulation

Hardware-in-the-Loop (HIL) is a technique used in embedded system development to simulate the environment in which the embedded system will operate. This allows hardware and software to be validated and tested in conditions very close to those in which the system is designed to operate, prior to final implementation.

To validate the proposed hardware architecture of the CA2D model, a hardware-in-the-loop environment was developed. In this HIL environment, a Communication Block IP Core (Core ComBlock) was used. This IP was created between the Multidisciplinary LABoratory (MLAB) of the Abdus Salam International Center for Theoretical Physics (ICTP, Italy) and the FPGA Department of Micro and Nanotechnology of the National Institute of Industrial Technology (CRESPO et al., 2021). The ComBlock IP was designed to solve the high-speed acquisition and processing in the FPGA and send the resulting data to a PC. In this communication the ARM processor of the SoCs Zynq7020 and Zynq UltraScale+ serves as a provider of data storage (DDR memory) and Ethernet connection.

The ComBlock provides several interfaces such as registers, RAM, and FIFOs, allowing the co-processors to be connected to the Processor System (PS) through the AXI on-chip protocol (MELO, 2019). This communication scheme allows the use of scripts in Python to send and receive data. These communication interfaces are:

- Up to 16 input and/or output registers (configurable from 1 to 32-bits).
- A true dual-port RAM that provides a simple RAM interface available to the user.
- Two asynchronous FIFOs, one from PL to PS and the other from PS to PL, with empty/full, almost empty/full and underflow/overflow status indications.

In the Vivado IP Package version, an AXI-Lite interface was used for the registers and the FIFOs, while an AXI4-Full interface was used for the RAM.

To develop the HIL environment, the custom input/output registers of the ComBlock were utilized to create a memory map interface for the CA2D hardware implementation.

For that, two main parts are required. The first one is the Processor System (PS), which communicates via Ethernet to a Python file in a PC, where the source data of the CA2D model are stored and sent to the SoC. In the same manner, the outputs of the CA2D are read and sent back to the PC using the Ethernet connection. The second part is the programmable logic, which includes the ComBlock IP and the CA2D architecture, both embedded in the FPGA. A schematic of this HIL is presented in Figure 30.

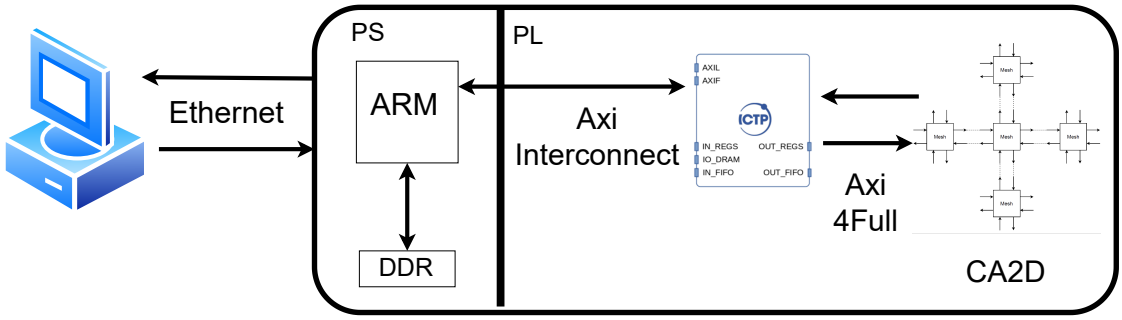


Figure 30 – Simulation results for the wall cell.

3.5.3 Validation Criteria

The validation criteria used in this work are presented below. These validation criteria can be divided into three groups:

- Frequency Domain Validation: These criteria are used to evaluate the consistency of the signals a the receivers cells. The *Fast Fourier Transform* (FFT) is used to obtain the frequency components of the signal at the receivers and compare them with the fundamental frequency components of the test signal used as a source. In addition, the *Power Spectral Density (PSD)*, which measures the energy of the signals, will be used to evaluate the attenuation phenomena produced by the proposed models. These criteria were calculated and plotted offline using Python software.
  - Software: The FFT and PSD results of the two receivers in both DWG and CA2D were compared to the FFT and PSD of the source, aiming to validate the consistency of the two models with the source signals. The FFT was then used to compare the response of the two models.
  - Hardware (simulation and implementation): for each simulation the FFT spectrum and the Power Spectral Density (PSD) plots were collected. The FFT spectrum results address two comparisons: the first one between the FFTs of the source and the receptor in hardware, and the second one between the software and hardware receptors in the same system. On the other hand, the PSD results presents three comparisons: the PSD of the source will be presented in blue,

the PSD in the software receptor will be presented in red, and the PSD in the hardware receptor will be presented in green.

- Validation of the Hardware Implementation: Three validation criteria were used for the hardware implementation. First, the *resource consumption*, which measures the consumption of *Look-Up Tables* (LUTs) Flip-Flops, DSPs blocks, and LUTRAMS or BRAM blocks, used by the CA2D model inside the FPGA. Second, *the power consumption*, which estimates the power consumed by the propsoed CA2D model embedded in the FPGA. Third, *the timing analysis*, which is used to validate the correct timing operation of the CA2D model for the selected frequency clock of 100MHz. The three criteria are obtained after the physical implementation of the CA2D model in the Vivado design suite.
- Performance Validation: To measure the computational performance of the CA2D model in software, the *execution time* of the CA2D is estimated on a desktop computer using the Python profile analysis tool, called *CProfile*. On the other hand, for the hardware implementation, the *latency* and *throughput* will be measured to validate the computational performance of the proposed model. These criteria will be measured using behavioral simulations and using the Integrated Logic Analyzer (ILA-core) from AMD-Xilinx for the physical implementations. Finally, to evaluate the impact of the hardware implementations of the CA2D model, the *acceleration factor* will be used also estimated.

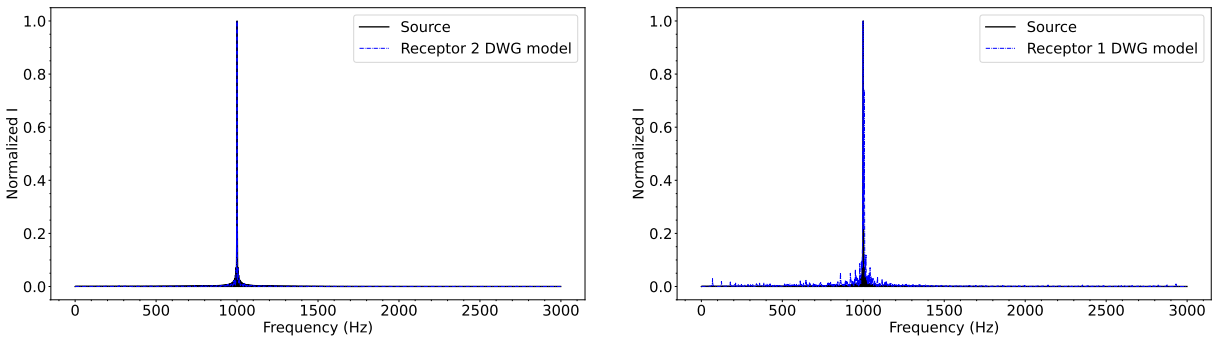
4 Results

This chapter presents the results achieved through the experimental validation of the proposed DWG and CA2D models. As shown in the flowchart of Figure 7, first the frequency analysis of the software implementations for both models is presented, followed by the frequency analysis and characterization of the hardware implementation of the CA2D model. In addition to the frequency analysis, .WAV audio files have been synthesized with each result obtained from the receivers for both implementations in software and hardware, providing a subjective validation of the proposed models.

4.1 Software Validation

4.1.1 Results for the Digital Waveguide Model

Figures 31 and 32 present FFT and PSD results obtained using the 1 kHz sinusoidal signal for the case study shown in Figure 14. It is possible to conclude that the received signals are consistent with the source signal, as the 1 kHz frequency of the source is the predominant frequency in both the receivers. However, in receiver 1 (obstructed by the obstacle), low-magnitude frequencies are observed around the main frequency. These minor frequencies can be attributed to reverberation phenomena, caused by reflections of the sound waves interacting with walls and obstacles. Figures 32a and 32b present the power spectral density at receivers 2 and 1, respectively. The attenuation produced by the barrier in receiver 1 is noticeable.

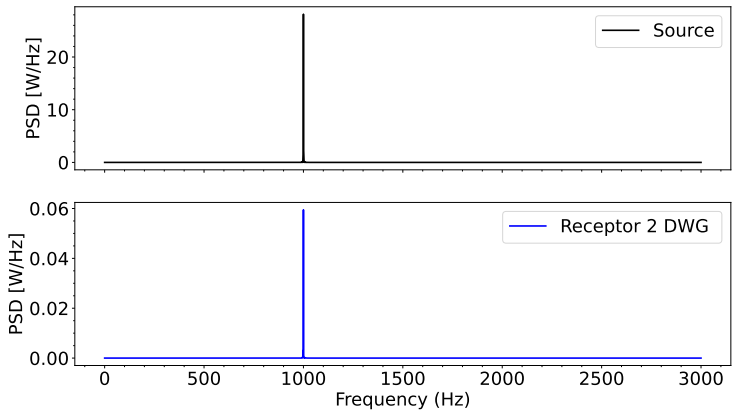


(a) FFT Spectrum Receptor 2.

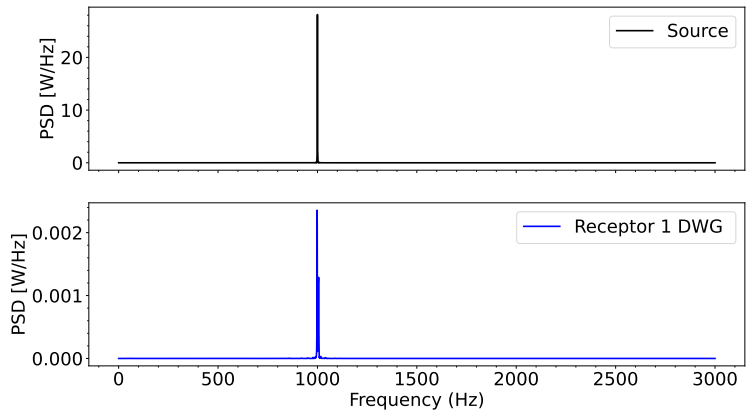
(b) FFT Spectrum Receptor 1.

Figure 31 – FFT results of the 64 × 64 DWG software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

Figures 33 and 34 refer to the results obtained using the G major chord guitar signal for the case study shown in Figure 14. Figures 33a and 33b show the FFT of the signal at



(a) PSD Spectrum Receptor 2.



(b) PSD Spectrum Receptor 1.

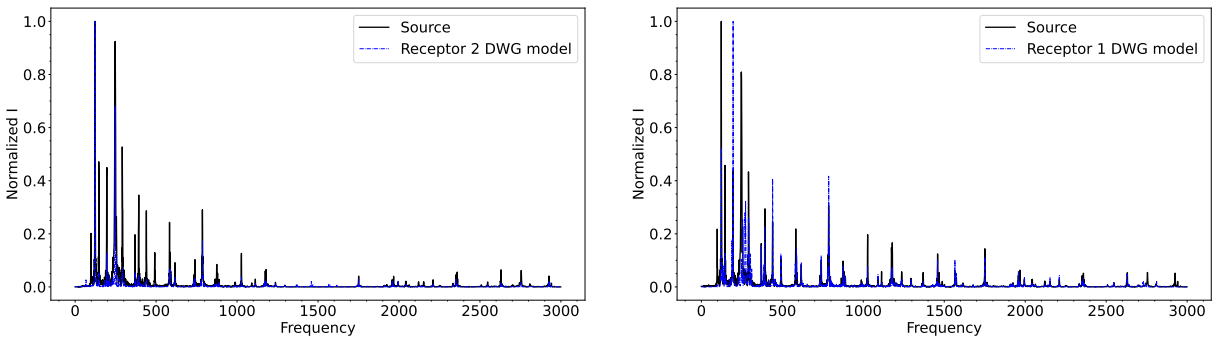
Figure 32 – PSD results for the 64 × 64 DWG software implementation using the sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

receivers 2 and 1, respectively. In receiver 2 (free field), the two most intense frequencies from the source are present, and many of the low-intensity frequencies are attenuated. Conversely, in receiver 1 (obstructed), some low-intensity frequencies from the source are amplified. This amplification can be attributed to the reverberation caused by the barrier in front of this receiver. Figures 34a and 34b show the power spectral density of the signals at receptors 2 and 1, respectively. At receptor 2, the two most intense frequencies correspond to the main frequencies of the source. Conversely, Figure 34b illustrates how the reverberation effect amplifies some of the lower-intensity frequencies from the source while attenuating the main frequencies in receptor 1.

4.1.2 Results for the Cellular Automata Model

Figures 35 and 36 refer to the results obtained by software implementation of the CA2D model, using the 1 kHz sinusoidal signal for the case study shown in Figure 29.

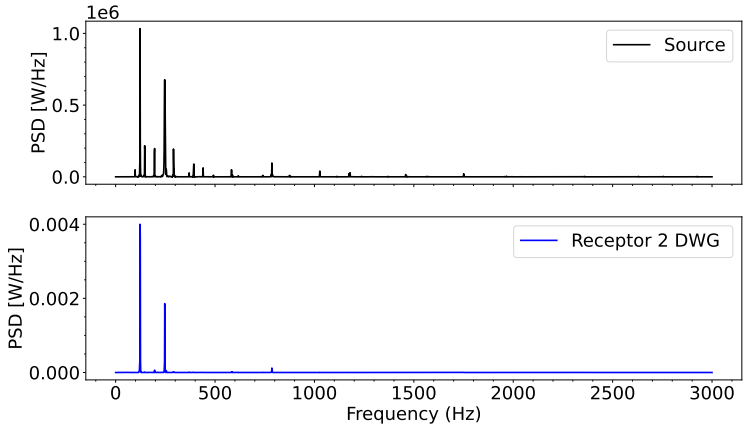
Figures 35a and 35b show the FFT of the signal at receptors 2 and 1, respectively.



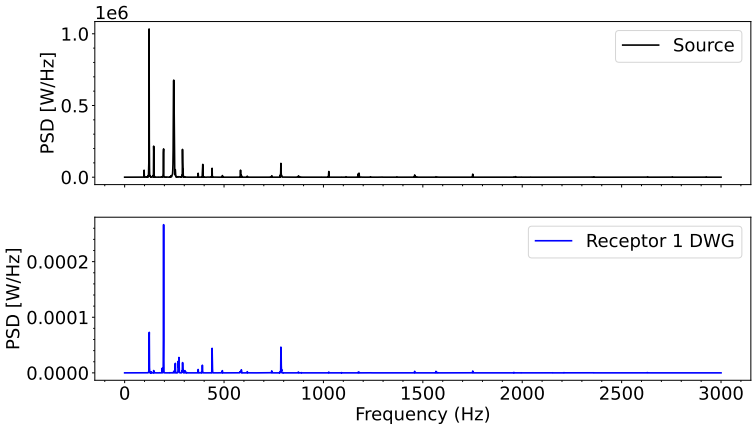
(a) FFT Spectrum Receptor 2.

(b) FFT Spectrum Receptor 1.

Figure 33 – FFT results of the 64 × 64 DWG software implementation using the G major chord guitar: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).



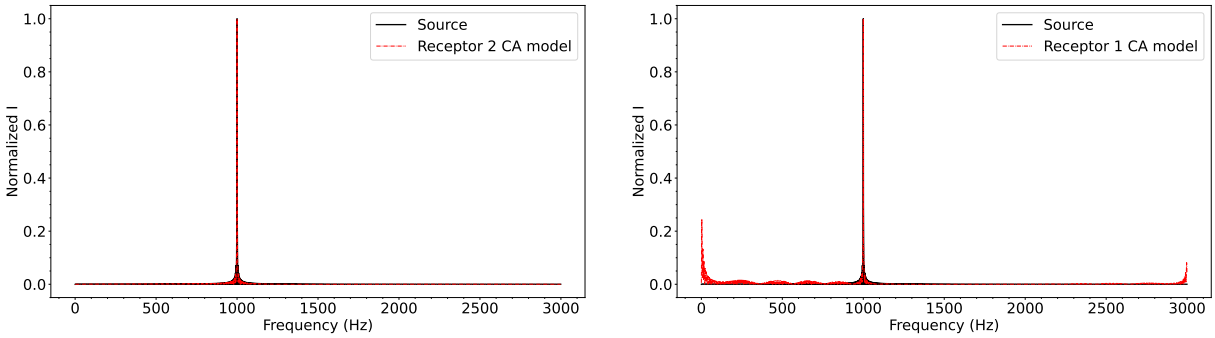
(a) PSD Spectrum Receptor 2.



(b) PSD Spectrum Receptor 1.

