Florida International University
FIU Digital Commons

FIU Electronic Theses and Dissertations | University Graduate School

11-12-2019

Continuous-time Algorithms and Analog Integrated Circuits for Solving Partial Differential Equations

Nilan Udayanga Galabada Kankanamge
Florida International University, ngala010@fiu.edu

Follow this and additional works at: https://digitalcommons.fiu.edu/etd

Part of the Electrical and Electronics Commons, Signal Processing Commons, and the VLSI and Circuits, Embedded and Hardware Systems Commons

Recommended Citation

Galabada Kankanamge, Nilan Udayanga, "Continuous-time Algorithms and Analog Integrated Circuits for Solving Partial Differential Equations" (2019). *FIU Electronic Theses and Dissertations*. 4327.
https://digitalcommons.fiu.edu/etd/4327

This work is brought to you for free and open access by the University Graduate School at FIU Digital Commons. It has been accepted for inclusion in FIU Electronic Theses and Dissertations by an authorized administrator of FIU Digital Commons. For more information, please contact dcc@fiu.edu.

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

CONTINUOUS-TIME ALGORITHMS AND ANALOG INTEGRATED CIRCUITS FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL AND COMPUTER ENGINEERING

by

Nilan Udayanga Galabada Kankanamge

2019

To: Dean John Leonidas Volakis
College of Engineering and Computing

This dissertation, written by Nilan Udayanga Galabada Kankanamge, and entitled Continuous-time Algorithms and Analog Integrated Circuits for Solving Partial Differential Equations, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Shubhendu Bhardwaj

Wujie Wen

Watson Lees

S.I. Hariharan

Arjuna Madanayake, Major Professor

Date of Defense: November 12, 2019

The dissertation of Nilan Udayanga Galabada Kankanamge is approved.

Dean John Leonidas Volakis
College of Engineering and Computing

Andrés G. Gil
Vice President for Research and Economic Development
and Dean of the University Graduate School

Florida International University, 2019

© Copyright 2019 by Nilan Udayanga Galabada Kankanamge

All rights reserved.

DEDICATION

I dedicate this dissertation to my family and my wife for their constant support and unconditional love.

ACKNOWLEDGMENTS

First, I would like to extend my heartfelt gratitude to my advisor, Dr. Arjuna Madanayake, for his continuous guidance, motivation, and scientific advice throughout my Ph.D. program. His support and patience empowered me to face hardships throughout this journey which would not have been possible if not for him. I would also like to express my deepest appreciation for Dr. S. I. Hariharan, who always supported and nurtured me during my Ph.D. research. He has provided me extensive personal and professional guidance and taught me a great deal about both scientific research and life in general. I would like to extend my sincere thanks to the rest of my committee members, Dr. Shubhendu Bhardwaj, Dr. Wujie Wen, and Dr. Watson Lees, for their great support and invaluable advice during this dissertation.

I would also like to extend my deepest gratitude to Dr. Soumyajit Mandal (Case Western Reserve University), Dr. Leo Belostotski (University of Calgary), and Dr. Len Bruton (University of Calgary) for sharing their knowledge and expertise to overcome scientific challenges throughout my dissertation. I very much appreciate their invaluable contributions and insightful suggestions. I also had the great pleasure of working with Jifu Liang (Case Western Reserve University), who always supported and extended a great amount of assistance for my Ph.D. research.

I am forever grateful to my parents Sumanawathie and Amarapala and my brothers Nuwan Janaranga and Asiri Sandaruwan and all my relatives for their endless affection. I would also like to extend my deepest gratitude to my loving and supportive wife, Dilani Nawagamuwa, who provides unending inspiration. This dissertation would not have been possible without their warm love, continued patience, and endless support.

I sincerely acknowledge the financial support from the Defense Advanced Research Projects Agency (DARPA) that funded my Ph.D. research and studies (sub-award via Ocius Technologies). I would like to thank Ocius Technologies for their support and profound belief in my work. Especially, I am extremely grateful to Dr. Dale Mugler (Ocius Technologies) for his encouragement, patient and valuable advice throughout my dissertation.

I would also like to thank my colleagues at the University of Akron, Florida International University, and fellow members of my research group, especially Chamith, Nilanka, Sewwandi, Arindam, Tharindu, Sunera, Prasad, Viduneth, Vishwa, Suranga, Gihan, Sravan, Najath, Udara and Hasantha for all their support. It was a complete pleasure working with you.

I would like to extend my sincere thanks to all the faculty members at Florida International University, University of Akron, and the University of Moratuwa for uplifting my knowledge and skills via graduate and undergraduate courses and other workshops. I would be failing in my duties if I don't mention all my teachers at Matugama C.W.W. Kannangara College, Udawela Sri Priyarathana College and Debarawewa President College for the solid foundation laid at the school level.

Last but not least, I would like to express my deepest gratitude to my friends, who were of great support in providing a happy distraction to rest my mind outside of my research.

ABSTRACT OF THE DISSERTATION

CONTINUOUS-TIME ALGORITHMS AND ANALOG INTEGRATED CIRCUITS FOR SOLVING PARTIAL DIFFERENTIAL EQUATIONS

by

Nilan Udayanga Galabada Kankanamge

Florida International University, 2019

Miami, Florida

Professor Arjuna Madanayake, Major Professor

Analog computing (AC) was the predominant form of computing up to the end of World War II. The invention of digital computers (DCs) followed by developments in transistors and thereafter integrated circuits (IC), has led to exponential growth in DCs over the last few decades, making ACs a largely forgotten concept. However, as described by the impending slow-down of Moores law, the performance of DCs is no longer improving exponentially, as DCs are approaching clock speed, power dissipation, and transistor density limits. This research explores the possibility of employing AC concepts, albeit using modern IC technologies at radio frequency (RF) bandwidths, to obtain additional performance from existing IC platforms. Combining analog circuits with modern digital processors to perform arithmetic operations would make the computation potentially faster and more energy-efficient. Two AC techniques are explored for computing the approximate solutions of linear and non-linear partial differential equations (PDEs), and they were verified by designing ACs for solving Maxwell's and wave equations. The designs were simulated in Cadence Spectre for different boundary conditions. The accuracies of the ACs were compared with finite-defference time-domain (FDTD) reference techniques.

The objective of this dissertation is to design software-defined ACs with complementary digital logic to perform approximate computations at speeds that are

several orders of magnitude greater than competing methods. ACs trade accuracy of the computation for reduced power and increased throughput. Recent examples of ACs are nearly accurate but have less than 25 kHz of analog bandwidth ($F_{compute}$) for continuous-time (CT) operations. In this dissertation, a special-purpose AC, which has $F_{compute}$ = 30 MHz (an equivalent update rate of 625 MHz) at a power consumption of 200 mW, is presented. The proposed AC employes 180 nm CMOS technology and evaluates the approximate CT solution of the 1-D wave equation in space and time. The AC is 100×, 26×, 2.8× faster when compared to the MATLAB- and C-based FDTD solvers running on a computer, and systolic digital implementation of FDTD on a Xilinx RF-SoC ZCU1275 at 900 mW (×15 improvement in power-normalized performance compared to RF-SoC), respectively.

TABLE OF CONTENTS

CHAPTER PAGE

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Analog computers (ACs) were the primary method of computation from the 1930s to the 1950s. As an example, Professor Vannevar Bush's differential analyzer (mechanical wheels and disk) [2] was used 24 hours a day during World war II [3]. During this time, mechanical ACs were used to solve many problems in different laboratories at the Massachusetts Institute of Technology (MIT). With the invention of the operational amplifier (op-amp), there was a huge interest in electronic AC. One of the first major successes of such op-amp-based computers was the M9 gun director developed at Bell Labs [4], and improved versions of these computers remained popular until the 1970s. However, the invention of digital computers (DCs) followed by transistor scaling surpassed ACs over the last few decades. According to Moore's law, the computing power and speed of DCs were doubled every two years. However, DCs are no longer improving exponentially as they are approaching clock speed, power dissipation, and transistor density limits. Fig. 1.1 shows how the expected number of transistors from Moore's law and the actual transistor count (of various state-of-the-art DCs) have varied over the last 15 years (at the end of 2005, 2010, 2015, and 2018). The pink colorbar shows the predicted value of Moore's law. The blue and green colorbars represent central processor units (CPUs) and graphics processor units (GPUs), respectively. It is clear that starting around 2015, Moore's law prediction surpassed the number of transistors in state-of-the-art DCs.

However, analog circuits realized in mainstream complementary metal-oxide-semiconductor (CMOS) technologies have shown tremendous performance improvement in recent years, which has not been sufficiently explored for computations. For example, today's radio-frequency (RF) transistors offer a unity current gain frequency greater than 500 GHz [5, 6]. Thus, analog and analog-digital hybrid