Figure 34 – PSD results of the 64 × 64 DWG software implementation using a G major chord guitar: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

These figures indicate that the mainly frequency for both receivers is the primary frequency of 1 kHz from the source. Similarly to the DWG model, in receptor 1 (obstructed), low-intensity oscillating frequencies in the range of 0 to 1000 Hz are observed, which could be the effect of the reverberation from the barrier and walls. One possible explanation for this phenomenon is that the reflected waves, which reach receiver 1, contaminate the main frequency due to periodic events caused by reflections and constructive/destructive interferences. Figures 36a and 36b show the Power Spectral Density of the signals in receivers 2 and 1, respectively. These results demonstrate that the highest power frequency in both receivers is the 1 kHz frequency originating from the source. Additionally, it is observed that the frequencies in the obstructed receiver are of much lower intensity.



(a) FFT Spectrum Receptor 2.

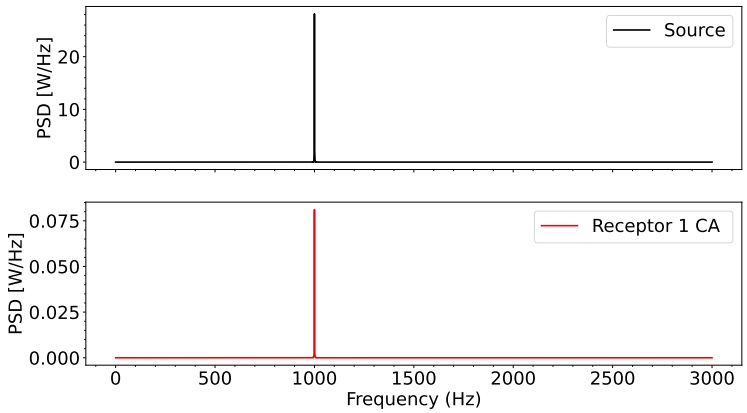
(b) FFT Spectrum Receptor 1.

Figure 35 – FFT results of the 64 × 64 CA2D software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

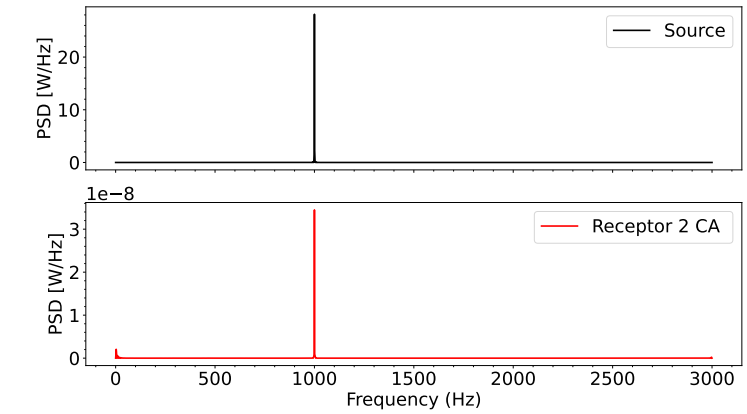
Figures 37 and 38 present the results obtained by the software implementation of the CA2D model for a G major chord guitar signal.

Figures 37a and 37b show the FFT of the signal received in receptors 2 and 1, respectively. It is observed that the signal from receiver 2 (free field) contains a spectrum very similar to that of the source, with some frequencies amplified. In contrast, receiver 1 exhibits reverberation effects with low intensity in the frequency range from 0 to 500 Hz and attenuation for frequencies above 500 Hz. Figures 38a and 38b show the Power Spectral Density of the signals in receptors 2 and 1, respectively. On the one hand, it should be noted that the PSD of receiver 2 is lower than that of the source due to the attenuations introduced by the CA2D model. On the other hand, in receiver 1, the PSD clearly shows the reverberation effect at frequencies in the range of 0 to 500 Hz, along with the evident attenuation of other frequencies. It is also observed that stronger attenuations across the entire spectrum are introduced by the CA2D model.

All the frequency responses demonstrate that both the DWG and CA2D models effectively replicate the behavior of acoustic wave propagation. The proposed CA2D model faithfully reproduces the source signal, whether it is a single-frequency signal or a more complex signal, such as that of a guitar chord. In order to compare the two models, the G

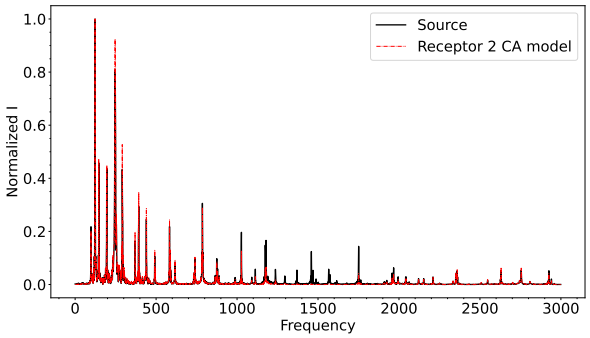


(a) PSD Spectrum Receptor 2.

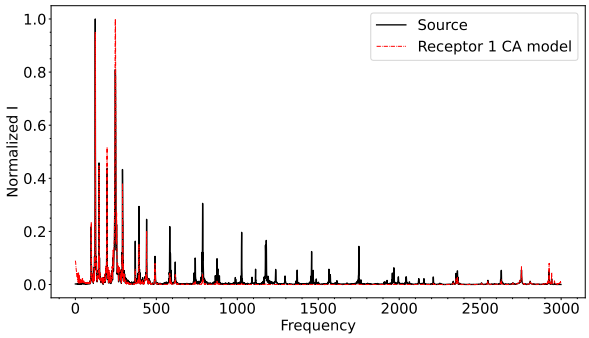


(b) PSD Spectrum Receptor 1.

Figure 36 – PSD results of the 64 × 64 CA2D software implementation using a sinusoidal signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

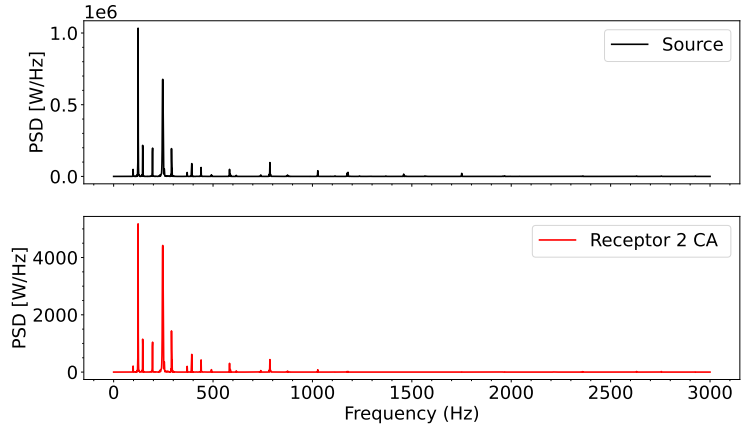


(a) FFT Spectrum Receptor 2.

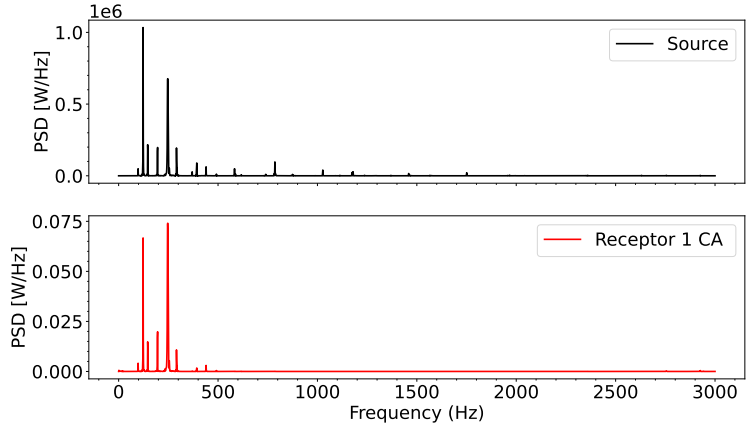


(b) FFT Spectrum Receptor 1.

Figure 37 – FFT results of the 64 × 64 CA2D software implementation using a G major Chord signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).



(a) PSD Spectrum Receptor 2.



(b) PSD Spectrum Receptor 1.

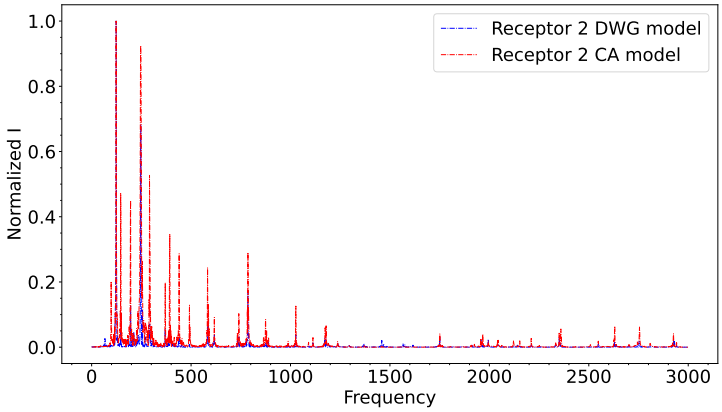
Figure 38 – PSD results of the 64×64 CA2D software implementation using a G major Chord signal: a) Receiver 2 (before the barrier). b) Receiver 1 (after the barrier).

major chord spectra of both receivers were overlaid in Figure 39. One can conclude that the CA2D model in the free field receiver is more faithful to the input signal if compared to the DWG model, capturing more frequencies of the source signal. On the other hand, the CA2D model in the obstructed receiver acts as a low-pass filter, attenuating frequencies above 500 Hz. This effect is more noticeable in the power spectral density graphs of the DWG (Figure 34) and CA2D (Figure 38).

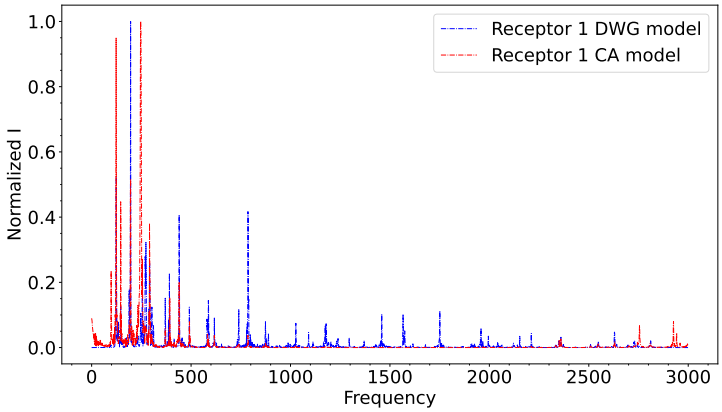
The audio files obtained from the results of the two receivers when the mesh is exposed to the guitar chord signal are used to perform an audible comparison of the two models. The following video illustrates this subjective comparison video [1].

To compare the computational performance of these two models, the execution times were measured in the same scenario using the G major guitar chord as the source signal. To provide statistical support for these measurements, twenty simulations were performed with each model to calculate an average execution time. The results are summarized in Table 5.

[1] Link video: https://youtu.be/n8YJcOShi5Q



(a) FFT Spectrum Receptor 2.



(b) FFT Spectrum Receptor 1.

Figure 39 – DWG and CA2D spectrum comparison for the G major chord guitar signal: a) Receiver 2 (before the barrier). b): Receiver 1 (after the barrier).

These results demonstrate that DWG remains computationally more efficient than CA2D. This discrepancy is likely due to the conditionals used by CA2D to differentiate cell types and their respective behaviors. These conditions must be evaluated for all cells in the mesh continuously until the simulation is complete. As the mesh size increases, the number of conditionals also increases, leading to higher computational costs for CA2D. This increased computational cost is reflected in longer execution times.

Model	Mean execution time (s)
DWG	1.119
CA2D	2.271

Table 5 – Software execution time for the meshes studied in Figure 14 and 29.

The frequency results confirmed the expected behavior of the proposed CA2D model. The following section presents the results obtained from its hardware implementation using SoC-FPGAs.

4.2 Hardware Implementation of the CA2D model

As will be discussed, the mesh size in the SoC-FPGA is constrained by the available resources. Accordingly, it was possible to implement two different mesh sizes, 10×10 and 20×20. The former uses 16-bits and 27-bits, and the latter uses 16-bits.

4.2.1 10×10 Mesh on the ZCU104 Board

A 10×10 mesh was implemented in the ZCU104 development board using a clock frequency of 100 MHz. Figure 40 illustrates a schematic of the implemented 10×10 mesh, with the source highlighted in red and the receiver in purple. The circuit layout of this mesh is shown in Figure 41 and Table 6 shows the resource consumption. It should be noted that the resources available within this FPGA are greater compared to those of the Pynq Z2. Therefore, the Zynq Ultrascale FPGA will be utilized to scale the CA2D.

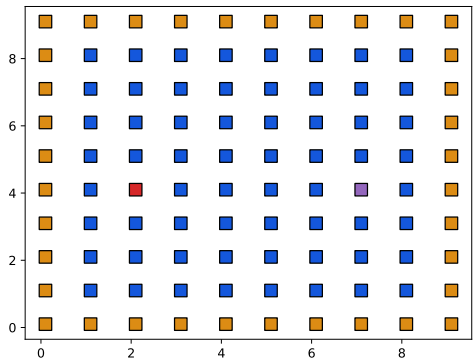


Figure 40 – 10×10 implemented mesh.

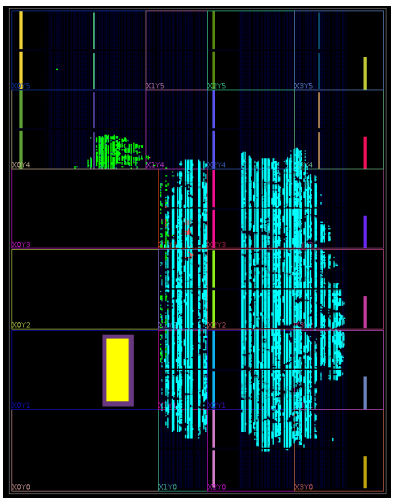


Figure 41 – Mesh 10×10 27 bits.

Resource	Utilization	Available	Utilization
LUT	61289	230400	27.64
LUTRAM	203	101760	0.20
FF	39076	460800	8.48
DSP	126	1728	7.29
BUFG	3	544	0.55

Table 6 – 10×10 resources consumption.

The *power estimate* shows a static power of 0.693 W and a dynamic power of 2.928 W, where 84% is attributed to the ARM processor. The total power consumption was 3.621 W. It

should be noted that the ZCU104 could implement larger meshes, but with an increase in energy consumption. In the *timing* analysis, the setup and hold critical paths are positive, 3.433 ns and 0.010 ns, respectively. This result shows that the timing constraints are not violated for a 100 MHz clock for this mesh size.

The implementation with 27-bit representation has high resource consumption, especially in the Pynq Z2. To implement larger meshes, the adder and multiplier IPs were regenerated using a 16-bit representation to reduce resource consumption. Using these new 16-bit IP cores, the CA2D cells were regenerated to implement larger meshes.

4.2.2 10×10 Mesh on the Pynq Z2 Board

A first implementation using the 16-bits IP cores was a 10×10 mesh using a clock frequency of 100 MHz, with 96 mesh cells. Figure 40 depicts this mesh. The circuit layout of this mesh is shown in Figure 42 and Table 7 shows the resource consumption.

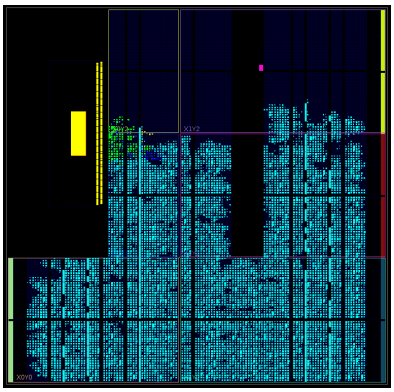


Figure 42 – Mesh 10×10 16 bits.

Resource	Utilization	Available	Utilization
LUT	31039	53200	58.34
LUTRAM	44	17400	0.25
FF	22121	106400	20.79
DSP	126	220	57.27
BUFG	1	12	3.13

Table 7 – 10×10 resources consumption.

The *power estimate* shows a static power of 0.142 W and a dynamic power of 1.588 W, where 79% is attributed to the ARM processor. The total power consumption was 1.731 W. Note how using 16-bit IPs results in noticeably lower dynamic power consumption than using 27-bit IPs, but for a larger mesh. The *timing* analysis shows that the setup and hold critical paths have positive slacks, 0.405 ns, and 0.016 ns, respectively. This result shows that the circuits were effectively mapped using a clock frequency of 100 MHz.

The next implementation aimed to increase the size of the previously implemented mesh. This mesh was mapped on the ZCU104, which has many more resources than the Pynq Z2.

4.2.3 20×20 Mesh on the ZCU104 Board

In the ZCU104, using this 16-bit representation, it is possible to implement a 20×20 mesh using a clock frequency of 100 MHz. Figure 43 illustrates a schematic of the implemented mesh, with the source highlighted in red and the receiver in purple. The circuit

layout of this mesh is shown in Figure 44 and Table 8 shows the resource consumption. This 20 × 20 mesh has 396 cells, compared to the 10 × 10 mesh with 96 cells previously implemented using the same board. As expected, the percentages of resource occupation increased.