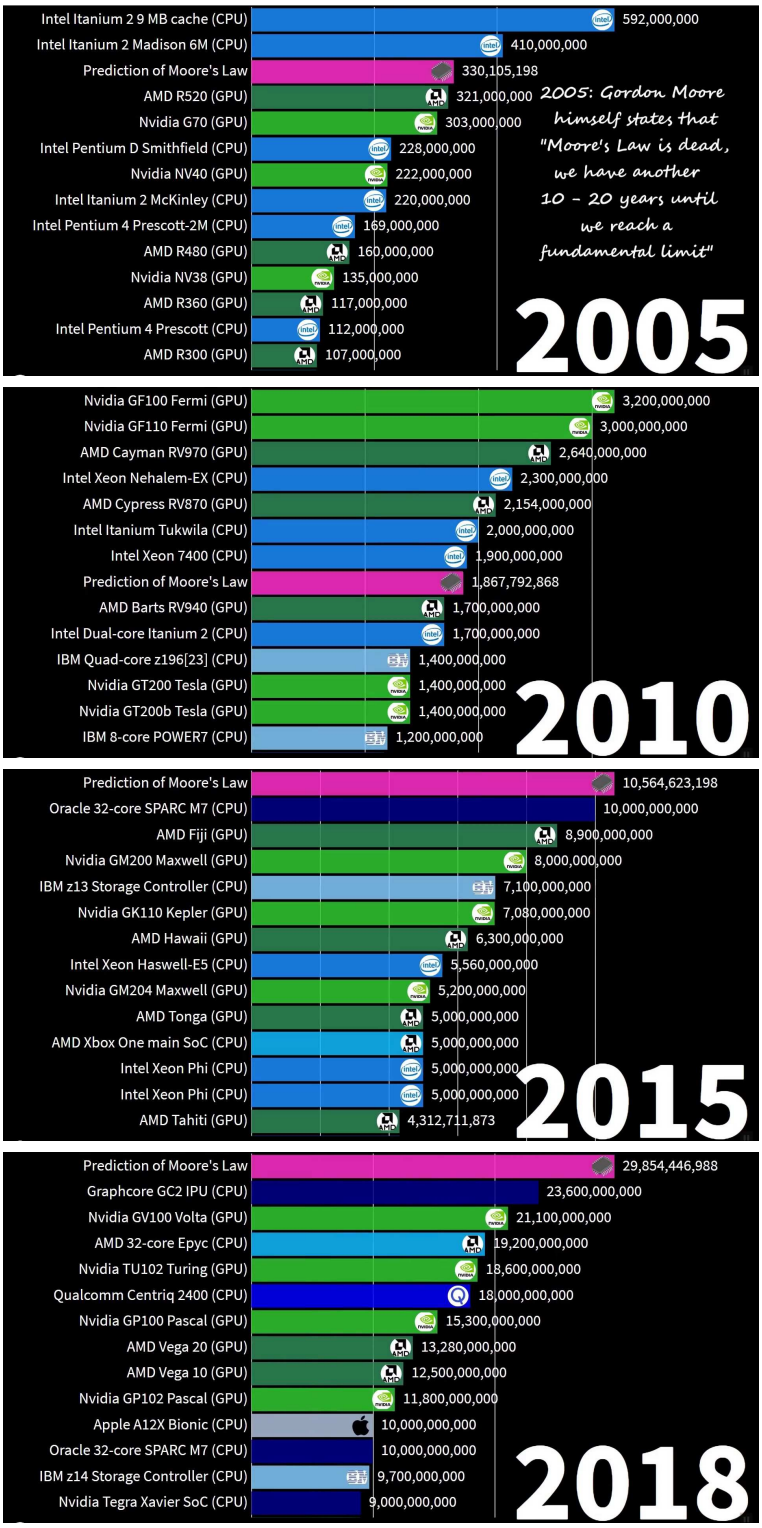

Figure 1.1: The variation of the expected number of transistors from Moore's law and the actual transistor count in the last 15 years (images from: [1])

computing platforms realized using scaled CMOS are becoming attractive for complex computations and simulations [7–18]. Columbia University's prototype analog accelerator is a recent example of an energy-efficient hybrid computer (HC) that accelerates ordinary differential equation (ODE)- and partial differential equation (PDE)-based computations [7–12, 14].

The research described in this dissertation takes a significant step forward towards designing and developing ACs that can perform physics-based computations at speeds that are 1-3 orders of magnitude greater than the speeds of the available ACs and DCs. For a given power and area budget, the analog bandwidth of our ACs is 1-3 orders of magnitude higher than the existing ACs. The proposed AC is equipped with modern digital processors (as supporting instrumentation) to perform PDE-based computations. Combining analog circuits with existing digital systems to perform arithmetic operations would make the computation potentially faster and more energy-efficient [7–12, 14]. In a general HC, the digital component serves as the controller (or the programmer) and performs basic logical and numerical operations, whereas the analog component performs more complex computations such as solving PDEs (electromagnetic, fluid dynamics, and plasmas) [19–23] and executing higher-dimensional matrix operations (inversion and multiplication) [24, 25].

Fig. 1.2 shows an overview of a hybrid computing platform. In general, ACs are exceedingly fast, since they can perform mathematical operations at the rate at which a signal transverses the analog circuit, which is an appreciable fraction of speed of light [26, 27]. Also, ACs can perform most power-intensive computations with far less energy as compared to DCs [7–11, 13]. However, the solutions of ACs are approximations to accurate solutions. In contrast, DCs are accurate, offer straightforward programmability, simple algorithmic operations, and ease of storage (memory) [7]. Thus, the hybrid architecture combines the best features of analog

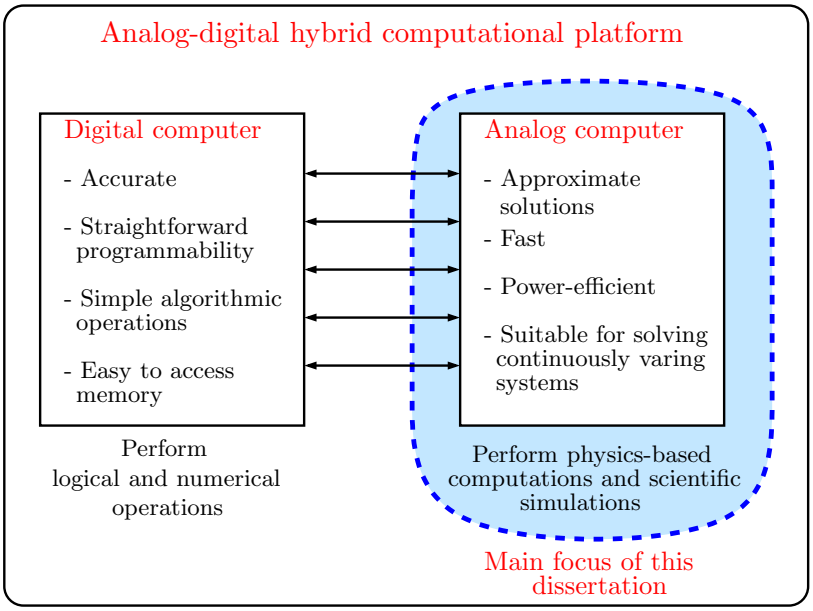

Figure 1.2: Overview of the envisioned hybrid accelerator.

and digital technologies to effectively perform complex computations. Here, the effectiveness of the computation is measured in terms of the computational speed, power, and area consumption. This dissertation mainly focuses on the analog subsystem of the HC and endeavors to bring back a modern version of an AC.

To understand how an HC can accelerate complex computations, consider the following example. In general, complex problems find their solutions using iterative methods, which require a large number of iterations to compute an accurate solution. One practical example is the solution of a nonlinear system of equations [16]. The selection of the initial guess is more critical in such a system–not only to reduce the number of iterations required but also to achieve convergence. The required number of iterations mainly depends on i) how good the initial guess (numerical seed) is and ii) how much accuracy is required [26]. Thus, the analog component of the HC can be used to obtain a good initial guess (in a very short time). The digital system is then fed by the resulted seed (initial guess) to compute the solution iteratively

and achieve the desired accuracy. Because of the good initial guess, the required number of iterations to achieve the desired accuracy will be reduced, which in turn reduces the total computation time and power consumption. An increase in speed and a 10-fold energy savings can be achieved with the hybrid systems compared to the standard DC systems [7, 9, 10, 12].

A given computational problem has features that lead to different computing demands. Some of these demands are not met by the capabilities of analog computers and others are not met by the structure of digital computers. Thus, the proper selection of a computational problem that can be accelerated using ACs is very important in designing an efficient hybrid computational platform. In general, analog computing can accelerate certain special classes of computational problems that are defined by continuously varying continuous-time (CT) systems [7, 9, 10, 13, 28–31]. Here, we selected *multidimensional (MD) PDEs* to solve using the proposed ACs, not only because PDEs are defined by continuously varying CT physical systems but also they are complex enough to accelerate using ACs. As an example, consider Maxwell's equations, which are a set of PDEs that form the foundation for electromagnetics. Modern computational simulators are dealing with a discretized version of Maxwell's equations, such as the finite-difference time-domain (FDTD) method [32–34]. However, Maxwell's equations are CT in nature, and inherently suitable for ACs. An AC that solves Maxwell's equations may yield approximate results compared to a digital FDTD-based solver, but with higher throughput.

Almost all physical systems that are described by theories of physics (or physical laws) are expressed using PDEs [19–23,35–37]. Examples of such systems have appilcations in fluid dynamics, thermodynamics, electromagnetics, quantum mechanics, magneto-hydrodynamics, and other areas [20–23, 36–39]. The dependent variables of the PDEs represent physical quantities such as temperature, pressure, electric

field, magnetic field, and fluid velocity, whereas the independent variables are space (three dimensions defined as x, y, and z) and time. Thus, the solutions of the PDEs describe the variation of one or more continuously varying physical quantities with space and time. Also, they must be defined at each space-time point. However, finding a continuously varying continuous space-time solution (analytic or closed-form solution) to a complex physical system having different boundary conditions is usually a difficult, time-consuming, and a computationally expensive task. In some cases, it is impossible to find an analytical solution. Moreover, for the physical systems that are described by coupled nonlinear PDEs, only numerical solutions are plausible [39].

In general, discrete-domain numerical methods running on powerful computers have been used to solve PDE-based problems. Examples of such methods include finite difference methods [32–34,40–42], the method of moments [43,44], finite volume methods [45,46], finite element methods [47,48], and spectral methods [49,50]. These methods discretize the spatial domain into different spatial grids. A suitable numerical integration (time-stepping) technique such as a Runge-Kutta method [51, 52], leapfrog method [40], or composition and splitting method [51, 53] is then applied along the time dimension to compute the solution over the temporal grid. Thus, all these methods discretize the PDEs in both time and space dimensions to enable computation using digital computers. As a result, the solutions are approximations to the exact, continuous-valued space-time solutions. In addition, implementations of fully discrete PDE solvers on high-speed digital processors, such as graphics processing units (GPUs), take many clock cycles to compute a single temporal frame of the update equation and thus have relatively low equivalent bandwidths. The proposed approach directly implements temporal recursions in continuous-time by using analog circuits. Such circuits can have bandwidths that greatly exceed the

equivalent bandwidths of GPUs. We expect our approach to provide a path beyond Moore's law scaling.

1.1 Research Questions

Due to the limitations in DCs, it is unlikely that a better solution for increasing the computational speed would be available by only using digital techniques. Thus, many research programs explore new material substrates for accelerating and solving difficult computation problems [54–56]. In this dissertation, we investigate the possibility of employing modern analog CMOS circuits and technologies (existing and well-established) for CT computations with significant advantages compared to existing DCs. The following research questions are addressed during the dissertation.

- Can we combine low-power analog circuits with modern digital systems to accelerate physics-based computations and simulations?
- Can we address the challenges in existing analog computing methods such that the proposed ACs have a significant acceleration for CT computations?
- How much speed-up we can gain from the ACs when compared to modern digital computing hardware?
- How much power reduction we can experience from the ACs when compared to modern DCs?
- How much accuracy do the solutions from ACs have when compared to the numerical methods running on DCs?

1.2 Proposed Hybrid Computing Architecture

In this dissertation, we propose digitally-programmable analog computing techniques for solving MD linear and non-linear PDEs in real-time. The objective is to design an analog (time-continuous) computational platform in the form of a software-defined integrated-circuit (IC) with supplementary digital logic for solving PDE-based simulations at speeds that are 1-2 orders of magnitude greater than available analog and digital high-performance computation platforms. Fig. 1.3 (a) shows an overview of the proposed accelerator architecture. All the computations are performed in the continuous-time domain by using analog chips that input and output signals via 50 Ω transmission lines. The outputs of the analog chip produce a set of time-varying voltages that are defined by the PDE, initial conditions, and boundary conditions. Fig. 1.3 (b) shows an example of an AC that computes the approximate CT solution of the one-dimensional (1-D) wave equation. Here, the AC produces the space-time variation of the propagating waves for given input excitations and boundary conditions. In other words, the AC generates time-varying voltages that correspond to the electric field variations of the propagating waves at different points in the spatial grid, which are defined by the 1-D wave equation.

The boundary conditions and the input excitations are generated inside a series of field-programmable gate array (FPGA) boards and supplied to the analog chip through digital-to-analog converter (DAC) channels. Computed analog solutions (computational outputs of the analog chip) are routed back into the FPGA through analog-to-digital (ADC) boards. Digitized results can then be used for post-processing inside the FPGA or sent to a Linux PC for analysis. Reconfiguration commands, status lines, and calibration signals are sent by microcontrollers/FPGAs. All FPGAs and microcontrollers are connected through 10 Gbps connections and

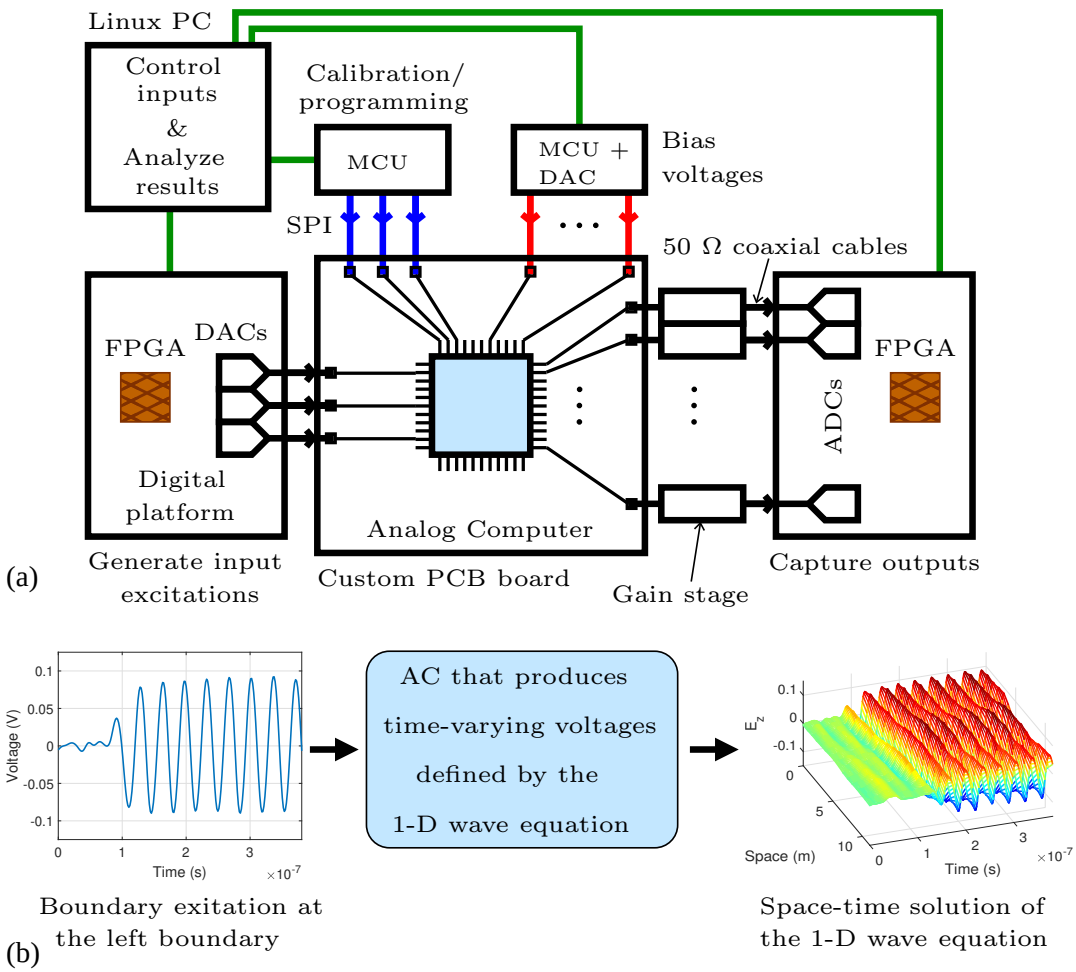

Figure 1.3: (a) Overview of the proposed continuous-time PDE solver with supplementary digital platforms. This dissertation mainly focuses on designing an analog chip that computes the continuous-time solution of a given PDE. (b) An example AC that computes the CT solution of the 1-D wave equation.

are controlled by the Linux PC. This dissertation mainly focuses on designing an analog chip that computes the continuous-time solution of a given PDE. It also demonstrates a prototype analog-digital computational platform using a custom-designed analog computing chip, FPGAs, ADCs, DACs, and microcontrollers.

1.3 Research Objectives

The research objectives of this dissertation are summarized below. Each of the objectives takes an important step forward towards designing a modern version of an AC that accelerates physics-based computations and simulations.

Objective 1: Investigate CT algorithms to map linear PDEs into CT mathematical models that can be realized using analog circuits. The corresponding analog circuit should produce time-varying voltages/currents that correspond to the solutions of the given PDE.

Objective 2: Verify the correctness of the proposed algorithms by applying them to a set of selected PDEs. The corresponding ACs can be designed and simulated using ideal analog circuits. Also, identify the main building blocks towards designing a CMOS-based AC.

Objective 3: Identify how the proposed methods can be scaled to the problem, size, and dimensionality as well as identify any restrictions on the generalizability of the approaches.

Objective 4: Evaluate the performance (accuracy) of the designed ACs by comparing the CT solutions that are produced by the ideal analog circuits with the solutions from the standard numerical solvers. This is essentially the best-case analysis (upper limits of the proposed methods).

Objective 5: Explore algorithms to design ACs that can compute the CT solutions of non-linear PDEs. Also, identify the main building blocks towards designing a CMOS-based AC.

Objective 6: Design a low-frequency prototype of an AC that solves a selected PDE using discrete integrated circuits (ICs). Verify its functionality.

Objective 7: Identify key challenges of realizing large-scale analog networks such

as the proposed PDE solvers for high-frequency applications using CMOS technologies and investigate possible solutions. Simulate the ACs using non-ideal analog circuits (dominant pole models) to verify the proposed approaches.

Objective 8: Extend non-ideal analog solvers towards CMOS-based implementations and quantify their accuracy. Design, simulate, layout and fabricate a selected AC using 180 nm CMOS technology towards a chip-level implementation. Digital programmability of the AC is important for compensating process, voltage, and temperature (PVT) variations following calibration of the chip.

Objective 9: Design and implement a prototype of an analog-digital hybrid computing platform using the fabricated analog chip, FPGA+ADC/DCA boards, and microcontrollers.

Objective 10: Calibrate the AC (digitally programmable gains and bias voltages) to improve the performance of the computation. Collect measurements and quantify the accuracy of the AC for different boundary conditions.

Objective 11: Estimate the relative speedup of the AC versus best-in-class DCs and existing ACs. Compare the power consumption of the AC with existing analog and digital computers.

1.4 Dissertation Roadmap

The roadmap for designing analog accelerators that solve linear and nonlinear PDEs is presented under three main sections as described below.

1) **CT algorithms and ideal ACs:** Explore new algorithms to map a given PDE into analog circuits that can eventually compute the solution of the PDE. The proposed algorithms were verified using ideal ACs that compute the solution of Maxwell's and wave equations.

2) **ACs in 180 nm CMOS technology:** Design and simulate ACs using 180 nm CMOS technology. Extend the AC that solves 1-D wave equations towards chip-level implementation (layout and fabrication).

3) **Prototype hybrid computational platform:** Design and implement a prototype analog-digital hybrid computation platform using the fabricated AC and FPGA+ADC/DAC platforms. The hybrid platform was used to take the measurements of the AC. The performance in terms of speed, area, and power of the AC were compared with existing ACs and DCs.