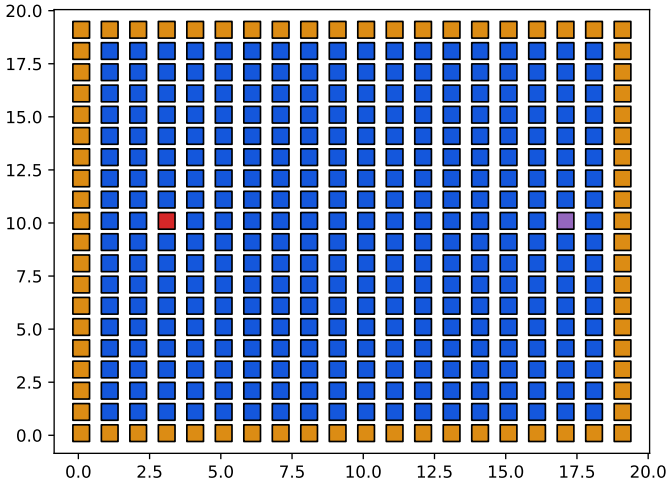


Figure 43 – 20 × 20 implemented mesh.

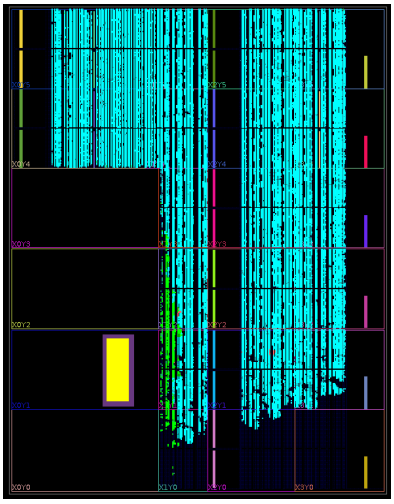


Figure 44 – Mesh 20 × 20 16 bits.

Resource	Utilization	Available	Utilization
LUT	154784	230400	67.18
LUTRAM	203	101760	0.20
FF	110790	460800	24.04
DSP	646	1728	37.38
BUFG	3	544	0.55

Table 8 – 20 × 20 resources consumption.

The *power estimate* shows a static power of 0.697 W and a dynamic power of 3.467 W, where 71% is attributed to the ARM processor. The total power consumption was 4.164 W. Note how this implementation with 300 more cells increases power consumption. The *timing* analysis shows that the setup and hold critical paths have positive slacks of 3.271 ns and 0.010 ns, respectively. This *timing* results demonstrate that the CA2D circuit is successfully implemented using a 100 MHz clock.

To complete the validation of the CA2D model proposed in Subsection 3.3.2, obstacles have been incorporated into the 10 × 10 and 20 × 20 meshes implemented using 16-bit IP

cores. The results of these implementations are shown in the next subsections.

4.2.4 10×10 Mesh with a Barrier on the Pynq Z2

Figure 45 presents a schematic of the 10×10 mesh with obstacles, where the source is highlighted in red, the receiver in purple, and the barrier in brown. This mesh was implemented using 16-bit IP cores on the Pynq Z2 board using an operating clock frequency of 100 MHz. The circuit layout of this mesh is shown in Figure 46 and Table 9 shows the resource occupation in the Pynq Z2. The *power estimate* shows a static power of 0.143 W and a dynamic power of 1.591 W, where 78% is attributed to the processor. The total power consumption was 1.591 W. The *timing* analysis shows setup and hold slacks of 0.632 ns and 0.017 ns, respectively. The circuit was effectively mapped on the Zynq7020 device using a 100 MHz clock.

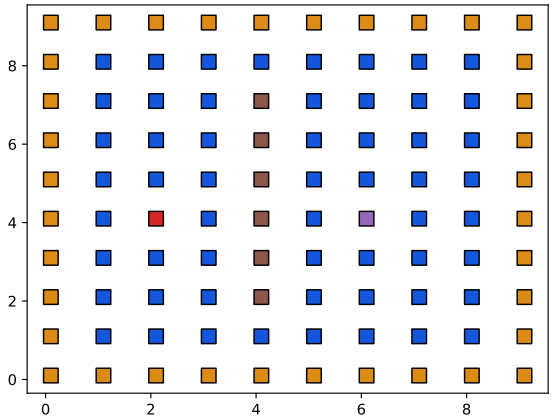


Figure 45 – 10×10 mesh with a barrier.

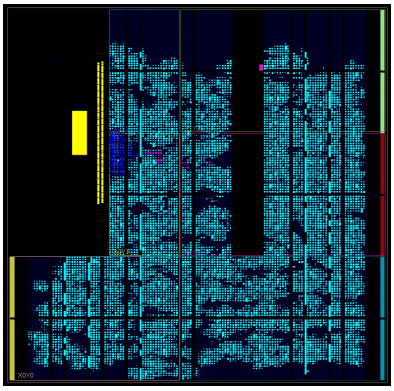


Figure 46 – Mesh 10×10 16 bits and barrier.

Resource	Utilization	Available	Utilization
LUT	31014	53200	58.30
LUTRAM	44	17400	0.25
FF	22121	106400	20.79
DSP	126	220	57.27
BUFG	1	32	3.13

Table 9 – 10×10 with barrier resources consumption.

4.2.5 20×20 Mesh with a Barrier on the ZCU104

Figure 47 presents a schematic of the 20×20 mesh with obstacles, where the source is highlighted in red, the receiver in purple, and the barrier in brown. This mesh was implemented using 16-bit IP cores on the ZCU104 board with a clock frequency of 100 MHz.

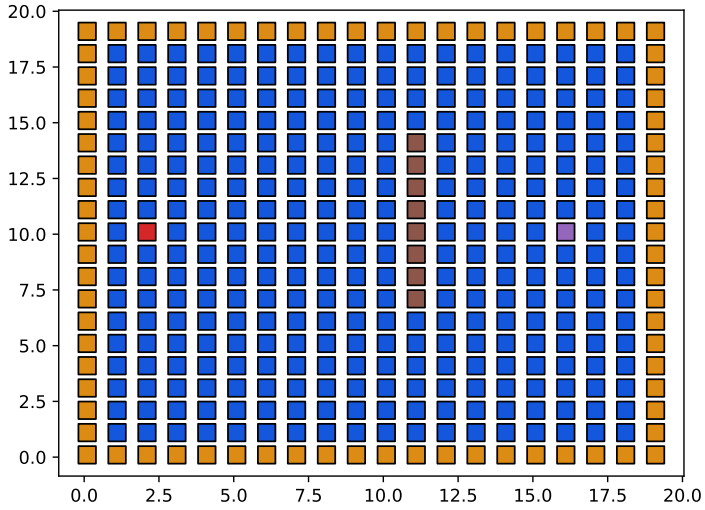


Figure 47 – 20×20 mesh with obstacle.

The circuit layout of this mesh is shown in Figure 48 and Table 10 shows the resources consumption in the Zynq Ultrascale+ of the ZCU104 board. The *power estimate* shows a static power of 0.697 W and a dynamic power of 3.485 W, where 83% is attributed to the ARM processor. The total power consumption was 4.182 W. The *timing* analysis demonstrated positive slacks for setup and hold, with values of 3.668 ns and 0.010 ns, respectively. Thus, one can conclude that the proposed CA2D circuits can operate at a clock frequency of 100 MHz in the Zynq Ultrascale+ of the ZCU104 board.

4.3 Hardware Validation using Behavioral Simulation

This section first present the FFT and PSD results for each possible configuration depicted in Section 3.5 and then provide some observations and discussions.

4.3.1 Mesh 10×10 using 27-bits on the ZCU104

Figure 49 presents the results of the behavioral simulation using 1 KHz sinusoidal and the G major guitar chord signals. Figures 49a and 49c depict the FFT and PSD results for the 1 KHz sinusoidal signal. Figures 49b and 49d show the frequency response for the G major guitar chord signal.

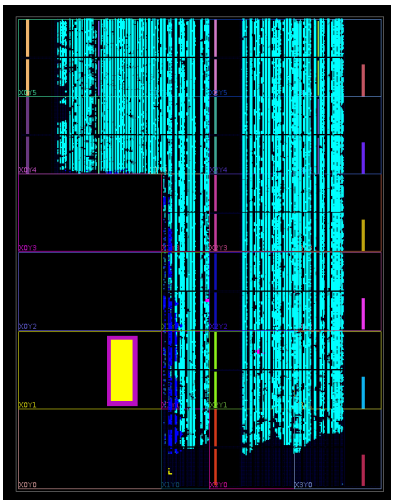
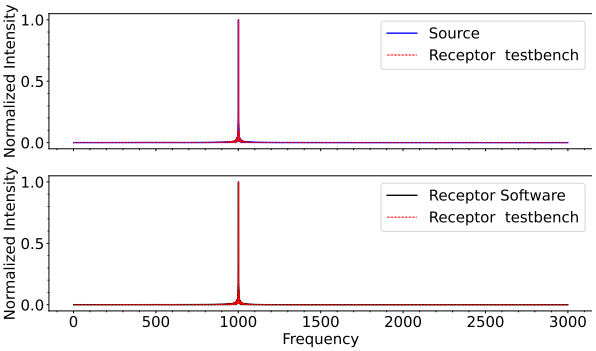


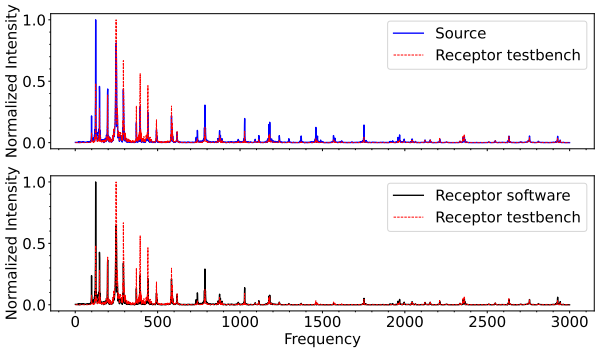
Figure 48 – Mesh 20×20 with obstacle 16 bits.

Resource	Utilization	Available	Utilization
LUT	154734	230400	67.16
LUTRAM	203	101760	0.20
FF	110790	460800	24.04
DSP	646	1728	37.38
BUFG	3	544	0.55

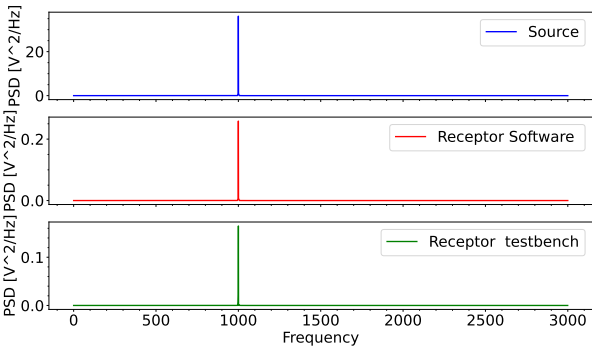
Table 10 – 20×20 with barrier resources consumption.



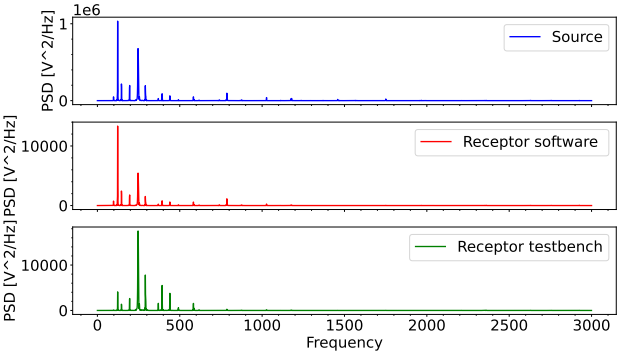
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).



(c) Power Spectral Density (sinusoidal).

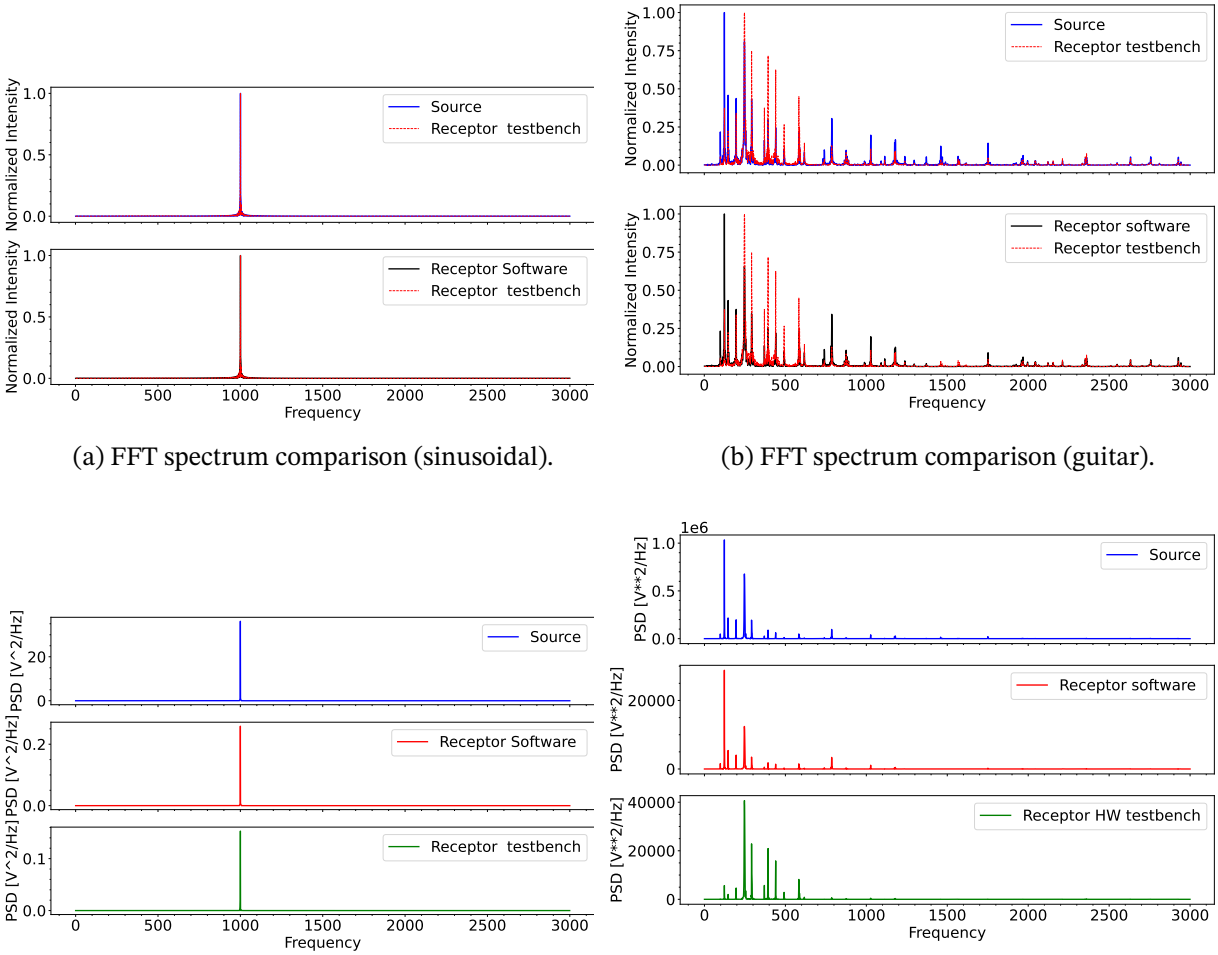


(d) Power Spectral Density (guitar).

Figure 49 – Simulation results of the hardware implementation of the CA2D using 27-bits and 10×10 mesh. a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major chord signals, respectively.

4.3.2 Mesh 10 × 10 16-bits on Pynq Z2

Figure 50 presents the results of the behavioral simulation for the 1 KHz sinusoidal and the G major chord signals. Figures 50a and 50c depict the frequency response for the sinusoidal signal. Figures 50b and 50d show the FFT and PSD results for the G major chord signal.



(a) FFT spectrum comparison (sinusoidal).

(b) FFT spectrum comparison (guitar).

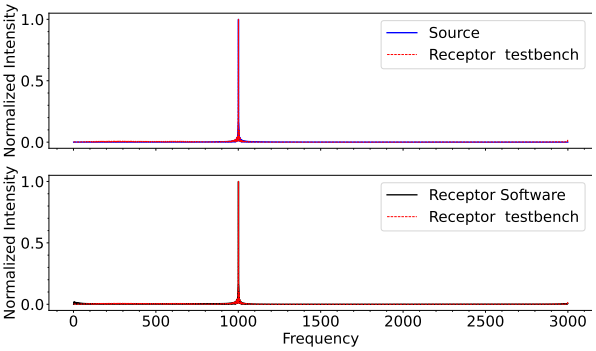
(c) Power Spectral Density (sinusoidal).

(d) Power Spectral Density (guitar).

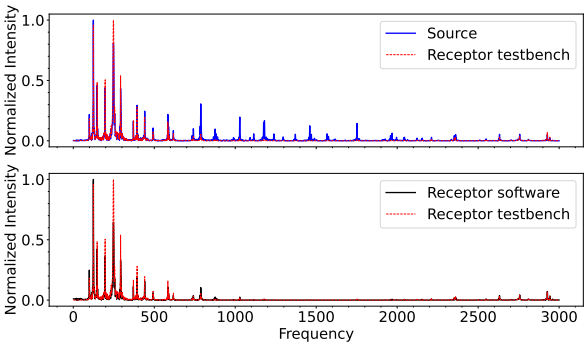
Figure 50 – Simulation results of the CA2D using 16-bits and 10 × 10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

4.3.3 Mesh 20 × 20 using 16-bits on ZCU104

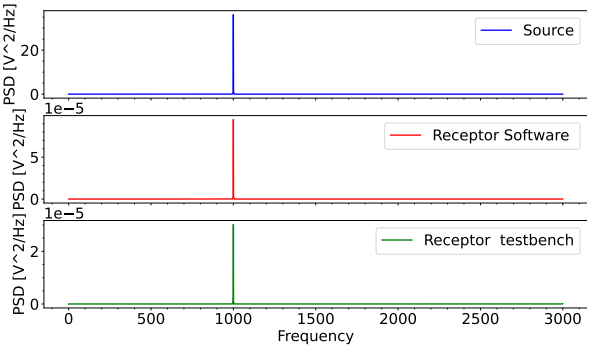
Figure 51 presents the results of the behavioral simulation using the 1 KHz sinusoidal and the G major chord signals. Figures 51a and 51c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 51b and 51d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.



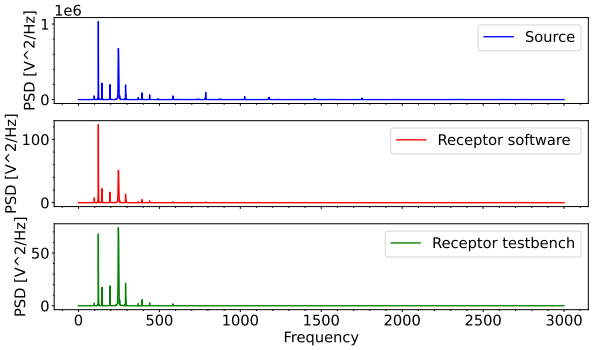
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).



(c) Power Spectral Density (sinusoidal).



(d) Power Spectral Density (guitar).

Figure 51 – Simulation results of the CA2D using 16-bits and 20 × 20 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

4.3.4 Mesh 10 × 10 using 16-bits with obstacle

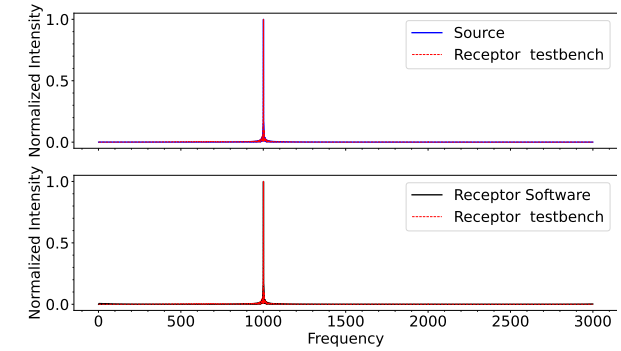
Figure 52 presents the results of the behavioral simulation using the 1 KHz sinusoidal and the G major chord signals. Figures 52a and 52c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 52b and 52d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.

4.3.5 Mesh 20 × 20 using 16-bits with obstacle

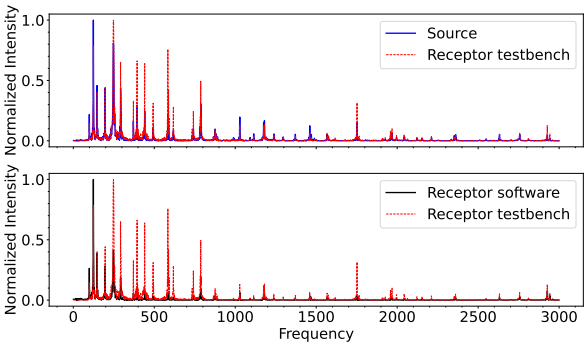
Figure 53 presents the results of the behavioral simulation using the 1 KHz sinusoidal and the G major chord signals. Figures 53a and 53c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 53b and 53d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.

4.3.6 Discussion regarding the behavioral simulation

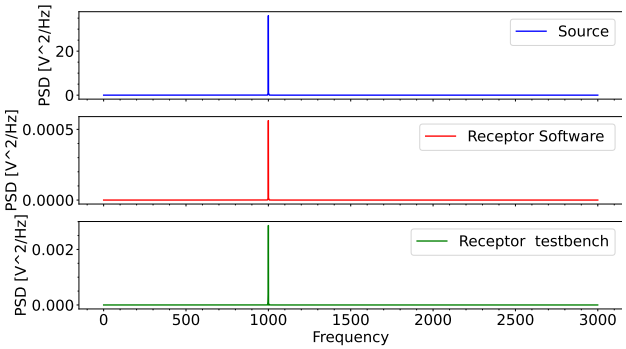
In Figures 49a, 50a, 51a, 52a and 53a, which present the results obtained from the FFT using a 1KHz sinusoidal signal as the source, it is observed that for all mesh sizes and



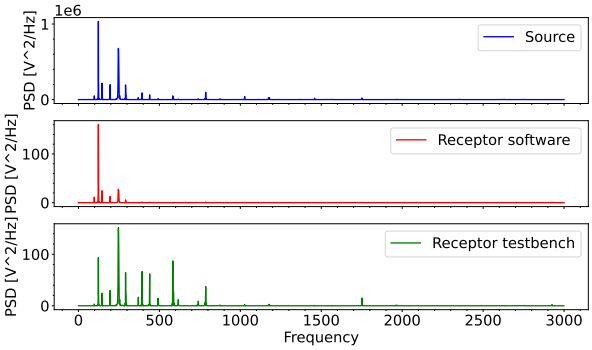
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).



(c) Power Spectral Density (sinusoidal).



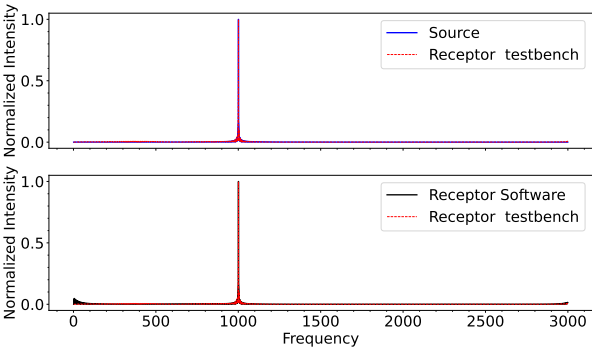
(d) Power Spectral Density (guitar).

Figure 52 – Simulation results of the CA2D using 16-bits and 10×10 mesh with osbtacle: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

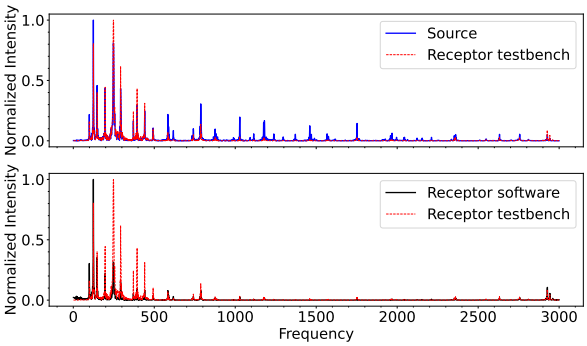
regardless of the presence of obstacles, the CA2D model manage to retain the expected frequency of this test signal.

On the other hand, the PSD results offer a different type of analysis. Firstly, Figures 49c and 50c, corresponding to 27 bits and 16 bits for the same mesh size of 10×10, respectively, show that the signals in the receivers have the same order of magnitude in energy, indicating that changing the word length does not affect the results. Secondly, comparing the PSDs of the 10×10 mesh with the 20×20 mesh (Figures 50c and 51c) demonstrates that increasing the mesh size results in greater attenuation (around 10^{-4} in magnitude order) of the signals at the receivers. Finally, Figures 52c and 53c illustrate that the inclusion of obstacles further increases the attenuation experienced by the signals at the receivers. For the analyzed configurations, one can observe that the CA2D model does not capture reverberation frequencies.

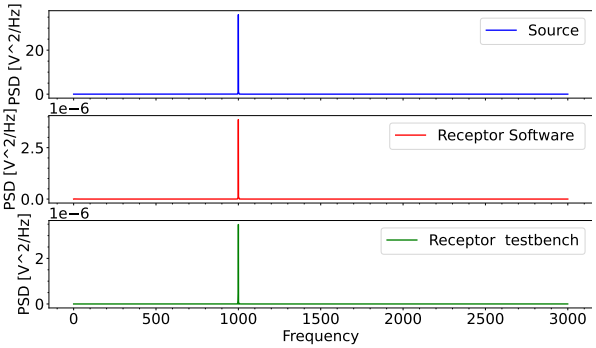
In the case of the G guitar chord signal, the FFTs in Figures 49b and 50b corresponding to meshes of 10×10 for 27 bits and 16 bits, respectively, show that the model intensified the signal frequencies between 300 and 500 Hz, and attenuated the frequencies outside this range. On the other hand, in the Figure 51b corresponding to the FFT of the mesh 20×20, a



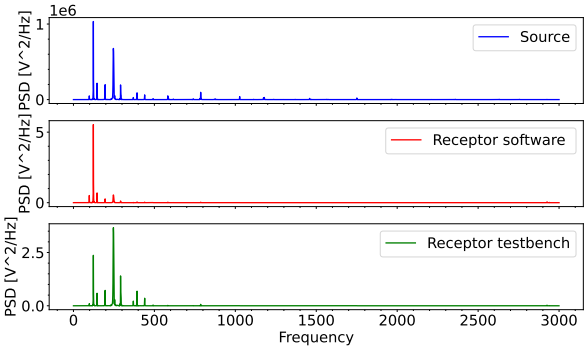
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).



(c) Power Spectral Density (sinusoidal).



(d) Power Spectral Density (guitar).

Figure 53 – Simulation results of the CA2D using 16-bits and 20×20 mesh with osbtacle: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

frequency spectrum more similar to that of the source was observed. Figures 52b and 53b show that the inclusion of obstacles increases the reverberation within the mesh, which increases the intensity of some frequencies present in the guitar signal.

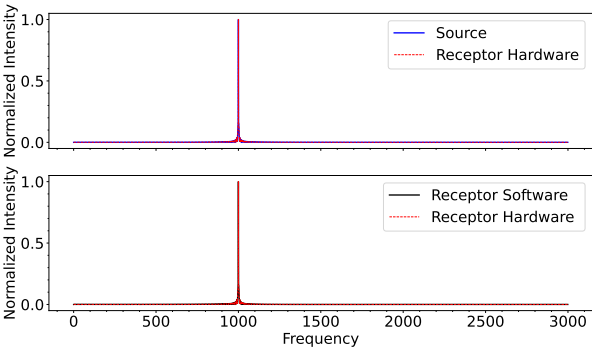
Regarding the PSD results for the free field receptor, it can be observed that the 10×10 implementations intensified the signal frequencies between 300 and 600 Hz, and attenuated the frequencies outside this range. The effect of increasing the mesh size to 20×20 resulted on a more similar spectrum but with an attenuation comparable to the software implementation. Finally, regarding the PSD results for the blocked receiver, it was observed that the CA2D model in hardware better preserves the main frequencies of the signal if compared to the software implementation. Similarly to the free field receptor, the effect of increasing the mesh size produces a more similar spectrum if compared to the software implementation with a comparable attenuation.

4.4 Validation of the Physical Implementation

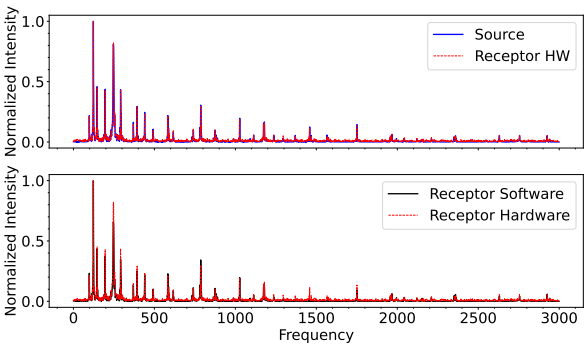
The physical hardware implementation of the CA2D model uses the hardware-in-the-loop scheme shown in Figure 30.

4.4.1 Mesh 10 × 10 using 27-bits on the ZCU104

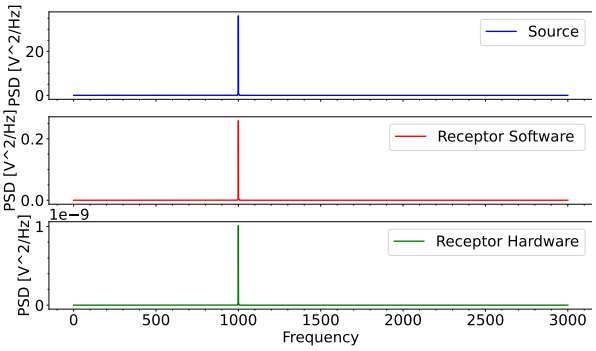
Figure 54 presents the results of the hardware implementation using the 1 KHz sinusoidal and the G major chord signals. Figures 54a and 54c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 54b and 54d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.



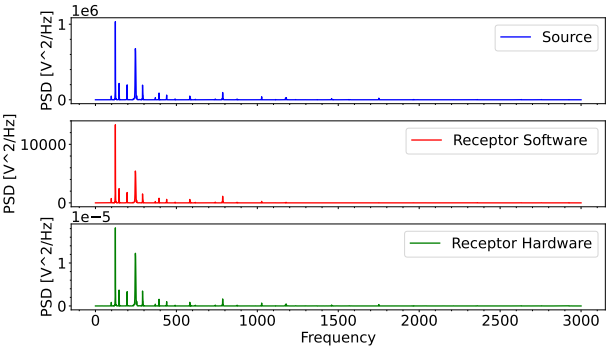
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).



(c) Power Spectral Density (sinusoidal).

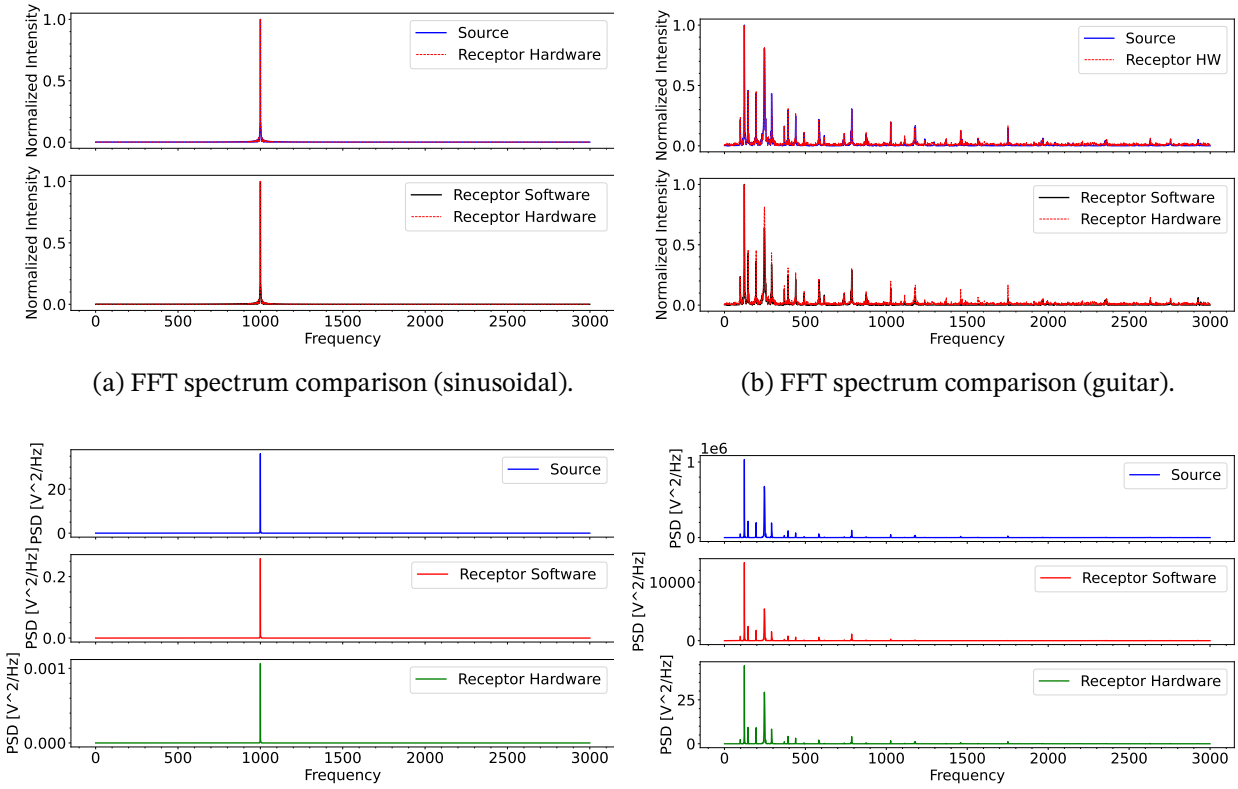


(d) Power Spectral Density (guitar).