This dissertation is organized as follows. Chapter 2 revisits the history of analog and hybrid computing techniques. It also includes a discussion of the modern era of computing and a review of recent examples of ACs. Fig. 1.4 shows how the remaining chapters in this dissertation are organized and presented. Chapter 3 introduces the continuous-time in Laplace domain (CTLD) and the all-pass delay approximation (APDA) methods for solving linear PDEs. In Chapter 4, the proposed methods are employed for the design of ACs that compute the solution of Maxwell's equations. In Chapter 5, the CT solution of 1-D and two-dimensional (2-D) wave equations are found using the CTLD and APDA methods. In Chapters 4 and 5, the corresponding ACs are simulated using ideal analog circuits, and the results are compared with the results for closely-related FDTD solutions. The APDA method is extended in Chapter 6 to the design of ACs for solving non-linear PDEs. A low-frequency prototype of a CTLD-based AC is implemented using discrete ICs in Chapter 7, and the corresponding measured results are presented. Chapter 8 discusses the key challenges of CMOS implementations. This chapter also explains the implementation details of the 180 nm CMOS AC that solves 1-D wave equations. The hybrid measurement setup is discussed in detail in Chapter 9 along with the initial measurement results. This chapter also discusses the calibration procedure of the AC that improves the

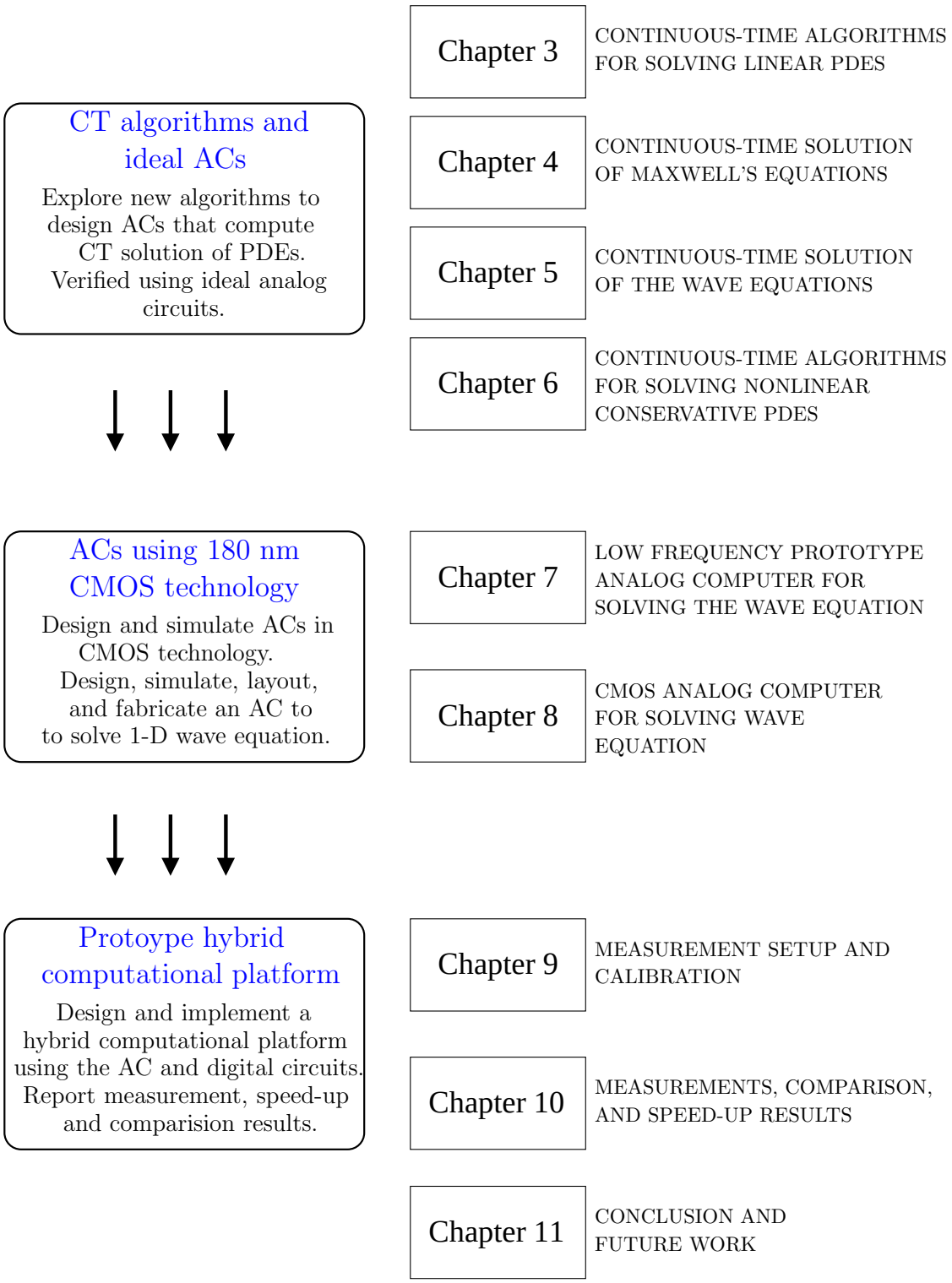

Figure 1.4: Roadmap of this dissertation.

accuracy of the computation. The measurements, comparison with previous work, and speed-up results are provided in Chapter 10. Here, the measurement results are provided for different boundary conditions and frequencies. Chapter 11, which concludes this dissertation, discusses the major outcomes of this research work.

CHAPTER 2

ANALOG COMPUTING TECHNIQUES

There are four classes of electronic systems in the world that can be expressed in terms of their operating domains (signal and time). The four classes are continuous-signal continuous-time (CSCT), discrete-signal discrete-time (DSDT), continuous-signal discrete-time (CSDT), and discrete-signal continuous-time (DSCT) [57]. Fig. 2.1 shows the four classes with example circuits. In general, continuous-signal (CS) and discrete-signal (DS) circuits are referred to as analog and digital circuits, respectively, irrespective of their time-domain representations [57]. Analog and digital systems can then be divided into the four separate classes based on the time-domain representation as shown in Fig. 2.1. Examples of CSCT systems includes op-amps, analog all-pass filters, and current mirrors. Digital processors and FPGA platforms are examples of DSDT systems. Switched capacitor filter and charge-coupled devices represent the electronics in the CSDT class. Asynchronous digital communication circuits are examples of DSCT systems. This dissertation mainly focuses on CSCT systems. In later chapters, DSDT systems along with CSCT electronics are employed in the design of analog-digital hybrid computational platform.

The proposed hybrid system solves PDEs that are defined by physical systems. In general, physical systems are continuous in signal, time, and space domains (in the dimensions x, y and z) and describe the continuous variation of one or more physical quantities (such as temperature, pressure, electric field, magnetic field, and fluid velocity) in space and time [19–23, 35–37]. Such systems have applications in fluid dynamics, thermodynamics, electromagnetics, magnetohydrodynamics, etc [20–23, 36–39]. The solutions to these PDEs are, therefore, continuous in the signal, time and space domains. However, finding an analytical closed-form solution (continuous

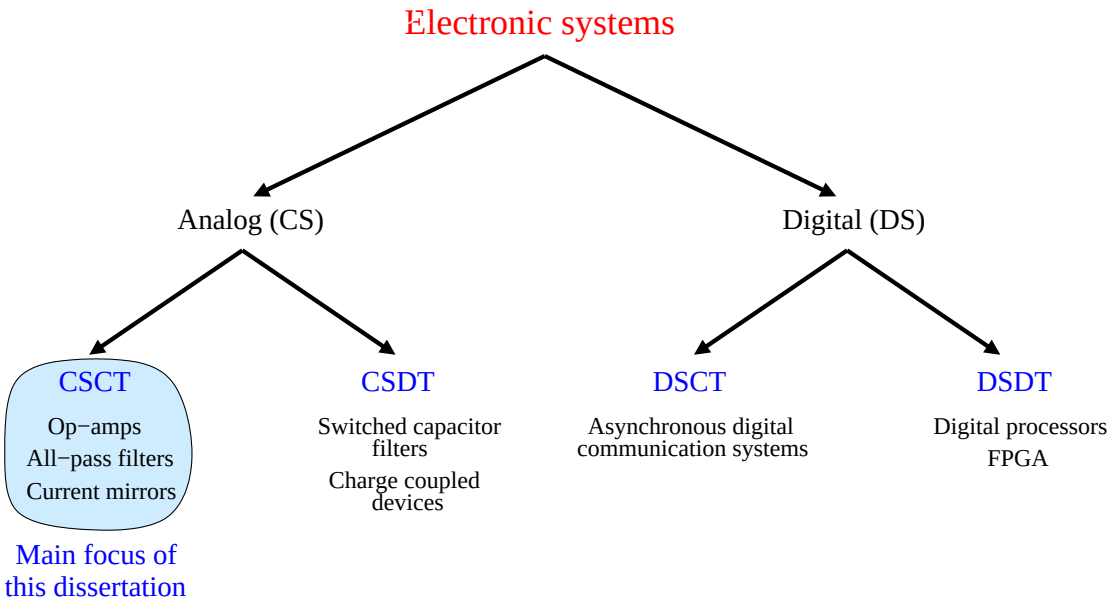

Figure 2.1: The four types of electronic systems with example circuits.

in both space and time) for a complex physical problem is a tedious, if not practically impossible task. Therefore, discrete-domain numerical methods running on powerful computers have been used to solve such problems. All of these methods discretize the PDEs in both dimensions to enable computation using digital computers. Note that a physical system at a given spatial point represents a CSCT system. These systems are more naturally solved using CSCT electronics (analog computing systems) based on spatially discrete but time-continuous update equations. In general, spatially-discrete time-continuous (SDTC) algorithms running on ACs can be potentially faster and more energy-efficient than fully discrete numerical solvers [7, 9, 10, 12–15]. Furthermore, ACs can have bandwidths that greatly exceed the equivalent bandwidths of modern digital computing platforms.