Figure 54 – Hardware implementation results of the CA2D using 27-bits and 10 × 10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

4.4.2 Mesh 10 × 10 using 16-bits on the Pynq Z2

Figure 55 presents the results of the hardware implementation using the 1 KHz sinusoidal and the G major chord signals. Figures 55a and 55c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 55b and 55d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.



(a) FFT spectrum comparison (sinusoidal).

(b) FFT spectrum comparison (guitar).

(c) Power Spectral Density (sinusoidal).

(d) Power Spectral Density (guitar).

Figure 55 – Hardware implementation results of the CA2D using 16-bits and 10×10 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

4.4.3 Mesh 20×20 using 16-bits on the ZCU104

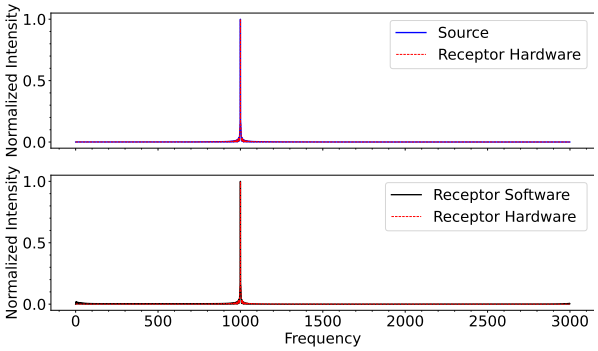
Figure 56 presents the results of the Hardware implementation using the 1 KHz sinusoidal and the G major chord signals. Figures 56a and 56c depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively. Figures 56b and 56d show the FFT spectrum comparison and PSD for the G major chord signal, respectively.

4.4.4 Mesh 10×10 using 16-bits on the Pynq Z2 with obstacle

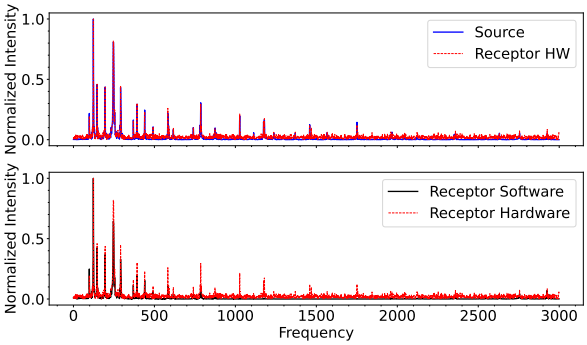
Figure 57 presents the results of the Hardware implementation using the 1 KHz sinusoidal. Figures 57a and 57b depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively.

4.4.5 Mesh 20×20 using 16-bits on the ZCU104 with obstacle

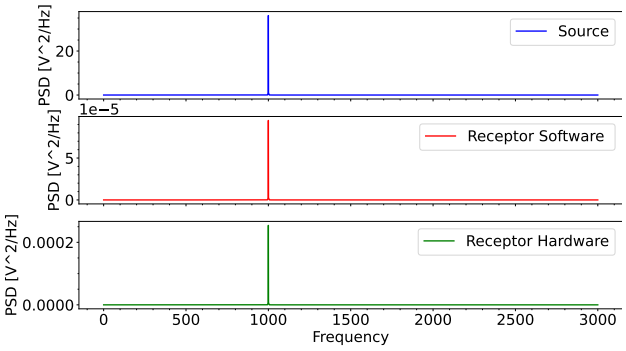
Figure 58 presents the results of the Hardware implementation using the 1 KHz sinusoidal. Figures 58a and 58b depict the FFT spectrum comparison and PSD for the sinusoidal signal, respectively.



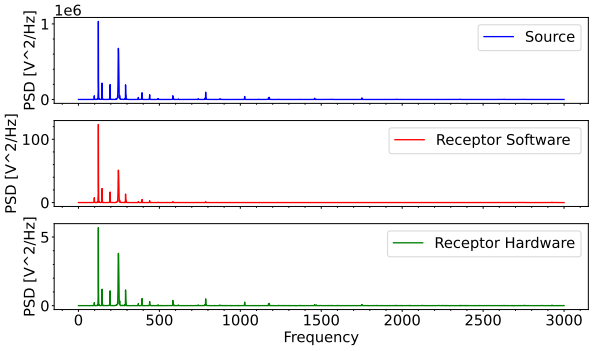
(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).

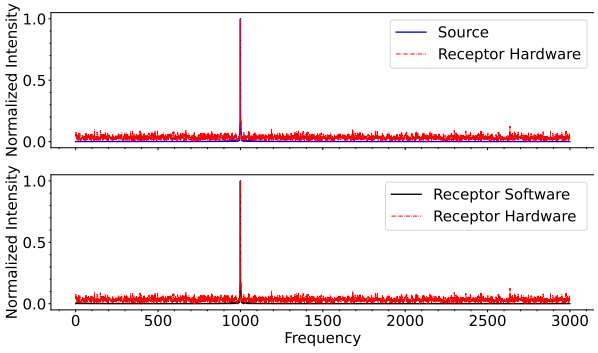


(c) Power Spectral Density (sinusoidal).

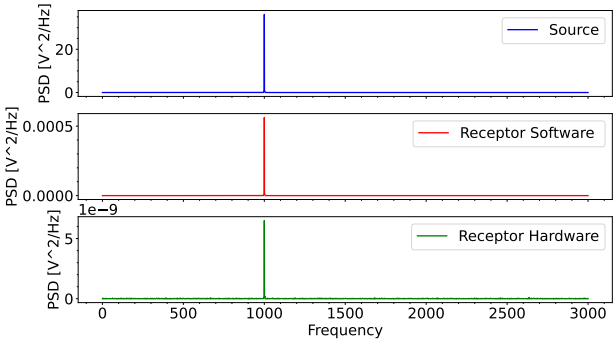


(d) Power Spectral Density (guitar).

Figure 56 – Hardware implementation results of the CA2D using 16-bits and 20×20 mesh: a) and b) FFT spectrum comparison for 1 KHz sinusoidal and G major guitar chord signals, respectively. c) and d) PSD of the source, receptor in software, and receptor in hardware for 1 KHz sinusoidal and G major guitar chord signals, respectively.

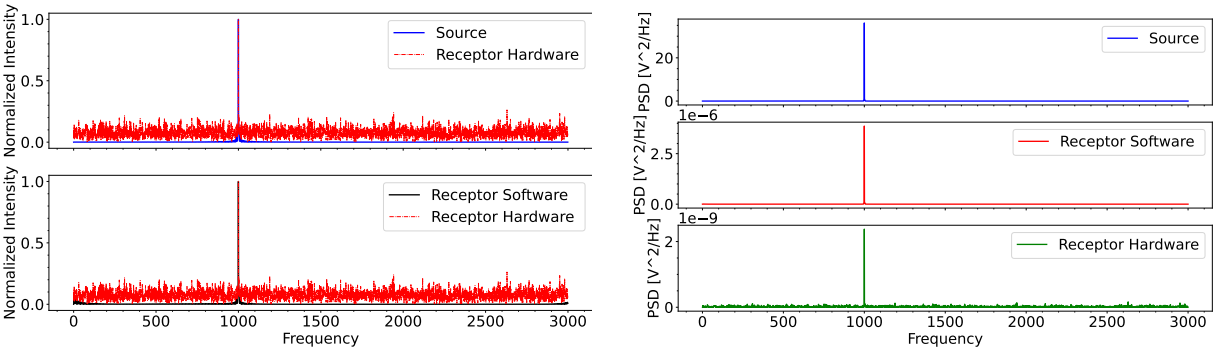


(a) FFT spectrum comparison (sinusoidal).



(b) FFT spectrum comparison (guitar).

Figure 57 – Hardware implementation results of the CA2D using 16-bits and 10×10 mesh with obstacle: a) and b) FFT spectrum and PSD comparison for 1 KHz sinusoidal, respectively.



(a) FFT spectrum comparison (sinusoidal).

(b) FFT spectrum comparison (guitar).

Figure 58 – Hardware implementation results of the CA2D using 16-bits and 20×20 mesh with obstacle: a) and b) FFT spectrum and PSD comparison for 1 KHz sinusoidal, respectively.

4.4.6 Discussion regarding the physical implementation

In Figures 54a, 55a, 56a, 57a and 58a, which present the results obtained from the FFT using a 1KHz sinusoidal signal as the source, it is observed that for all mesh sizes the model manage to retain the expected frequency. However, the hardware implementation suffers from a strong attenuation (between 10^{-2} and 10^{-7} in magnitude order) if compared to the software approach, as demonstrated in Figures 54c and 55c. The presence of obstacles attenuated in a similar way both the hardware and software implementations (between 10^{-3} and 10^{-5} in magnitude order), as demonstrated by Figures 57b and 58b. However, it is important to point out that the hardware implementation resulted in such strong attenuation that the base of the spectrum is approximately 10% of the main frequency peak. A lesser attenuation was observed when the numerical precision was reduced from 27 bits to 16 bits. When increasing the mesh size, it was observed that the 20×20 mesh configuration attenuates the signal by one magnitude order.

In Figures 54b, 55b, and 56b, which present the results obtained from the FFT using the G guitar chord signal, it is observed that for all mesh sizes the model manage to represent the expected frequencies. In particular, the hardware implementation better represents the six main peak frequencies than the software model. However, the hardware implementation suffers from a greater attenuation than the software implementation, as can be observed in Figures 54d, 55d, and 56d. The effect of reducing the bit-width representation from 27-bits to 16-bits resulted in a lesser attenuation (PSD values of 10^{-9} for 27-bits and 10^{-5} for 16-bits). Increasing the mesh size from 10×10 to 20×20 resulted in a slight attenuation (5 times lower in PSD values).

A remarkable observation is that when including obstacles, both the 10×10 and the 20×20 meshes experienced such strong attenuation that their signal levels became indistinguishable from the noise level, making it impossible to obtain a discernible signal.

4.5 Hearing Validation of the CA2D Model

As observed in the FFT and PSD spectra obtained from the free field receiver, one can conclude from the results using the behavioral simulation and the physical implementation, that the proposed CA2D model is a feasible solution to simulate acoustic phenomena, capturing many characteristic frequencies of the source signal. This holds true for both study cases: the sinusoidal signal and the G major guitar chord signal. Audio files obtained for the three scenarios using the G major guitar chord as a source signal are shown in Table 11.

Mesh size, bit-width	Link
10×10, 27-bits	https://youtu.be/alDDvRdkU0o
10×10, 16-bits	https://youtu.be/D36LPwW7fEs
20×20, 16-bits	https://youtu.be/pj1XPQvHY9k

Table 11 – Audio links with the hearing validation of the CA2D model using the free field receptor.

The attenuation phenomena observed in the audio files can be attributed to certain design characteristics of the CA2D. The first characteristic is the proportional transmission coefficient α_{tp}, which controls the amount of sound pressure transmitted to neighboring cells in the non-preferential direction, remembering that the non-preferential direction is perpendicular to the direction of incidence of the acoustic wave. This was included to create a propagation preference in the acoustic wavefront. The second characteristic is the negative sign adopted in Equation 3.30 that affects the reflected wave. This negative sign is also found in research such as (Y. KAGAWA T. TSUCHIYA; TAKEUCHIT, 1998)(Y. KAGAWA T. TSUCHIYA; TAKEUCHIS, 1999) that discretizes Huygens principle to simulate wave propagation and was included in the CA2D to facilitate negative number propagation through the mesh. The third characteristic is the phase inversion that the CA2D imposes on sound waves incident and reflected on the walls, which was included to attenuate the pressure values reflected on the walls. These parameters are used to control the pressure value divergence within the CA2D meshes. Among these, only the reflection coefficient is derived from the DWG model.

4.6 Computational Performance Results

To calculate the acceleration factor, it is first necessary to determine the number of updates per second that the proposed CA2D model achieves in both software and hardware implementations.

For the software implementation, it is necessary to estimate the execution times for the 10×10, and 20×20 meshes. This estimation was performed from the average of twenty simulations conducted for each mesh size. In these simulations, the G major guitar chord signal was used as the source to create a one-second simulation, which required 6000 mesh

updates to be completed. Table 12 summarizes these software execution times and calculates the latency (execution time divided by the number of updates) corresponding to each mesh size.

Mesh Size	Mean execution time (s)	Time for one iteration (μs)
10×10	0.369	61.5
20×20	0.737	122.83

Table 12 – Software execution time and hardware latency for the meshes studied.

To estimate the number of updates per second, the 20×20 mesh was used, being the largest implemented case. This mesh has a latency of $\approx 123 \mu s$, representing the time it takes for the mesh to complete one update. Thus, the CA2D model in software achieves $\approx$ 8141.11 updates per second.

Conversely, the execution time in hardware was measured by the behavioral simulation of CA2D in the *Vivado* software. Since the CA2D model on FPGA updates all cells in parallel, the simulation time remains unaffected by an increase in mesh size, resulting in a constant total simulation time for the mesh sizes. Thus, to perform 6000 updates it was necessary 1.2003 ms (200 μs for each update: 190 ns for the mesh cells + 10 ns for asserting the *start* signal). It is important to highlight that each mesh cell required 19 clock cycles (190 ns) to update its outputs. Therefore, the CA2D model implemented in hardware achieves 5000000 mesh updates per second.

Finally, the acceleration factor is estimated by,

$$acceleration factor = \frac{5000000\ updates/second}{8141.11\ updates/second} \approx 614.17,$$

demonstrating that the computational performance of the FPGA implementation running at a clock frequency of 100MHz is approximately six hundred times greater than the software implementation running at 3.6 GHz.

5 Conclusions

This work presented the development of a model based on cellular automata, called CA2D, to simulate the behavior of acoustic waves in rooms. The model is based on a study of acoustic wave propagation using Digital Waveguides (DWG). CA2D model makes uses of simple arithmetic operators such as additions and multiplications, adopting local behavioral rules influenced by the sound pressure of acoustic waves. This avoids the complex mathematical solutions that ordinary differential equations (ODEs) or numerical methods such as the Finite Element Method (FEM) or the Boundary Element Method (BEM) use to approximate this type of behavior. Based on the results reported in (MOURA; MUÑOZ, 2021), (PEREIRA, 2022), one can conclude that the execution time of a one-dimensional CA for 1480 cells is around 5,45 seconds using a Intel core i7 processor. On the other hand in (ANAYA et al., 2025) a Finite Element of the 1480 nodes requires 14,48 seconds using the same CPU processor. This findings demonstrates the suitability of using cellular automatas for improving the execution time of room acoustic simulators.

The CA2D model has been successfully implemented on both the Pynq Z2 and the ZCU104 SoC-FPGAs development boards. These implementations required 31039 LUTs, 22121 FFs, and 126 DSPs for the 10×10 mesh on the Pynq Z2; 61289 LUTs, 39076 FFs, and 126 DSPs for the 10×10 mesh on the ZCU104; and 154784 LUTs, 110790 FFs, and 646 DSPs on the ZCU104 for the 20×20 mesh. These implementations achieve a parallelization of 96 cells in the 10×10 meshes and 396 cells in the 20×20 mesh. Additionally, it should be noted that the CA2D operated correctly at an operating frequency of 100MHz.

In free-field scenarios, the CA2D model successfully reproduces the behavior of acoustic waves, as evidenced by the receiver spectra being very similar to the spectrum of the source signal. However, the Power Spectral Density results reveal that strong attenuation phenomena significantly affect the signals at the receivers. These attenuation phenomena can result from the negative additions of the reflected wave component of the mesh cell equations (in this work a reflection coefficient of 0,5 was inherited from the DWG model), the proportional coefficient for the non-preferential direction, and phase inversion in the cell walls. These features of CA2D, which are not inherited from the DWG model, help to counteract the chaotic nature of cellular automata and prevent the pressure values within the mesh from diverging after a few interactions. This facilitates the transmission of a signal (periodic values over time with defined amplitude and frequency) within the mesh. As expected, this phenomenon increases with the size of the mesh, making the appearance of noise in the collected signals more noticeable.

In implementations with obstacles, CA2D successfully replicates the phenomena of

acoustic wave propagation when utilizing a 1KHz sinusoidal signal. However, it is noteworthy that there is a notable increase of noise. Conversely, when employing the G major guitar chord signal, it was not possible to obtain a conclusive result. The heightened noise levels can be attributed to attenuation phenomena within the mesh, which effectively diminish the intensity of the source signal. This result shows that the design parameters that allow the correct operation of the CA2D are very strict in the case of meshing with obstacles.

On the other hand, the parallel computational behavior of the cells in the hardware implementation ensures that both the update time of $1.2003ms$ and the throughput of 20 ns remain independent of the mesh size. In this manner, the CA2D model can run approximately 614,17 times faster on hardware than on software. However, it is important to note that CA2D's scalability is constrained by the resources available on the FPGA, with 20×20 being the largest mesh size implemented in this research.

It is important to note that the hardware architectures developed for the CA2D model allow the modeling of delays in the propagation of acoustic waves between adjacent cells. This aspect enables the simulation of wave propagation inside obstacles or in different types of media where the propagation velocity varies.

5.1 Future Works

1. To minimize the attenuation experienced by the CA2D model and to enhance the quality of the receiver signals, one possible solution is to optimize the transmission, proportional transmission, and reflection coefficients. For that, it is necessary to use a reference model, such as the DWG model, and a fitness function based on the error of the sound propagation in each cell.

2. More research is necessary to extend the rectangular two-dimensional CA2D model to a three-dimensional model, exploring cubic topology networks. In a similar way, new hardware architectures can be proposed to evaluate the quality of acoustic simulations through various two-dimensional mesh topologies, such as triangular and hexagonal, as well as three-dimensional topologies, such as pyramidal structures.

3. An important task is to incorporate mechanisms into the hardware architecture to emulate the behavior of acoustic waves in different propagation media. These mechanisms, which can be implemented using FIFOs or synchronous counters in the CA2D cell architectures, will enable to simulate the acoustic phenomena changing the wave propagation velocity.

4. Extend the CA2D model to more complex systems by increasing the number of on-board hardware cells. Two strategies can be employed for this purpose. The first strategy involves extending the staggered solution previously developed by (PEREIRA, 2022)

for a one-dimensional cellular automaton to two dimensions. The second strategy is to explore dynamic partial reconfiguration mechanisms on the FPGA.

References

ANAYA, S. G.; MOURA, H. G.; TEODORO, E. B.; MIRANDA, R. F. de; MUÑOZ, D. M. A comprehensive digital waveguide formulation using the impedance method for acoustic simulation. **Mechanical Systems and Signal Processing**, v. 224, p. 112047, 2025. ISSN 0888-3270. DOI: https://doi.org/10.1016/j.ymssp.2024.112047. Cit. on pp. 22, 85.

ARNELA, M.; DABBAGHCHIAN, S.; BLANDIN, R.; GUASCH, O.; ENGWALL, O.; VAN HIRTUM, A.; PELORSON, X. Influence of vocal tract geometry simplifications on the numerical simulation of vowel sounds. **Journal of the Acoustical Society of America**, v. 140, n. 3, p. 1707–1718, 2016. DOI: 10.1121/1.4962488. Cit. on p. 17.

BAKHTERI, R.; CHENG, J.; SEMMELHACK, A. Design and Implementation of Cellular Automata on FPGA for Hardware Acceleration. **Procedia Computer Science**, Elsevier B.V., v. 171, p. 1999–2007, 2020. ISSN 18770509. DOI: 10.1016/j.procs.2020.04.214. Cit. on pp. 32, 36.

BOURNEZ, O. Computability, complexity and programming with ordinary differential equations. In: SYMPOSIUM on Theoretical Aspects of Computer Science (STACS). Mar. 2020. DOI: 10.4230/LIPIcs.STACS.2020.3. Cit. on p. 17.

BRETON, V.; FONVIEILLE, H.; ', P. G.; GUICHENEY, C.; JOUSSET, J.; ROBLIN, Y.; TAMIN, F. Application of neural networks and cellular automata to interpretation of calorimeter data. **Nuclear Instruments and Methods in Physics Research A**, v. 362, p. 478–486, 1995. DOI: 10.1016/0168-9002(95)00217-0. Cit. on p. 30.

BRÜGGE, E. 2. 2.; VERLEYSEN, M. 1.; LOUVAIN, U. catholique de; LEUVEN, K. U.; SOCIETY, I. N. N.; INTELLIGENCE, C.; ARTIFICIAL NEURAL NETWORKS, M. L. 2. 2.-2. B. E. S. on; BRUGES, E. 2. 2.-2. **26th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning ESANN 2018 : Bruges, Belgium, April 25, 26, 27, 2018 : proceedings**. ISBN 9782875870476. Cit. on p. 30.

CHATZIIOANNOU, V.; VAN WALSTIJN, M. Energy conserving schemes for the simulation of musical instrument contact dynamics. **Journal of Sound and Vibration**, v. 339, p. 262–279, 2015. ISSN 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2014.11.017. Cit. on p. 17.

CHAUDHURI, P. P.; GHOSH, S.; DUTTA, A.; CHOUDHURY, S. P. **A new kind of computational biology: Cellular automata based models for genomics and proteomics**. Springer Singapore, Jan. 2018. P. 1–335. ISBN 9789811316395. DOI: 10.1007/978-981-13-1639-5. Cit. on p. 30.

CRESPO, M. L.; FOULON, F.; CICUTTIN, A.; BOGOVAC, M.; ONIME, C.; SISTERNA, C.; MELO, R.; FLORIAN SAMAYOA, W.; GARCÍA ORDÓÑEZ, L. G.; MOLINA, R.; VALINOTI, B. Remote Laboratory for E-Learning of Systems on Chip and Their Applications to Nuclear and Scientific Instrumentation. **Electronics**, v. 10, n. 18, 2021. ISSN 2079-9292. DOI: 10.3390/electronics10182191. Cit. on p. 59.

D'AMBROSIO, D.; RANGO, A. D.; OLIVERIO, M.; SPATARO, D.; SPATARO, W.; RONGO, R.; MENDICINO, G.; SENATORE, A. The Open Computing Abstraction Layer for Parallel Complex Systems Modeling on Many-Core Systems. **Journal of Parallel and Distributed Computing**, Academic Press Inc., v. 121, p. 53–70, Nov. 2018. ISSN 07437315. DOI: 10.1016/j.jpdc.2018.07.005. Cit. on p. 30.

DOURVAS, N. I.; SIRAKOULIS, G. C.; ADAMATZKY, A. I. Parallel Accelerated Virtual Physarum Lab Based on Cellular Automata Agents. **IEEE Access**, Institute of Electrical and Electronics Engineers Inc., v. 7, p. 98306–98318, 2019. ISSN 21693536. DOI: 10.1109/ACCESS.2019.2927815. Cit. on pp. 34, 36.

EVANGELISTA, G.; ECKERHOLM, F. Player–Instrument Interaction Models for Digital Waveguide Synthesis of Guitar: Touch and Collisions. **IEEE Transactions on Audio, Speech, and Language Processing**, v. 18, n. 4, p. 822–832, 2010. DOI: 10.1109/TASL.2009.2038822. Cit. on pp. 28, 30.

FISCHER, M.; GAUGER, U.; GAUL, L. A multipole Galerkin boundary element method for acoustics. **Engineering Analysis with Boundary Elements**, v. 28, n. 2, p. 155–162, 2004. ISSN 0955-7997. DOI: https://doi.org/10.1016/j.enganabound.2003.07.001. Cit. on p. 24.

GARDNER, M. **MATHEMATICAL GAMES The fantastic combinations of John Conway's new solitaire game "life"**. Accessed on December 10, 2024. 1970. Available from: <https://web.stanford.edu/class/sts145/Library/life.pdf>. Cit. on p. 30.

GERMAIN, F.; EVANGELISTA, G. Synthesis of guitar by digital waveguides: Modeling the plectrum in the physical interaction of the player with the instrument. In: 2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. 2009. P. 25–28. DOI: 10.1109/ASPAA.2009.5346502. Cit. on pp. 28, 30.

GULLY, A. J.; DAFFERN, H.; MURPHY, D. T. Diphthong Synthesis Using the Dynamic 3D Digital Waveguide Mesh. **IEEE/ACM Transactions on Audio, Speech, and Language Processing**, v. 26, n. 2, p. 243–255, 2018. DOI: 10.1109/TASLP.2017.2774921. Cit. on pp. 29, 30.

GUPTA, A.; GÖZEN, I.; TAYLOR, M. A cellular automaton for modeling non-trivial biomembrane ruptures. **Soft Matter**, Royal Society of Chemistry, v. 15, p. 4178–4186, 20 2019. ISSN 17446848. DOI: 10.1039/c8sm02032a. Cit. on pp. 34, 36.

HACIHABIBOGLU, H.; GUNEL, B.; KONDOZ, A. M. On the Accuracy of First-Order Numerical Derivatives in Multidimensional Digital Waveguide Mesh Topologies. **IEEE Signal Processing Letters**, v. 15, p. 9–12, 2008. DOI: 10.1109/LSP.2007.911162. Cit. on pp. 28, 30.

HACIHABIBOGLU, H.; GUNEL, B.; CVETKOVIC, Z. Simulation of Directional Microphones in Digital Waveguide Mesh-Based Models of Room Acoustics. **IEEE Transactions on Audio, Speech, and Language Processing**, v. 18, n. 2, p. 213–223, 2010. DOI: 10.1109/TASL.2009.2025100. Cit. on pp. 29, 30.

JIMÉNEZ-MORALES, F.; GUISADO, J. L.; GUERRA, J. M. Simulating Laser Dynamics with Cellular Automata. In: NONLINEAR Systems, Vol. 1: Mathematical Theory and Computational Methods. Springer Verlag, 2018. P. 405–422. DOI: 10.1007/978-3-319-66766-9_14. Cit. on pp. 34, 36.

JOHNSTON, N.; GREENE, D. **Conway's Game of Life : mathematics and construction**. P. 480. ISBN 9781794816961. Cit. on p. 30.

KUMARAVEL, A.; MEETEI, O. N. An Application of Non-uniform Cellular Automata for Efficient Cryptography. In: 2013 IEEE Conference on Information Communication Technologies. 2013. P. 1200–1205. DOI: 10.1109/CICT.2013.6558283. Cit. on p. 30.

LAMOUREUX, M. P. **The mathematics of PDEs and the wave equation**. 2006. Accessed on December 10, 2024. Available from: <https://mathtube.org/sites/default/files/lecture-notes/Lamoureux_Michael.pdf>. Cit. on p. 24.

LI, W.; CHAI, Y.; LEI, M.; LIU, G. Analysis of coupled structural-acoustic problems based on the smoothed finite element method (S-FEM). **Engineering Analysis with Boundary Elements**, v. 42, p. 84–91, 2014. Advances on Meshfree and other Mesh reduction methods. ISSN 0955-7997. DOI: https://doi.org/10.1016/j.enganabound.2013.08.009. Cit. on p. 24.

LIN, Z.; ZHAO, X. An improved approach to simulate seismic events based on cellular automata. **International Journal of Modern Physics C**, World Scientific Publishing Co. Pte Ltd, v. 31, 4 Apr. 2020. ISSN 01291831. DOI: 10.1142/S0129183120500539. Cit. on pp. 35, 36.

LIU, Y. On the BEM for acoustic wave problems. **Engineering Analysis with Boundary Elements**, v. 107, p. 53–62, 2019. ISSN 0955-7997. DOI: 10.1016/j.enganabound.2019.07.002. Cit. on p. 24.

LUO, K.; WANG, Z.; LEI, X. The cellular automata model of sound propagations and its application in structural noise calculations. **Applied Acoustics**, Elsevier BV, v. 182, p. 108262, Nov. 2021. ISSN 0003682X. DOI: 10.1016/j.apacoust.2021.108262. Cit. on pp. 35, 36.

MELO, R. A. **Core-ComBlock (GitLab)**. 2019. Accessed on December 10, 2024. Available from: <https://gitlab.com/rodrigomelo9/core-comblock>. Cit. on pp. 22, 59.

MILLER, D. A. Huygens's wave propagation principle corrected. **Optics letters**, Optica Publishing Group, v. 16, n. 18, p. 1370–1372, 1991. Accessed on December 10, 2024. Available from: <https://www-ee.stanford.edu/~dabm/146.pdf>. Cit. on p. 48.

MORDILLAT, P.; COLANGELI, C.; DI TOMMASO, F.; BRITTE, L. Sound Simulator for Hybrid Vehicle NVH Development. In. Cit. on p. 17.

MORSE, P. M. Vibration and Sound. **Nature**, v. 163, p. 232–232, 1949. DOI: 10.1038/163232a0. Cit. on p. 27.

MOURA, H. G.; MUÑOZ, D. M. Modeling wave propagation using cellular automata on Chip. In: 2021 34th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI). 2021. P. 1–6. DOI: 10.1109/SBCCI53441.2021.9529978. Cit. on pp. 35, 36, 85.

MOURA, H. G. de. **Implementação em chip de sistemas celulares autômatos dedicados à emulação da propagação de ondas acústicas em sistemas físicos**. 2022. Trabalho de Conclusão de Curso – Universidade de Brasilia. Cit. on p. 37.

MOURA, H. G. de. **Simulação Da Propagação De Ondas Acústicas Através De Uma Malha De Guias Digitais De Ondas**. 2006. Trabalho de Dissertação de Mestrado – Universidade Federal de Uberlândia. Cit. on pp. 27, 37.

MULLEN, J.; HOWARD, D. M.; MURPHY, D. T. Waveguide physical modeling of vocal tract acoustics: Flexible formant bandwidth control from increased model dimensionality. **IEEE Transactions on Audio, Speech and Language Processing**, v. 14, p. 964–971, 3 May 2006. ISSN 15587916. DOI: 10.1109/TSA.2005.858052. Cit. on pp. 28, 30.

MUÑOZ, D. M.; SANCHEZ, D. F.; LLANOS, C. H.; AYALA-RINCÓN, M. FPGA based floating-point library for CORDIC algorithms. In: 2010 VI Southern Programmable Logic Conference (SPL). 2010a. P. 55–60. DOI: 10.1109/SPL.2010.5483002. Cit. on p. 50.

MUÑOZ, D. M.; SANCHEZ, D. F.; LLANOS, C. H.; AYALA-RINCÓN, M. Tradeoff of FPGA Design of a Floating-point Library for Arithmetic Operators. **Journal Integrated Circuits and Systems**, v. 5, p. 42–52, 2010b. Accessed on December 10, 2024. Available from: <https://jics.org.br/ojs/index.php/JICS/article/view/309>. Cit. on p. 50.

MURPHY, D. T.; BEESON, M. The KW-boundary hybrid digital waveguide mesh for room acoustics applications. **IEEE Transactions on Audio, Speech and Language Processing**, v. 15, p. 552–564, 2 Feb. 2007. ISSN 15587916. DOI: 10.1109/TASL.2006.881681. Cit. on pp. 28, 30.

NEDJAH, N.; BEZERRA, H. D.; MOURELLE, L. M. Automatic generation of harmonious music using cellular automata based hardware design. **Integration**, Elsevier B.V., v. 62, p. 205–223, June 2018. ISSN 01679260. DOI: 10.1016/j.vlsi.2018.03.002. Cit. on p. 34.

NEUMANN, J. V.; BURKS, A. W. **Theory of Self-Reproducing Automata**. USA: University of Illinois Press, 1966. Cit. on p. 29.

OKUZONO, T.; YOSHIDA, T.; SAKAGAMI, K.; OTSURU, T. An explicit time-domain finite element method for room acoustics simulations: Comparison of the performance with implicit methods. **Applied Acoustics**, v. 104, p. 76–84, 2016. ISSN 0003-682X. DOI: https://doi.org/10.1016/j.apacoust.2015.10.027. Cit. on p. 24.

PEREIRA, G. **Modelagem e implementação de sistema celular autômato escalonado unidimensional**. 2022. Trabalho de Conclusão de Curso – Universidade de Brasilia. Cit. on pp. 22, 85, 86.

PEREIRA, G.; ANAYA, S.; MOURA, H.; MUÑOZ, D. FPGA Implementation of Staggered Cellular Automata for Wave Propagation Simulation. In: 2023 Southern Programmable Logic Conference, SPL. 2023. P. 7–12. Available from: <http://www.splconf.org/spl23/pdfs/Proceedings_SPL23.pdf>. Cit. on pp. 22, 35, 36.

PIND, F.; ENGSIG-KARUP, A. P.; JEONG, C.-H.; HESTHAVEN, J. S.; MEJLING, M. S.; STRØMANN-ANDERSEN, J. Time domain room acoustic simulations using the spectral element method. **Journal of the Acoustical Society of America**, v. 145, n. 6, p. 3299–3310, 2019. DOI: 10.1121/1.5109396. Cit. on p. 17.