By the 1940s, the high speed, low cost, and small size of electronic computing elements were already making electronic computers more popular than the earlier *mechanical* ACs [4, 58–61] (such as differential analyzers [2] and tide-predicting machines [62]), although at the cost of lower accuracy [58]. Electronic circuits have been

used to simulate impurity profiles of semiconductor junctions [63], nuclear fission reactors [64], and process control loops involving proportional-integral-differential (PID) controllers [65]. Such circuit models can be simulated using electronic ACs, which can either be passive or active [58, 64, 66]. Passive ACs use a network (grid) of resistive and reactive elements to directly implement the circuit model, i.e., capture spatial distributions of the physical quantities being modeled (e.g., voltage, power, temperature, pressure, fluid flow rate, or wave amplitude) [59–61, 66, 67]. By contrast, active analog computers generally use operational amplifiers (op-amps) to implement analog computations, thus enabling accurate and flexible devices that can be easily reconfigured to solve a variety of problems [10, 68–70]. One of the first major successes of such op-amp-based computers was the M9 gun director developed at Bell Labs during World War II [4], and improved versions remained popular until the 1970s.

2.1 Analog Computing History

Analog computing techniques and instruments have a long history of mechanical analog systems ranging from sundials, castel clock to complex astrolabes [71]. Among them, the Antikythera mechanism is considered to be the worlds oldest known mechanical AC, dating back to 200 B.C. [72]. An astrolabe is also a similar device, but less complex, and is thought to have existed around the same time. In recent times, several mechanical instruments and computers had been invented to solve different computational problems. In general, ACs can be divided into two main categories: special-purpose computers and general-purpose computers. Special-purpose computers aim to solve one particular problem, whereas the general-purpose computers can alter the arrangement of interconnections between the computing elements (or programmed) to solve many different problems in a variety of

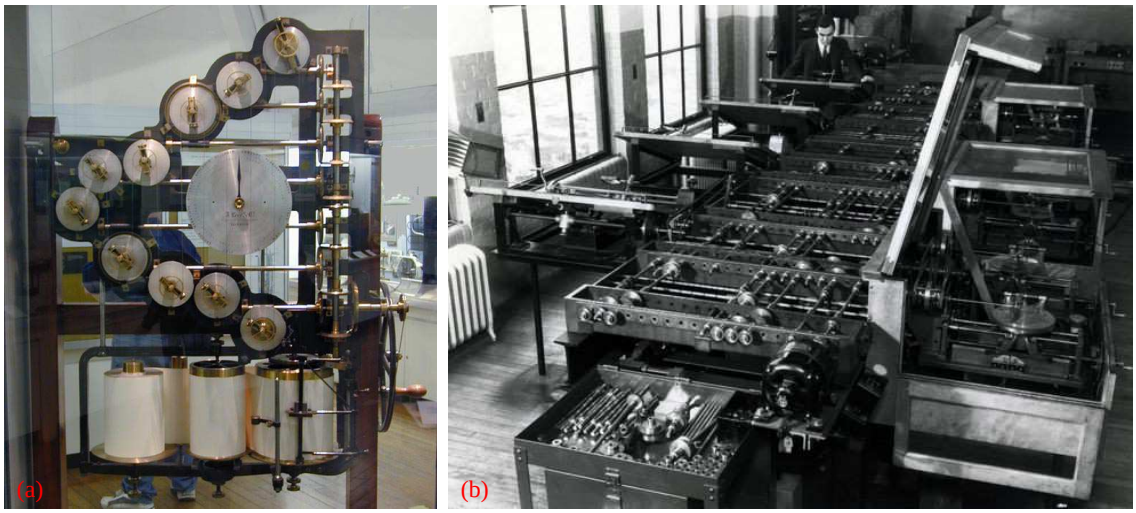

Figure 2.2: (a) A 10-component tide-predicting machine by Sir William Thomson (Lord Kelvin) (Image: https://en.wikipedia.org/wiki/Tide-predicting_machine), (b) Vannevar Bush's differential analyzer (Image: https://www.computerhistory.org/revolution/analog-computers/3/143/311).

applications. Also, ACs can be sub-categorized based on the type of computational elements: mechanical, electromechanical, and electronic. The tide predicting machine that was invented by Lord Kelvin to predict the ebb and flow of sea tides [62] is one of the earliest examples of special-purpose mechanical analog computers (Lord Kelvin is considered as the father of analog computing). Fig. 2.2 (a) shows an image of the tide-predicting AC. Mechanical analog computing reached its zenith with the differential analyzer that was invented by Professor Vannevar Bush at MIT in 1930 using mechanical wheels and disk integrators [2]. This computer was used 24 hours a day during world war II and helped to solve problems from the MIT Radiation Laboratory. Fig. 2.2 (b) shows an image of the MIT differential analyzer.

In 1938, the United States Navy had developed a trigonometry-based electromechanical AC (Torpedo data computer) to solve the problem of firing a torpedo at a moving target [73]. This can be considered as the first electromechanical AC. Several electromechanical differential analyzers were also developed after the MIT

mechanical analyzer, especially after World war II. Nordsieck's "Synchro Operated Differential Analyzer" is a successful example of such analyzers [74]. Early DCs were also classified as electromechanical systems since they were constructed from switches and relay logic rather than vacuum tubes (thermionic valves) or transistors. By the 1940s, electronic analog computing methods were becoming the most preferred methods for computations and simulations rather than mechanical and electromechanical ACs. This was mainly due to the speed, cost and size characteristics of the electronic computing elements compared to its mechanical counterpart [4, 58–61]. However, it is worth noting that the mechanical computers were more accurate than the early electronic-based systems [58].

The invention of the op-amp had largely superseded the mechanical and electromechanical ACs. The op-amp is a device that can perform the mathematical operations of addition, subtraction, integration, and differentiation electronically. Antiaircraft gun detectors that were designed by the United States and the United Kingdom are successful developments of the op-amp-based fully-electronic analog computers [4, 71]. During that time, the main advantage of this computer was the high computational speed compared to the mechanical counterpart. The development of very high gain op-amps in the late 1940s led to a significant improvement in electronic analog computing technologies. A general-purpose analog computer (GPAC) was developed at Bell Labs by following the technology from the M9 antiaircraft gun director and was operational by 1949 [3]. At the same time, a flight simulator was developed at MIT using analog computing techniques [3]. This was one of the largest post-world-war analog computing projects. The first electronic training simulator was build by The Foxboro Company to simulate a proportional-integral-differential (PID) controller [65].

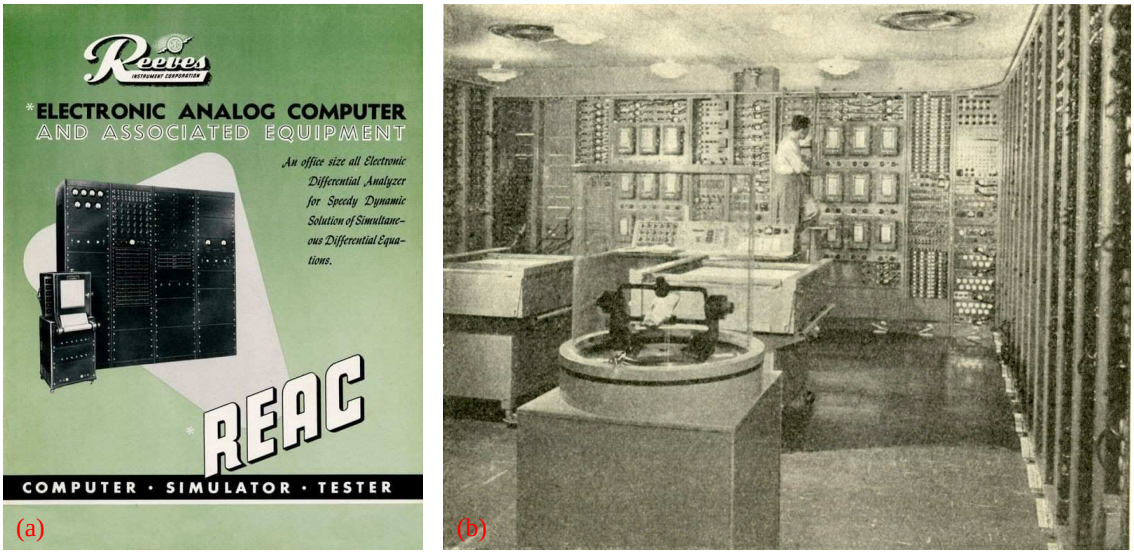

Figure 2.3: (a) Brochure of the Reeves Instrument Corporation's electronic analog computer (Image: https://www.computerhistory.org/revolution/analog-computers/3/150/354), (b) overview of the "Project Typhoon" analog computer (Image: http://www.joostrekveld.net/?p=1409).

The Reeves Instrument Corporation (RIC) designed and constructed an analog computation and simulation laboratory to conduct guided-missile simulations [3,71]. This was one of the most significant analog computing projects that were funded by the Office of Naval Research (ONR) and was called Project Cyclone. In 1948, RIC become the first company to market a complete general-purpose analog computing system, which was able to solve larger, more complex problems. The Reeves electronic analog computer (REAC) is shown in Fig. 2.3 (a). Project Typhoon, also funded by ONR, was a follow-up project to Cyclone and was targeted on obtaining the highest possible performance and precision from analog computations [3,71]. It was also designed to simulate guided missiles. Fig. 2.3 (b) shows an overview of the "Project Typhoon" AC. The Goodyear electronic differential analyzer (GEDA) [71,75] and the Boeing electronic analog computer (BEAC) [71] were two other advanced ACs that were developed by the U.S. Air Force for simulating missile systems.

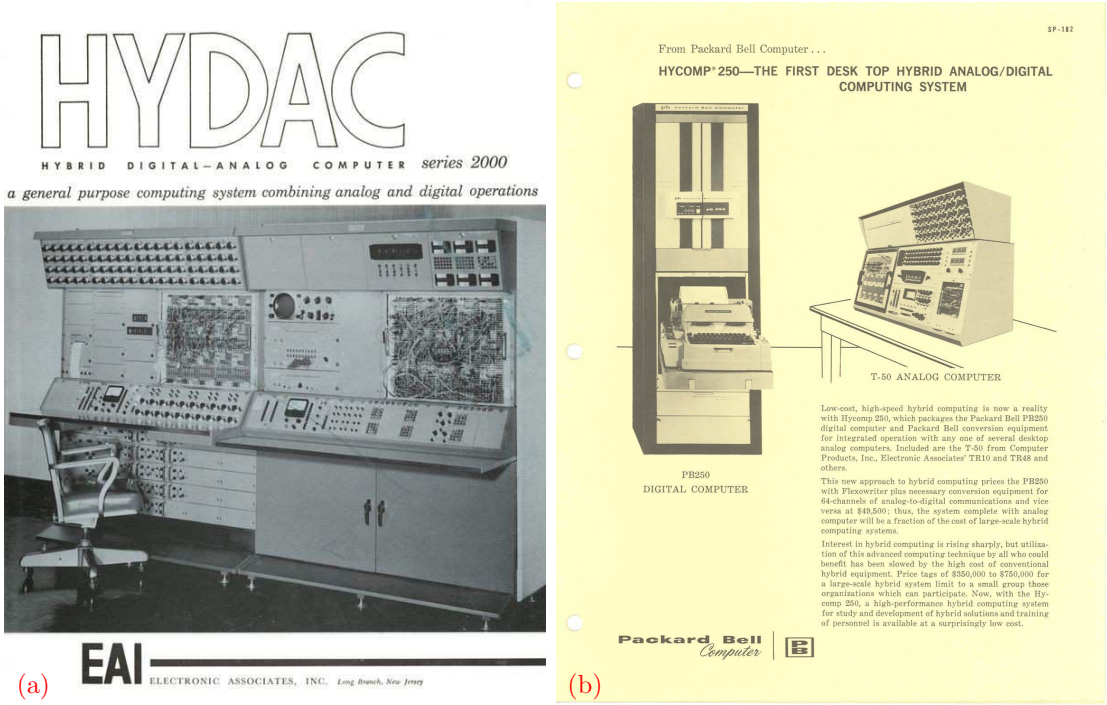

Figure 2.4: Brochure of (a) the HYDAC 2000 and (b) the Hycomp 250 analog-digital hybrid computers (Image: https://www.computerhistory.org/brochures/doc-4372957168f00/).

By the late 1950s, electronic ACs were largely replaced by analog-digital HCs. The intercontinental ballistic missile (ICBM) program was one of the main influences for the inversion of hybrid computational techniques. The first hybrid computer was developed in 1954 at Convair Astronautics [76, 77]. The Ramo-Wooldridge Corporation developed an HC in 1955 by combining commercial analog and digital computers using a special interface called Add-a-Verter [3, 71]. They have used the PACE system by Electronic Associate, Inc. and the IBM 704 system as the analog and digital components of the HC, respectively [76, 77]. A digital control circuit was introduced in the HYDAC 2000 hybrid computer, where the main computation was done in an AC. Hycomp 250, which was developed by Packard Bell in 1961, was considered as the first desktop hybrid computing system. Here, the HC was designed using the PB250 digital computer and the T-50 analog computer [3, 71]. Figs. 2.4

(a) and (b) show the marketing brochures of the HYDAC 2000 and Hycomp 250 HCs, respectively.

2.1.1 Analog and Hybrid Computers that Solve PDEs

A physical system described by a set of equations can be modeled using analog circuits only if the set forms a tensor equation [78]. Within this constraint, analog circuits can be used to model a wide variety of continuous-time physical systems described by PDEs. These include models of fluid dynamics, thermodynamics, electromagnetics, quantum mechanics, and magnetohydrodynamics [4,58–61,63,64,78–82]. In [83], a multidimensional resistance network was proposed to compute the solution of PDEs with different boundary conditions. As an example, the solution of two-dimensional (2-D) Laplace's equation was computed using a 2-D resistance network. The accuracy and simplicity of these networks made them a useful tool for analog computations. A similar method to solve PDEs using resistance networks was proposed in [84], where the solver was used to compute the electric potential variation in a plane diode as a function of the distance from the cathode. An analog circuit model of Maxwell's equations in three dimensions was first proposed by Kron [81]. The voltage differences and branch currents associated with each circuit element in this passive network (which consists of resistors, capacitors, inductors, and ideal transformers) represent the electric and magnetic properties of the physical system. A corresponding passive circuit model was then developed to simulate the transient response of a dipole antenna to an incident electromagnetic field. The matrix form of Maxwell's equations was later used to demonstrate the formal relationship between field variables (electrical and magnetic fields) and circuit variables (current and voltages) [82].

The electronic differential analyzers were extended in [85] towards solving PDEs by approximating one or more partial derivatives using appropriate finite differences. Here, linear PDEs were first converted into one or more ODEs by applying the method of separation of variables. Mathematical derivations and corresponding analog circuits were introduced for solving heat, wave and beam equations. The accuracy of the proposed method was reported for different applications. A general method for solving PDEs using op-amp based integrators and differentiators was proposed in [86]. Examples of analog circuits to solve Laplace, Poisson, and Helmholtz equations were introduced with their theoretical derivations. Equivalent mathematical formulations of the PDEs characterizing field problems were discussed in [87] along with the corresponding resistance-reactance networks. The proposed analog computing techniques were applied for solving elliptic, parabolic, hyperbolic, and biharmonic equations. Several applications were considered and analog circuits were introduced to compute the CT solutions of PDEs.

An analog-digital HC is proposed in [88] to solve PDEs with time-dependent problems involving second-order derivatives. The heat equation and the wave equation were considered for hybrid computations with multidimensional problems and boundary conditions. The development of a hybrid simulator that solves the transient field problems in nuclear reactors and power plants was proposed in [89] with a comprehensive theoretical framework. Monte Carlo methods were developed in [90] to compute approximate solutions of the PDEs using an analog-digital HC. A similar method was proposed in [91] that utilizes sequential estimation techniques for Monte Carlo–based hybrid computations. Applications to the solution of PDEs and stochastic perturbation–based optimization techniques were reported. Hybrid computing techniques for solving parabolic and hyperbolic PDEs were proposed in [92] along with the application to heat transfer equations. In [93], a digital-computer

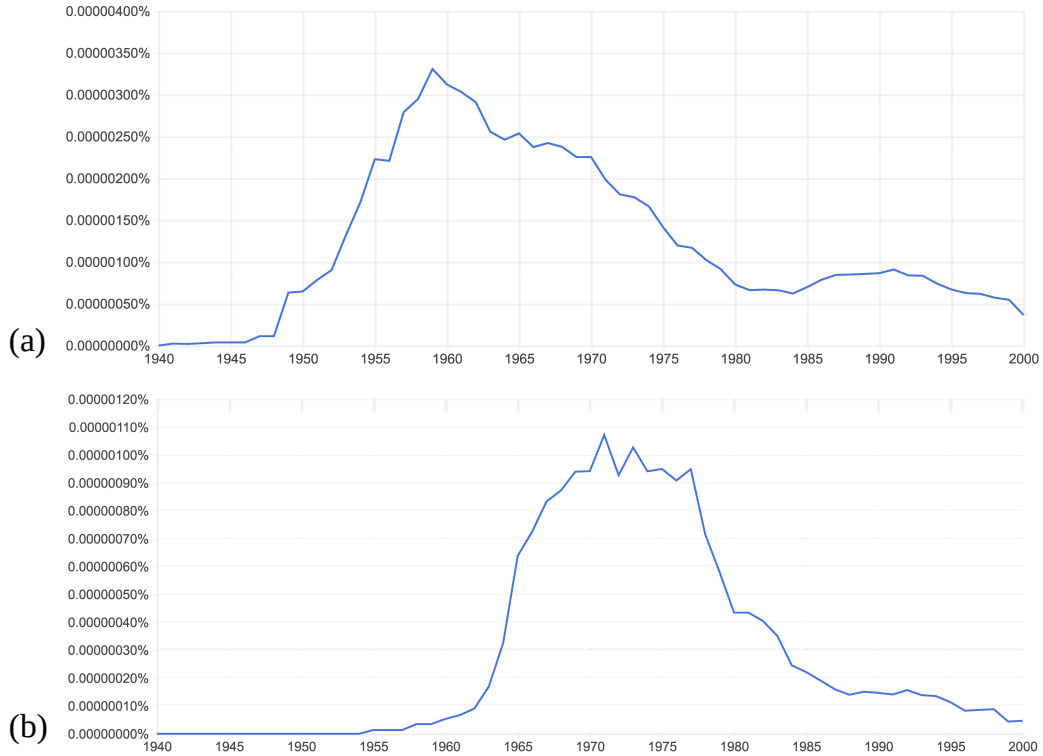

Figure 2.5: Results from Google Ngram Viewer for the two words (a) "analog computing" and (b) "hybrid computing" over the period of 1940-2000.

oriented hybrid system was proposed to compute the solutions of nonlinear PDEs. A hybrid nonlinear PDE simulator was proposed in [94] to analyze the flow of fluids in underground formations. Here, the matrix inversion operations in the computation were realized using an analog resistance network.

Fig. 2.5 shows the results from Google Ngram Viewer for the two wards "analog computing" and "hybrid computing" over the period of 1940 to 2000. Google Ngram Viewer shows how the two words have been used over time using more than 30 million books in print. It is clear that both analog and hybrid computing techniques were popular around 1950 to 1970 and then entered a period of rapid decline with the invention of digital computers [3, 71].

2.2 Modern Era of Analog/Hybrid Computers

Although ACs were surpassed by the exponential growth in DCs (followed by transistor scaling), ACs are returning as DCs are approaching limits in clock-speed, power dissipation, and transistor density [7, 9, 10, 12]. Combining analog circuits with modern digital processors to perform arithmetic operations would make the computation potentially faster and more energy-efficient [7, 9, 10, 13–16]. In general, ACs trade off the accuracy of the computation for less power and higher throughput.