RANGO, A. D.; FURNARI, L.; SENATORE, A.; D'AMBROSIO, D.; STRAFACE, S.; MENDICINO, G. Massive simulations on GPGPUs of subsurface flow on heterogeneous soils. In: 2021 29th Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP). 2021. P. 249–252. DOI: 10.1109/PDP52278.2021.00047. Cit. on pp. 35, 36.

ROND D'ALEMBERT, J. le. recherches sur la courbe que forme une corde tendue mise en vibration. **Histoire de l'académie royale des sciences et belles lettres de Berlin**, v. 3, p. 214–219, 1747. Cit. on p. 24.

SADD, M. H. Chapter 5 - Formulation and solution strategies. In: SADD, M. H. (Ed.). **Elasticity (Fourth Edition)**. Fourth Edition: Academic Press, 2021. P. 97–121. ISBN 978-0-12-815987-3. DOI: https://doi.org/10.1016/B978-0-12-815987-3.00005-0. Cit. on p. 18.

SHAFIEI, M.; KHAJI, N.; ESKANDARI-GHADI, M. An adaptive cellular automata approach with the use of radial basis functions for the simulation of elastic wave propagation. **Acta Mechanica**, Springer, v. 231, p. 2723–2740, 7 July 2020. ISSN 16196937. DOI: 10.1007/s00707-020-02669-9. Cit. on pp. 34–36.

SKEEL, R. D.; BERZINS, M. A Method for the Spatial Discretization of Parabolic Equations in One Space Variable. **SIAM Journal on Scientific and Statistical Computing**, v. 11, n. 1, p. 1–32, 1990. DOI: 10.1137/0911001. Cit. on p. 24.

SMITH III, J. O. Physical Modeling using Digital Waveguides. **Computer Music Journal special issue on Physical Modeling of Musical Instruments, Part I**, Winter, v. 16, p. 74491, 4 1992. Cit. on pp. 25, 26.

SOUTHERN, A.; LOKKI, T.; SAVIOJA, L.; MURPHY, D. The perceptual effects of dispersion error on room acoustic model auralization. English. In: FORUM Acusticum, Aalborg, Denmark, June 27 - July 1. 2011. P. 1553–1558. Cit. on p. 47.

SPEED, M.; MURPHY, D. T.; HOWARD, D. M. Three-Dimensional Digital Waveguide Mesh Simulation of Cylindrical Vocal Tract Analogs. **IEEE Transactions on Audio, Speech, and Language Processing**, v. 21, n. 2, p. 449–455, 2013. DOI: 10.1109/TASL.2012.2224342. Cit. on pp. 29, 30.

SUN, J.-H.; WU, T.-T. Propagation of acoustic waves in phononic-crystal plates and waveguides using a finite-difference time-domain method. **Physical Review B - Condensed Matter and Materials Physics**, v. 76, n. 10, 2007. DOI: 10.1103/PhysRevB.76.104304. Cit. on p. 24.

TANASYUK, Y.; PEREPELITSYN, A.; OSTAPOV, S. Parameterized FPGA-based implementation of cryptographic hash functions using cellular automata. In: 2018 IEEE International Conference on Dependable Systems, Services and Technologies. 2018. P. 225–228. DOI: 10.1109/DESSERT.2018.8409133. Cit. on pp. 34, 36.

THIES, H. **Uniform computational complexity of ordinary differential equations with applications to dynamical systems and exact real arithmetic**. 2018. Cit. on p. 17.

THOMAS, R. **Wayverb Realistic, fast impulse-response synthesis. GPU accelerated. Free and open source**. 2016. Accessed on Month Day, Year. Available from: <https://reuk.github.io/wayverb/introduction.html>. Cit. on p. 47.

TREFETHEN, L. N. **FINITE DIFFERENCE AND SPECTRAL METHODS FOR ORDINARY AND PARTIAL DIFFERENTIAL EQUATIONS**. Available from: <http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/pdetext.html.]>. Cit. on p. 24.

WOLFRAM, S. **A New Kind of Science**. Wolfram Media, 2002. ISBN 1579550088. Available from: <https://www.wolframscience.com>. Cit. on p. 32.

WOLFRAM, S. **Statistical mechanics of cellular automata**. Cit. on p. 31.

WOLFRAM, S. **UNIVERSALITY AND COMPLEXITY IN CELLULAR AUTOMATA**. 1984. P. 1–35. Cit. on p. 31.

WOODHOUSE, J.; POLITZER, D.; MANSOUR, H. Acoustics of the banjo: Measurements and sound synthesis. **Acta Acustica**, v. 5, n. 4, 2021. DOI: 10.1051/aacus/2021009. Cit. on p. 17.

Y. KAGAWA T. TSUCHIYA, K. F.; TAKEUCHIS, M. **DISCRETE HUYGENS' MODEL APPROACH TO SOUND WAVE PROPAGATION—REVERBERATION IN A ROOM, SOUND SOURCE IDENTIFICATION AND TOMOGRAPHY IN TIME REVERSAL**. v. 225. 1999. Available from: <http://www.idealibrary.comon>. Cit. on pp. 48, 83.

Y. KAGAWA T. TSUCHIYA, K. F.; TAKEUCHIT, M. **DISCRETE HUYGENS' MODEL APPROACH TO SOUND WAVE PROPAGATION**. v. 218. 1998. P. 419–444. Cit. on pp. 48, 49, 53, 83.

YU, D.; DU, D.; YANG, H.; TU, Y. Parallel computing simulation of electrical excitation and conduction in the 3D human heart. **2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2014**, v. 2014, p. 4315–9, Aug. 2014. DOI: 10.1109/EMBC.2014.6944579. Cit. on pp. 33, 36.

6 Resumo Estendido em Língua Portuguesa

Título:Cellular automata applied in linear acoustics and their implementation in reconfigurable hardware
Autor: Santiago Guzman Anaya
Orientador: Daniel Mauricio Muñoz Arboleda
Programa de Pós-Graduação em Sistemas Mecatrônicos
Brasília, 01 de dezembro de 2024

Palavras-chave: Autômatos celulares. Acustica linear. Hardware reconfigurável. Guias digitais de onda.

Introdução

No campo da acústica, um ramo da física relacionado ao estudo do comportamento das ondas sonoras, as simulações são comumente empregadas para aproximar a dinâmica das ondas sonoras sob várias condições. Essas simulações são importantes em áreas como acústica de salas (PIND et al., 2019), síntese de som (WOODHOUSE; POLITZER; MANSOUR, 2021), modelagem de instrumentos musicais (CHATZIIOANNOU; VAN WALSTIJN, 2015), estudo de cabines de veículos (MORDILLAT et al., 2021), entre outras, todas as quais podem ser aproximadas usando acústica linear. Essas simulações geralmente envolvem a solução de equações diferenciais parciais (PDEs) que governam o comportamento das ondas acústicas. A equação de onda (consulte Equação 6.1) é um dos principais modelos de PDE para acústica linear.

$$\nabla^2 P = \frac{1}{c_0^2}\frac{\partial^2 P}{\partial t^2} \tag{6.1}$$

onde, $\nabla^2 P$ é o operador Laplaciano, t, P e c_0 representam o tempo, a pressão e a velocidade de propagação da onda acústica, respectivamente.

A obtenção de soluções para esses PDEs geralmente requer muitos cálculos e, em determinadas circunstâncias, derivar uma solução analítica pode ser bastante desafiador. Consequentemente, os métodos numéricos são frequentemente empregados para aproximar as soluções dessas equações. Essas técnicas matemáticas iterativas discretizam um problema contínuo, dividindo-o em componentes mais simples, reduzindo assim a complexidade computacional. Os métodos numéricos mais comuns são o Método dos Elementos Finitos

(FEM), o Método dos Elementos de Contorno (BEM) e o Método das Diferenças Finitas (FDM), todos métodos versáteis capazes de aproximar soluções para qualquer PDE, mas que exigem a realização de cálculos de matrizes complexas. Por outro lado, para aproximar soluções para a equação de onda, há um método numérico no domínio do tempo conhecido como Digital Waveguides (DWG), que foi desenvolvido com base no conceito de dispersão de ondas sonoras para resolver problemas de propagação de ondas unidimensionais (III, n.d.). No entanto, por ter sido originalmente projetado para soluções unidimensionais, o DWG encontra dificuldades ao representar a propagação de ondas acústicas em problemas de dimensões maiores. Uma possível solução para essas limitações é o uso de Autômatos Celulares (CA), um modelo que emprega uma rede ordenada de elementos discretos (células), que são atualizados em etapas sucessivas de acordo com um conjunto de regras predefinidas que levam em conta o ambiente ao redor de cada célula (NEUMANN; BURKS, 1966). Essas regras, baseadas na compreensão física do fenômeno, dependem de operações de baixa complexidade que permitem o uso de aceleradores de hardware para melhorar o desempenho da simulação (DOURVAS; SIRAKOULIS; ADAMATZKY, 2019), (MOURA; MUÑOZ, 2021).

Materiais e Métodos

Guias de onda digitais

Esta pesquisa começa com o estudo dos fenômenos de transmissão e reflexão que ocorrem em uma sala acústica usando o modelo DWG. Como resultado desse estudo, é apresentada uma formulação passo a passo para o DWG, introduzindo um método que considera as alterações de impedância local para caracterizar com precisão os fenômenos de reflexão e transmissão da onda sonora. Matematicamente, essas alterações resultam em

$$\mathbf{p}_J(n) = \sum_{i=1}^{2N} a_i \mathbf{p}_i(n-1) - \left\{ \sum_{i=1}^{2N} a_i - 1 \right\} \mathbf{p}_J(n-2), \quad where \quad a_i = \frac{2R_i}{\sum_{i=1}^{2N} R_i}. \tag{6.2}$$

Uma desvantagem dos modelos DWGs é que o som só se propaga em uma direção preferencial entre as juntas de dispersão adjacentes. Para superar essa desvantagem, esta tese desenvolve um modelo de autômato celular bidimensional, chamado 2DCA, com base no estudo de DWGs e no princípio discreto de Huygens-Fresnel. Esse modelo propõe regras que usam operadores computacionais de baixo custo para emular o comportamento de propagação acústica.

Autômatos celulares

A arquitetura CA2D é embarcada em um acelerador de hardware usando FPGAs (Field Programmable Gate Arrays) e a linguagem de descrição de hardware VHDL. Os circuitos digitais foram desenvolvidos usando o conjunto de design Vivado 2018.3 e dispositivos FPGAs System-on-Chip (Soc) da AMD-Xilinx. Essa arquitetura consiste em três tipos de células:

- Célula de Parede: determina o comportamento das paredes em um organismo CA2D, ou seja, elas definem as condições de limite da rede. Essa célula foi projetada para emular condições de contorno perfeitamente reflexivas, o que significa que toda a pressão sonora que atua nas paredes é refletida, mas a magnitude e a direção são invertidas. Matematicamente, $P_{out} = -P_{in}$. A Figura 60b mostra as conexões da Célula de parede.

- Célula de Fonte: célula responsável por injetar os valores de pressão sonora na malha. Para fazer isso, essa célula roteia o valor da pressão para as saídas Norte (Pn_{out}), Sul (Ps_{out}), Leste (Pe_{out}) e Oeste (Pw_{out}). Essa célula controla o tempo das outras células no CA2D. Para fazer isso, ela usa um sinal *ready* que sincroniza a atualização da malha. Além disso, essa célula usa um contador síncrono para respeitar a latência de dezenove ciclos de clock da Mesh Cell. A Figura 60c mostra as conexões da célula de origem.

- Célula de Malha: célula usada para propagar a pressão sonora ao longo da malha. Essa célula tem quatro entradas, que são usadas por uma máquina de estado finito para calcular as quatro saídas. Essa máquina de estado usa dois somadores e multiplicadores IP-Cores de ponto flutuante (MUÑOZ et al., 2010) que são usados em diferentes estágios para realizar os cálculos, conforme mostrado na Figura 59. Essas quatro equações descrevem a propagação de ondas acústicas relacionando linearmente os coeficientes de reflexão (α_r), transmissão (α_t) e transmissão proporcional (α_{tp}) às quatro entradas da célula. Os dois últimos coeficientes são responsáveis pelo controle da energia transmitida tanto na direção preferencial de transmissão quanto na direção não preferencial (direção em que a onda de entrada não atinge). A Figura 60a mostra as conexões da Mesh Cell.

A integração desses três tipos de células em um organismo CA2D foi obtida com o uso de um gerador automático de código VHDL, chamado *vCA2Dgen*. Esse gerador de código facilita a conexão de todas as células dentro da rede e permite a criação de CA2Ds de diferentes tamanhos, possibilitando a alteração das posições das células de fonte e dos receptores, bem como a inserção de obstáculos na malha. Para implementar o CA2D nos FPGAs SoC Pynq-Z2 e ZCU 104, foi desenvolvida uma estrutura de hardware-in-the-loop (HIL) usando o ComBlock IP-core (MELO, 2019). Que cria uma interface de mapa de

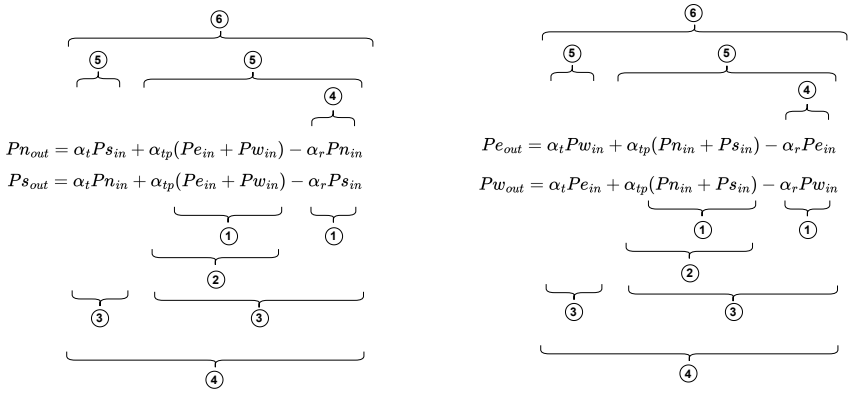
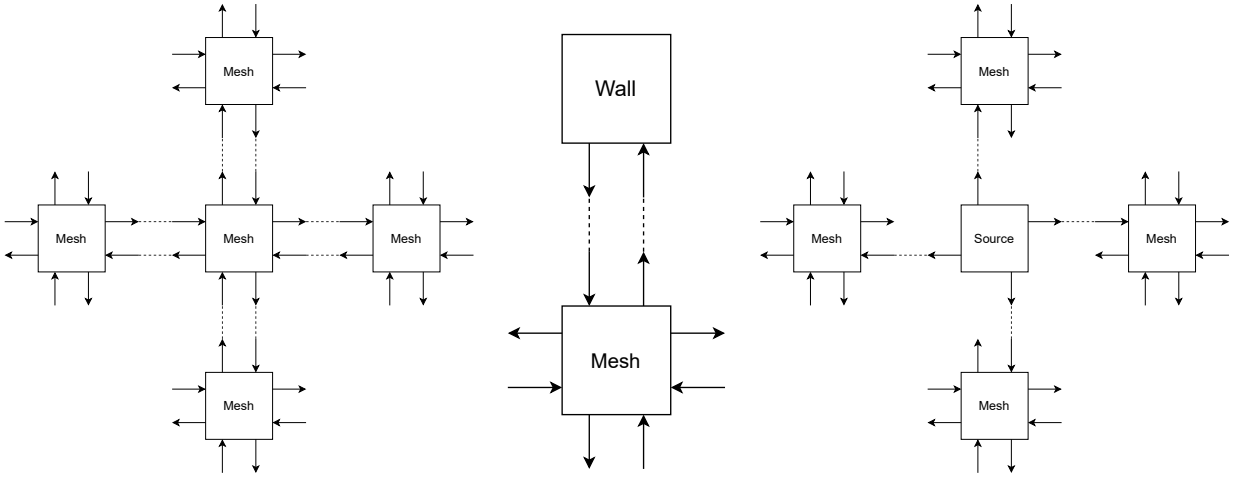


Figure 59 – Equações implementadas pela célula de malha. São mostrados os estágios em que essas equações são implementadas pela máquina de estado finito **??**

.



(a) Conexões da Célula de Malha (b) Conexões da Célula de Parede (c) Conexões da Célula de Fonte

Figure 60 – Conexões de vizinhança e fluxo de informações nas três células do CA2D.

memória que permite a comunicação entre o FPGA SoC e um computador por meio de uma conexão Ethernet. Um esquema do HIL é mostrado na Figura 61.