Yannis Tsividis' group at Columbia University has developed two hybrid computing chips (in 2005 and 2015) for energy-efficient ODE- and PDE-based computations. Both ACs solve ODEs and PDEs using an integrator-based signal flow graph. An 80th order AC was designed by Glenn Cowan in 2005 [10, 11] employing 250 nm CMOS technology. His analog computer contains a large number of signal routing switches and functional blocks (integrators, VGA/2-input multipliers, fanout blocks, logarithm blocks, exponential blocks, and programmable blocks which can implement different mathematical operations). The chip only consumes 300 mW of power and is 400 times faster than the MATLAB FDTD-based solvers (at that time). A fourth-order HC was designed by Ning Guo in 2015 [8, 9, 14, 16]. This prototype HC is a successor to the earlier design built by Cowan and has features that permit calibration for more accurate results and easier interfacing with conventional digital architectures. The chip can implement nonlinear functions using DACs, static random access memory (SRAM), and ADCs in a table-lookup manner. The application of nonlinear PDEs on Guos' chip is discussed in [16] with several example problems. In [95], the prototype hybrid solver is used to demonstrate a case study in solving the Black–Scholes stochastic differential equation.

Memristor-based computing techniques have also gained interest in recent years for power-efficient analog computations and simulations. A memristor is a two-terminal device that can switch its resistances based on the history of applied voltage and current. The electronic tunability of the memristor made it significant for analog computations. It was introduced by Chua in 1971 [96]. The physical implementation of the memristors was reported in 2008 at HP Labs [97]. Prospective applications for memristive devices were discussed in [98] along with their key challenges in nano-scale implementations. The vector-matrix multiplication was implemented using a memristor crossbar in [99] with applications in signal and image processing. A memristor crossbar array was used in [100] to accelerate algorithms in deep neural networks where they demonstrated a 128×64 array for high precision analog tuning and control. A memristor-based in-memory computing system was proposed in [101] to solve elliptic and hyperbolic PDEs. The water wave propagation problem and the plasma evolution in a reactor were demonstrated using a high-precision memristor crossbar. A memristor-based linear equation accelerator was proposed in [102]. The proposed approach was 1500 times faster and 8.5 times energy-efficient compared with the existing solvers.

2.3 Towards This Dissertation

Most of the existing and past analog and hybrid computers are based on passive elements or active integrators and differentiators. The use of passive elements can reduce errors in the continuous-time computation since passive elements are less noisy. However, it is difficult to realize high-quality on-chip inductors and transformers for broadband operation, which limits the analog bandwidth $F_{compute}$ for CT operations of such passive solvers. Furthermore, it is impossible to compensate for

signal losses in a fully-passive AC, whereas the active methods have the flexibility to control the gains appropriately. In active ACs, the finite bandwidth of practical integrators/differentiators is one of the main challenges that need to be addressed when targeting CT computations at high frequencies. The finite bandwidth of the circuits introduces a signal propagation delay (PD) from its input to the output (which is inevitable for all practical elements). This PD affects the CT computation and needs to be compensated, especially at high frequencies. However, at low frequencies, the PD is insignificant compared to the frequency of interest. The analog bandwidth of the Columbia hybrid prototype is ∼20-25 kHz [7, 9, 10, 16].

Considering these factors, this dissertation investigates new CT algorithms that are more convenient for high-speed implementations of analog/hybrid computational platforms. These algorithms are then extended towards an active circuit-based analog accelerator for solving PDE-based computations. The proposed ACs only use resistors, capacitors, and active elements, which make them suitable for realization as CMOS integrated circuits. Moreover, the proposed solvers can also compensate for implementation errors such as i) losses in the circuit elements and ii) propagation delays between the nodes, unlike the existing and past designs.

CHAPTER 3

CONTINUOUS-TIME ALGORITHMS FOR SOLVING LINEAR PDES

There have been immense efforts in the computational engineering community to simulate physical systems that are defined by one or more PDEs. In general, these simulations start by first discretizing the equations using a staggered computational grid. Discretized equations are then solved using numerical methods running on digital computers (using the software) [34, 42, 103–105]. However, these physical systems are themselves time-continuous and therefore are more naturally solved using analog computing systems based on SDTC update equations. Furthermore, SDTC algorithms running on analog computers can be potentially faster and more energy-efficient than fully-discrete numerical solvers [7–15]. This dissertation proposes two novel algorithms to map a given PDE into an SDTC update equation, which can then be implemented using analog circuits. Here, the dependent variables of the PDEs such as temperature, pressure, electric field, magnetic field, and fluid velocity are represented using voltages or currents in the analog circuit. Electronic elements are then interconnected in such a way that they produce a defined set of time-varying currents/voltages that are proportional to the solutions of the PDE at each spatial point. More precisely, the variation of the currents/voltages is defined by the PDE.

In the first method, the partial derivatives in the spatial dimension are approximated using discrete finite differences, while the LT is applied to partial derivatives in the time dimension [29–31, 106–109]. The resulting mixed domain (spatially-discrete and time-continuous) update equation is used to design an analog circuit that can compute the solution for a given spatial point. The resulting analog computing modules are then interconnected in a systolic array architecture to compute the solution over the whole spatial grid. The second method replaces the

discrete-time difference operators in the standard FDTD cell (Yee cell), using a continuous-time delay operator, which can be realized as an analog all-pass filter (APF) [110–116]. The summing and scaling operations in the Yee cell are realized using op-amps. Individual cells can then be interconnected in a systolic array to compute the complete solution, as in the first method. Different boundary conditions and problem geometries can be simulated by modifying the boundary cells and inter-cell connectivity, respectively. In this chapter, we introduce the necessary mathematical derivations of the two methods based on a one dimensional (one spatial dimension and time) second-order linear PDE. The following two chapters will apply these two algorithms to solve Maxwell's and wave equations.

3.1 Background

This section briefly discusses the necessary background before introducing the two CT analog computing methods. Note that the proposed algorithms are discrete in space and continuous in time such that it can be computed using an array of analog circuits. Here, the spatial discretization is achieved by applying the finite difference approximation to the spatial domain partial derivatives of the PDEs.

3.1.1 Finite Difference Approximation

Partial derivatives of a PDE can be approximated using differential quotients. There exist several methods for approximating differentials of a given function. Examples of such methods are Taylor series expansion and polynomial expansion of degree n [32, 33, 117]. Here, only the Taylor series expansion is considered. Given an analytical function $W(x,t)$, the first and higher partial derivatives about x can be approximated by analyzing the function values around x. The accuracy of the

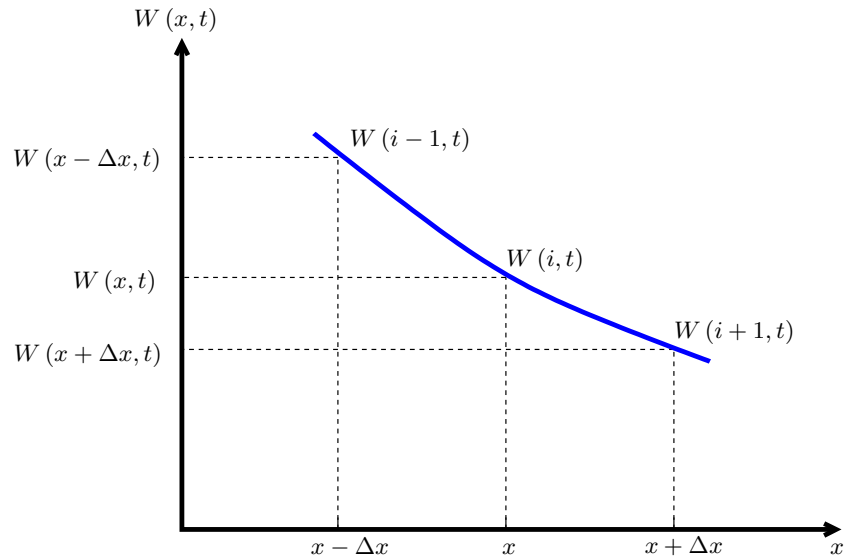

Figure 3.1: Discretization of the function $W(x,t)$ in spatial domain x (only shows the spatial dimension x).

approximation depends on the number of terms selected from the Taylor series expansion.

In order to derive the approximate expressions for partial derivatives, consider the discretization of the spatial dimension as shown in Fig. 3.1 (only shows the spatial dimension), where Δx is the spatial step size. The function is then defined at integer grid points $x = i\Delta x$ (i.e., $i \in \mathrm{Z}$), where i is the spatial index. The Taylor series expansion of $W(x+\Delta x,t)$ about x is

$$W(x+\Delta x,t) = W(x,t) + (\Delta x)\frac{\partial W}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 W}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 W}{\partial x^3} + \cdots . \tag{3.1}$$

This leads to

$$\begin{aligned}\frac{\partial W}{\partial x} &= \frac{W(x+\Delta x,t) - W(x,t)}{\Delta x} - \frac{(\Delta x)}{2!}\frac{\partial^2 W}{\partial x^2} + \frac{(\Delta x)^2}{3!}\frac{\partial^3 W}{\partial x^3} + \cdots , \\ &= \frac{W(x+\Delta x,t) - W(x,t)}{\Delta x} + \mathcal{O}(\Delta x) , \\ &= \frac{W(i+1,t) - W(i,t)}{\Delta x} + \mathcal{O}(\Delta x) ,\end{aligned} \tag{3.2}$$

which is an approximation for the partial derivative $\frac{\partial W}{\partial x}$ about x. Here, $x = i\Delta x$ and $W(i,t) \equiv W(i\Delta x,t)$. This is called the forward finite difference approximation.