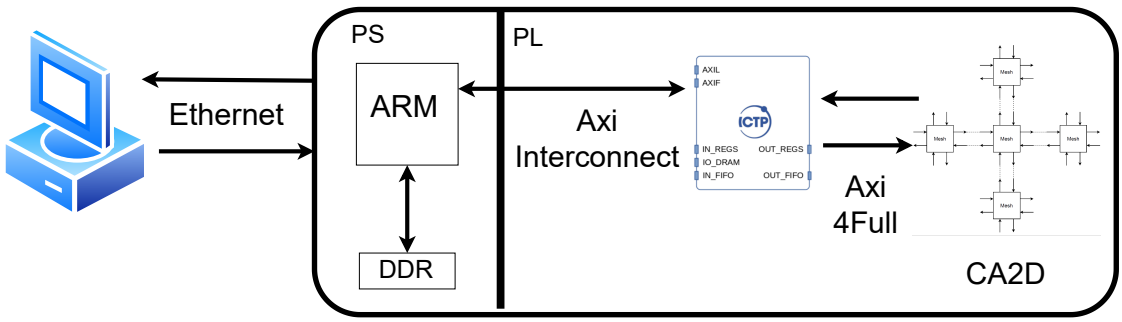


Figure 61 – Hardware-in-the-loop framework for the proposed CA2D hardware architecture.

Resultados e Discussões

The CA2D model was evaluated in both software and hardware. Two types of signals were used in this evaluation: a 1 KHz sinusoidal signal and a G major guitar chord signal. The system response was characterized using Fast Fourier Transform (FFT) and Power Spectral Density (PSD) as evaluation and comparison criteria. In addition, .WAV files were synthesized from both the software and hardware results.

Resultados do software

No software, uma malha de elementos de 64×64 foi implementada para os modelos DWG e CA2D usando a linguagem Python. Uma barreira e dois receptores foram inseridos nessa malha de forma que um receptor recebesse o estímulo no campo livre e o outro fosse bloqueado pela barreira. Os resultados coletados usando arquivos .WAV para os dois receptores usando os dois modelos estão resumidos neste vídeo [1].

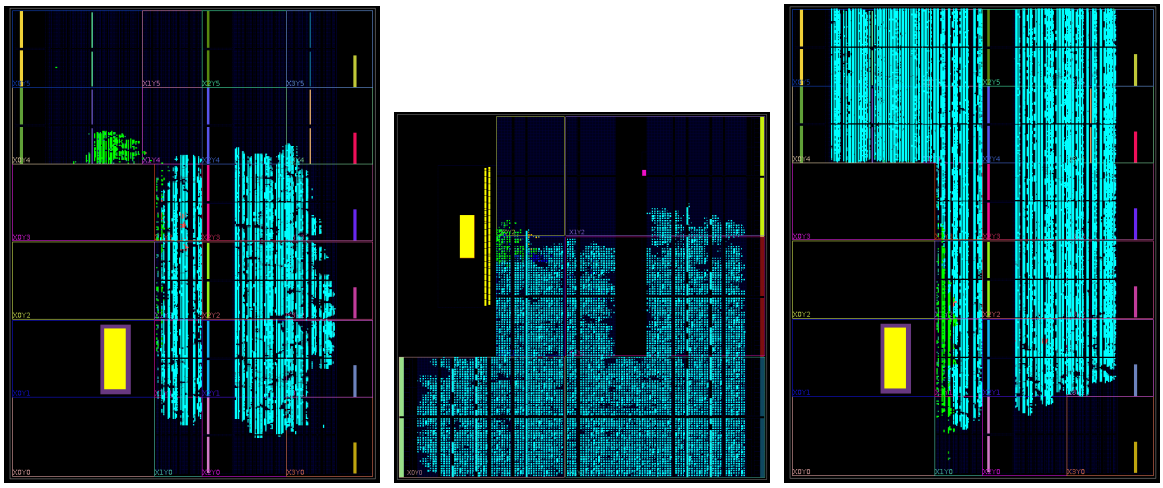
Resultados de hardware

Dois sistemas foram implementados em hardware: um com uma configuração de elementos de 10×10 e o outro com uma configuração de elementos de 20×20, com base nos recursos disponíveis nos SoC-FPGAs. Foram usados dois comprimentos de bits diferentes, começando com 27-bits (para a configuração de 10×10 no ZCU104) para avaliar o modelo com maior precisão numérica, e com 16 bits (para a configuração de 10×10 no PynqZ2 e a configuração de 20×20 no ZCU104). Essa última implementação visa reduzir o consumo de recursos e implementar sistemas maiores. Essas implementações exigiram 61289 LUTs (27,64%), 39076 FFs (8,48%) e 126 DSPs (7,29%) para a malha de 27-bits e 10×10 na ZCU104 (Figura 62a); 31039 LUTs (58,34%), 22121 FFs (20,79%) e 126 DSPs (57. 27%) para a malha de 16 bits e 10×10 no Pynq Z2 (Figura 62b); e 154784 LUTs (67,18%), 110790 FFs (24,04%) e 646 DSPs (37,38%) no ZCU104 para a malha de 16 bits e 20×20 (Figura 62c). Essas implementações atingem uma paralelização de 96 células nas malhas de 10×10 e 396 células na malha de 20×20.

O CA2D operou corretamente em uma frequência de 100 MHz. A tabela 13 mostra as estimativas de consumo de energia e a análise de tempo para as três implementações

Em cenários de campo livre, o modelo CA2D reproduz com sucesso o comportamento das ondas acústicas, conforme evidenciado pelo fato de que os espectros do receptor são muito semelhantes ao espectro do sinal da fonte (consulte as Figuras 63 64a 65a). No entanto, os resultados da densidade espectral de potência mostram que os fenômenos de atenuação

[1] Link vídeo: https://youtu.be/A2rkO4l$_GKg$



(a) Mesh 10 × 10 27-bits (b) Mesh 10 × 10 16-bits (c) Mesh 20 × 20 16-bits

Figure 62 – Lyouts das malhas CA2D implementadas

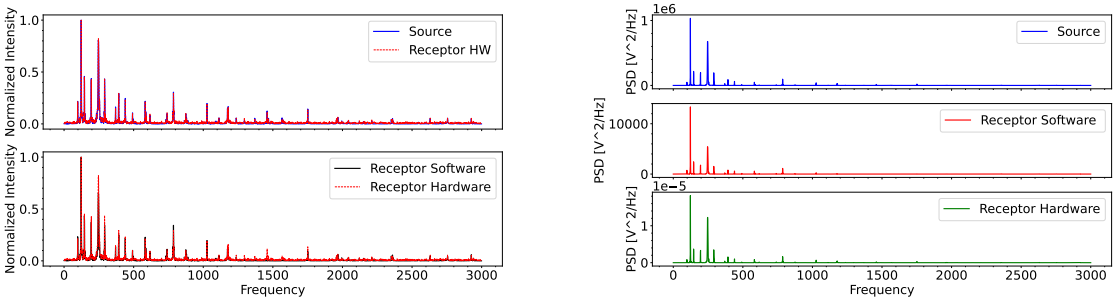
Tamanho da malha	potência estática (W)	potência dinâmica (W)	potência total (W)	Setup (ns)	Hold (ns)	Pulse Width (ns)
10 × 10 (ZCU104)	0.693	2.928	3.321	3.433	0.010	3.828
10 × 10 (Pynq Z2)	0.142	1.588	1.731	0.405	0.016	4.020
20 × 20 (ZCU104)	0.697	3.467	4.164	3.271	0.010	3.828

Table 13 – Estimativa de potência e análise de tempo para as três implementações do CA2D

forte afetam significativamente os sinais nos receptores (consulte as Figuras 63b 64b 65b). Esses fenômenos de atenuação podem resultar de adições negativas do componente da onda refletida nas equações da Mesh Cell (neste trabalho, um coeficiente de reflexão de 0,5 foi herdado do modelo DWG), do coeficiente proporcional para a direção não preferencial e da inversão de fase na Célula da parede. Esses fenômenos de atenuação ficam evidentes com o aumento do tamanho da malha, tornando mais perceptível o aparecimento de ruído nos sinais coletados. Quando foram introduzidas barreiras dentro da malha, não foi possível obter um resultado conclusivo, pois a atenuação aumentou ainda mais, exacerbando os níveis de ruído nos sinais coletados. Os recursos do CA2D ajudam a neutralizar a natureza caótica dos autômatos celulares e evitam que os valores de pressão dentro da malha divirjam após algumas interações. Os arquivos .WAV obtidos para as três implementações, usando o sinal de acorde de guitarra em sol maior como fonte, são mostrados na Tabela 11.

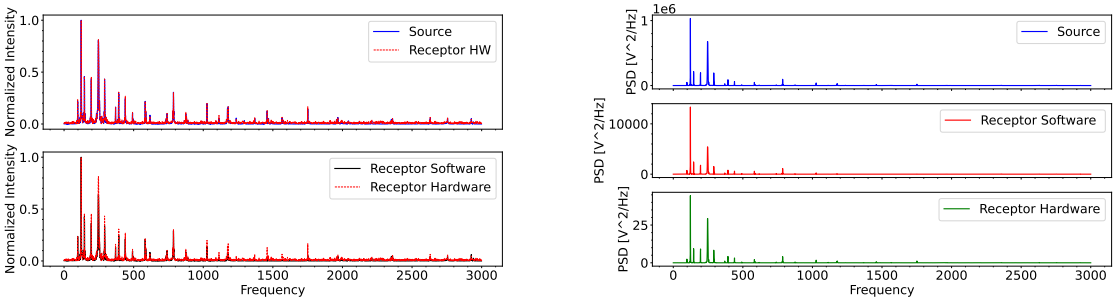
Tamanho da malha, bit-width	Link
10 × 10, 27-bits	https://youtu.be/alDDvRdkU0o
10 × 10, 16-bits	https://youtu.be/D36LPwW7fEs
20 × 20, 16-bits	https://youtu.be/pj1XPQvHY9k

Table 14 – Link with the hearing validation of the CA2D model using the free field receptor.



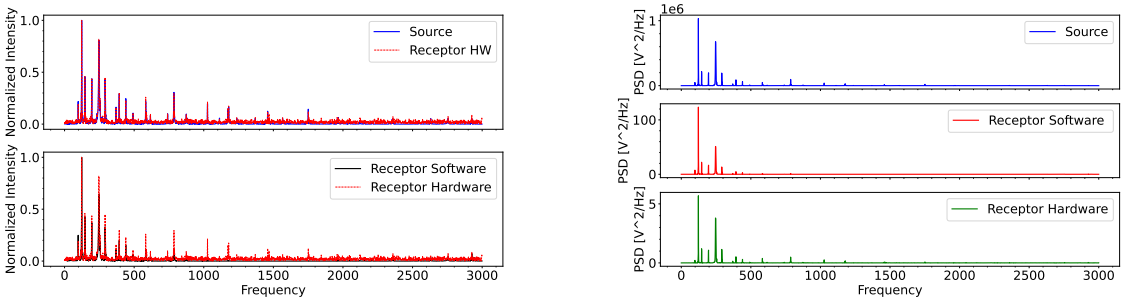
(a) Frecuency results(FFT comparison) (b) Power Spectral Density

Figure 63 – Hardware implementation results of the CA2D using 27-bits and 10×10 mesh for the major chord signal. Right: FFT comparison using a sinusoidal signal. Left: PSD of the source, receptor in software and receptor in hardware.



(a) Frecuency results(FFT comparison) (b) Power Spectral Density

Figure 64 – Hardware implementation results of the CA2D using 16 bits and 10×10 mesh for the G major chord signal. Right: FFT comparison using a sinusoidal signal. Left: PSD of the source, receptor in software and receptor in hardware.



(a) Frecuency results(FFT comparison) (b) Power Spectral Density

Figure 65 – Hardware implementation results of the CA2D using 16 bits and 20×20 mesh for the G major chord signal. Right: FFT comparison using a sinusoidal signal. Left: PSD of the source, receptor in software and receptor in hardware.

Resultados de Desempenho Computacional

A eficiência computacional do modelo CA2D foi avaliada tanto no hardware quanto no software. Para realizar essa avaliação, foi essencial determinar o número de atualizações por segundo obtidas pelo modelo CA2D proposto. Essas estimativas foram realizadas para a malha de 20×20 usando o sinal de acorde de guitarra em sol maior.

No software, o tempo de execução foi estimado pela média dos resultados de vinte simulações. Essa malha apresentou uma latência de $\approx 123\mu s$, atingindo $\approx$ 8.141,11 atualizações por segundo.

No hardware, o tempo de execução foi estimado pela simulação da implementação de hardware do CA2D no Vivado e confirmado pela medição da latência usando o núcleo IP do Integrated Logic Analyzer (ILA) da AMD-Xilinx.

Como o modelo CA2D no FPGA atualiza todas as células em paralelo, o tempo de simulação não é afetado por um aumento no tamanho da malha, resultando em um tempo total de simulação constante para os tamanhos de malha implementados. Para simular o sinal do acorde do violão, foram necessárias 6.000 atualizações, as quais foram executadas em 1.2003 ms. Como cada atualização de malha requer 20 ciclos de relógio (200 ns), a implementação de hardware do modelo CA2D alcança 5.000.000 atualizações de malha por segundo no hardware. Assim, um fator de aceleração de 5000000 (atualizações/segundo)/8141,11 (atualizações/segundo) $\approx$ 614,17. Esses resultados demonstram que o desempenho computacional do modelo CA2D no FPGA executado a uma frequência de clock de 100 MHz é seiscentos vezes maior do que a implementação do software em um Intel Core i7 7700 executado a 3,6 GHz.

Conclusão

- Neste trabalho, o modelo CA2D proposto foi efetivamente implementado usando FPGAs SoC para representações aritméticas de ponto flutuante de 16- e 27-bits, permitindo que as simulações acústicas lineares fossem aceleradas explorando o paralelismo intrínseco dos autômatos celulares

- Os resultados da validação foram realizados com configurações de malha de 10×10 e 20×20, incluindo barreiras e um sinal de acorde de guitarra na célula de origem. O espectro FFT obtido indica que o sinal nos receptores é semelhante ao sinal de origem. No entanto, os resultados do espectro PSD demonstram uma forte atenuação que pode ser explicada pelas adições negativas do componente da onda refletida nas Células de malha, pelo coeficiente proporcional para a direção não preferencial e pela inversão de fase nas Células de parede.

- modelo CA2D é capaz de ser executado aproximadamente 600 vezes mais rápido em hardware do que em software. No entanto, é importante observar que a escalabilidade do CA2D é limitada pelos recursos disponíveis no FPGA, sendo 20×20 o maior tamanho de malha implementado nesta pesquisa.

Referências

CHATZIIOANNOU, V.; VAN WALSTIJN, M. Energy conserving schemes for the simulation of musical instrument contact dynamics. **Journal of Sound and Vibration**, v. 339, p. 262–279, 2015. ISSN 0022-460X. DOI: https://doi.org/10.1016/j.jsv.2014.11.017. Cit. on p. 95.

DOURVAS, N. I.; SIRAKOULIS, G. C.; ADAMATZKY, A. I. Parallel Accelerated Virtual Physarum Lab Based on Cellular Automata Agents. **IEEE Access**, Institute of Electrical and Electronics Engineers Inc., v. 7, p. 98306–98318, 2019. ISSN 21693536. DOI: 10.1109/ACCESS.2019.2927815. Cit. on p. 96.

III, S. A. V. D. J. O. S. **Physical Modeling with the 2-D Digital Waveguide Mesh**. Cit. on p. 96.

MELO, R. A. **Core-ComBlock (GitLab)**. 2019. Accessed on December 10, 2024. Available from: <https://gitlab.com/rodrigomelo9/core-comblock>. Cit. on p. 97.

MORDILLAT, P.; COLANGELI, C.; DI TOMMASO, F.; BRITTE, L. Sound Simulator for Hybrid Vehicle NVH Development. In. Cit. on p. 95.

MOURA, H. G.; MUÑOZ, D. M. Modeling wave propagation using cellular automata on Chip. In: 2021 34th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI). 2021. P. 1–6. DOI: 10.1109/SBCCI53441.2021.9529978. Cit. on p. 96.

MUÑOZ, D. M.; SANCHEZ, D. F.; LLANOS, C. H.; AYALA-RINCÓN, M. Tradeoff of FPGA Design of a Floating-point Library for Arithmetic Operators. **Journal Integrated Circuits and Systems**, v. 5, p. 42–52, 2010. Accessed on December 10, 2024. Available from: <https://jics.org.br/ojs/index.php/JICS/article/view/309>. Cit. on p. 97.

NEUMANN, J. V.; BURKS, A. W. **Theory of Self-Reproducing Automata**. USA: University of Illinois Press, 1966. Cit. on p. 96.

PIND, F.; ENGSIG-KARUP, A. P.; JEONG, C.-H.; HESTHAVEN, J. S.; MEJLING, M. S.; STRØMANN-ANDERSEN, J. Time domain room acoustic simulations using the spectral element method. **Journal of the Acoustical Society of America**, v. 145, n. 6, p. 3299–3310, 2019. DOI: 10.1121/1.5109396. Cit. on p. 95.

WOODHOUSE, J.; POLITZER, D.; MANSOUR, H. Acoustics of the banjo: Measurements and sound synthesis. **Acta Acustica**, v. 5, n. 4, 2021. DOI: 10.1051/aacus/2021009. Cit. on p. 95.