The order of the approximation is Δx (a first order error term). By applying the Taylor series expansion on $W(x-\Delta x, t)$

$$W(x-\Delta x, t) = W(x,t) - (\Delta x)\frac{\partial W}{\partial x} + \frac{(\Delta x)^2}{2!}\frac{\partial^2 W}{\partial x^2} - \frac{(\Delta x)^3}{3!}\frac{\partial^3 W}{\partial x^3} + \cdots, \quad (3.3)$$

the backward difference approximation of $\frac{\partial W}{\partial x}$ can be obtained as

$$\begin{aligned}\frac{\partial W}{\partial x} &= \frac{W(x,t) - W(x-\Delta x, t)}{\Delta x} + \mathcal{O}(\Delta x), \\ &= \frac{W(i,t) - W(i-1,t)}{\Delta x} + \mathcal{O}(\Delta x).\end{aligned} \quad (3.4)$$

The order of the approximation is Δx. By subtracting (3.1) from (3.3) the centered difference approximation of $\frac{\partial W}{\partial x}$ can be obtained as

$$\frac{\partial W}{\partial x} = \frac{W(x+\Delta x, t) - W(x-\Delta x, t)}{2\Delta x} + \mathcal{O}(\Delta x)^2. \quad (3.5)$$

The order of the approximation is $(\Delta x)^2$ (a second order error term).

In order to derive the approximate expression for the higher order derivatives, consider the Taylor series expansion of $W(x+2\Delta x, t)$ as

$$W(x+2\Delta x, t) = W(x,t) + (2\Delta x)\frac{\partial W}{\partial x} + \frac{(2\Delta x)^2}{2!}\frac{\partial^2 W}{\partial x^2} + \frac{(2\Delta x)^3}{3!}\frac{\partial^3 W}{\partial x^3} + \cdots. \quad (3.6)$$

Multiplying (3.1) by 2 and subtracting it from (3.6) leads to

$$\begin{aligned}-W(x+\Delta x, t) + W(x+2\Delta x, t) = -W(x,t) &+ (\Delta x)^2\frac{\partial W}{\partial x} + (\Delta x)^2\frac{\partial^2 W}{\partial x^2} \\ &+ (\Delta x)^3\frac{\partial^3 W}{\partial x^3} + \cdots.\end{aligned} \quad (3.7)$$

Solving the expression (3.7) for $\frac{\partial^2 W}{\partial x^2}$ leads to

$$\frac{\partial^2 W}{\partial x^2} = \frac{W(x+2\Delta x, t) - 2W(x+\Delta x, t) + W(x,t)}{(\Delta x)^2} + \mathcal{O}(\Delta x). \quad (3.8)$$

This represents the forward difference approximation of the second derivative of $W(x,t)$. The order of the approximation is Δx. Similarly, backward difference approximation of the derivative can be obtained as

$$\frac{\partial^2 W}{\partial x^2} = \frac{W(x,t) - 2W(x-\Delta x, t) + W(x-\Delta x, t)}{(\Delta x)^2} + \mathcal{O}(\Delta x). \quad (3.9)$$

By adding (3.1) and (3.3), the central difference approximation of the second derivative can be obtained as

$$\frac{\partial^2 W}{\partial x^2} = \frac{W(x+\Delta x,t) - 2W(x,t) + W(x-\Delta x,t)}{(\Delta x)^2} + \mathcal{O}(\Delta x)^2. \tag{3.10}$$

These expressions are used in this chapter to derive new SDTC algorithms. Approximations for higher-order derivatives can be obtained by following the same procedure. Also, higher-order error terms can be obtained for the approximation by selecting the required number of terms from the expansion.

3.1.2 Laplace Transform

The Laplace transform (LT) is a powerful tool that has numerous applications in electrical engineering. It transforms a given function $W(t)$ with a real variable t (can be time or a space variable) into a complex function $\bar{W}(s)$ with a complex variable s. The definition of the Laplace transform can be expressed as

$$\mathcal{L}[W(t)] = \bar{W}(s) = \int_{0^-}^{\infty} W(t)\, e^{-st} dt. \tag{3.11}$$

Here, $s = \sigma + j\omega$ is the Laplace domain variable. The corresponding inverse transform is defined as

$$\mathcal{L}\left[\bar{W}(s)\right] = W(t) = \frac{1}{2\pi j}\int_{\omega_1 - j\omega}^{\omega_1 + j\omega} \bar{W}(s)\, e^{st} ds. \tag{3.12}$$

Similarly, the Laplace transform can be applied to a two dimensional function $W(x,t)$ as

$$\mathcal{L}[W(x,t)] = \bar{W}(x,s) = \int_{0^-}^{\infty} W(x,t)\, e^{-st} dt. \tag{3.13}$$

Note that this is not a multidimensional LT (m integrations are required for a $m-$dimensional transform). Here, the Laplace transform is only applied with respect to the time variable t (this property is important when deriving the proposed

algorithms). Also, the LT is a linear operator such that it can be applied to any linear operation. Which means

$$\mathcal{L}\left[aW_1\left(x,t\right)+bW_2\left(x,t\right)\right]=a\mathcal{L}\left[W_1\left(x,t\right)\right]+b\mathcal{L}\left[W_2\left(x,t\right)\right], \tag{3.14}$$

where a and b are constants. The LTs of the first and second order partial derivatives can be obtained as

$$\begin{aligned}\mathcal{L}\left[\frac{\partial W\left(x,t\right)}{\partial t}\right]&=sW\left(x,s\right)-W\left(x,0\right),\\ \mathcal{L}\left[\frac{\partial^2 W\left(x,t\right)}{\partial t^2}\right]&=s^2W\left(x,s\right)-sW\left(x,0\right)-W'\left(x,0\right),\end{aligned} \tag{3.15}$$

respectively. Here, $W\left(x,0\right)$ and $W'\left(x,0\right)=\left.\frac{\partial W(x,t)}{\partial t}\right|_{t=0}$ are the initial conditions of the function. These two properties are used when deriving the first CT algorithm. The LTs of the higher order derivatives can be obtained in a similar way. The time shifting property of the Laplace transform is

$$\mathcal{L}\left[W\left(x,t-\tau\right)u\left(t-\tau\right)\right]=e^{-\tau s}\bar{W}\left(x,s\right). \tag{3.16}$$

Here, $u\left(t\right)$ is the Heaviside step function. This property is used when deriving the second CT algorithm.

In general, LT is used to transform the time-domain circuits into s domain circuits since it simplifies the solution of integral differential equations into a manipulation of a set of algebraic equations. In the same way, the LT is used in our methods to map the SDTC algorithms into an analog circuit.

3.2 Continuous-time Algorithms for Solving PDEs

A general form of a second-order linear PDE given in (3.17) is utilized to formulate the two CT algorithms that solve linear PDEs. Here, only two dimensions are considered: space x and time t. However, the proposed methods can be extended to

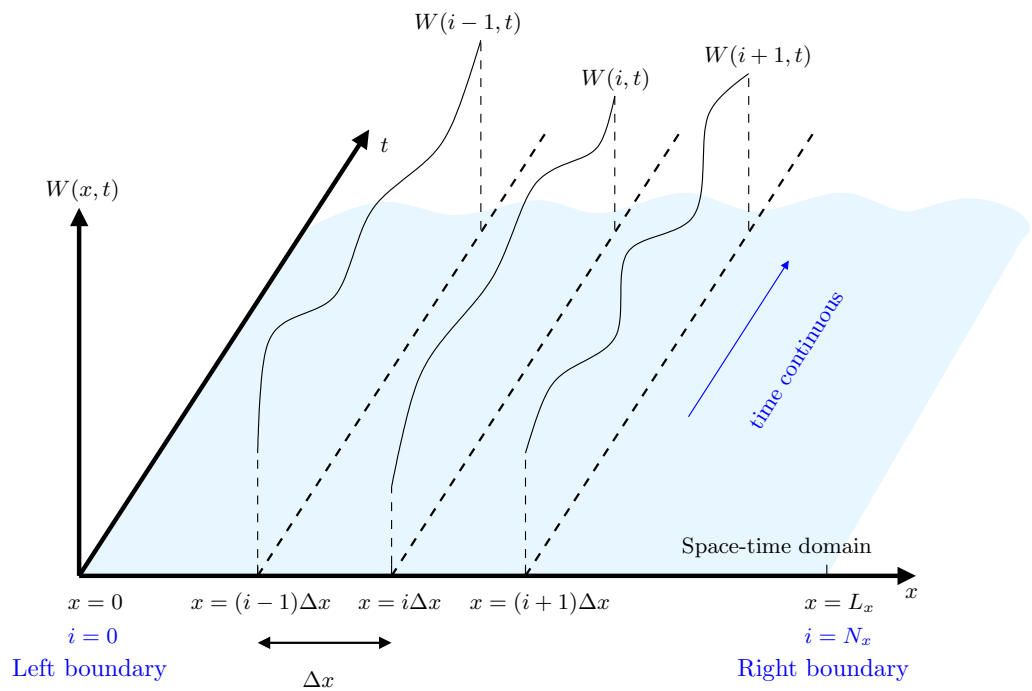

Figure 3.2: The discretization of the spatial dimension x from 0 to L_x with the spatial step size Δx. Here i is the discrete spatial index.

multiple dimensions by following the same procedures. Chapter 5 provides an example of a 2-D PDE solver (space x, y and time t). Consider a PDE given in the form

$$A\frac{\partial^2 W(x,t)}{\partial t^2} + B\frac{\partial^2 W(x,t)}{\partial x^2} + C\frac{\partial W(x,t)}{\partial t} + D\frac{\partial W(x,t)}{\partial x} + EW(x,t) + F = 0, \quad (3.17)$$

where x and t are the spatial and time variables, respectively. A, B, C, D, E, and F are constant coefficients. Consider the space-time domain as shown in Fig. 3.2, where $0 \leq x \leq L_x$ and $0 \leq t < \infty$. Note that both algorithms are discrete in space. Thus, the initial step of the algorithms is the discretization of the spatial dimension. In particular, a uniform grid is selected with a grid spacing of $\Delta x = \frac{L_x}{N_x}$, where i denotes the spatial index ($i \in \{0, 1, 2, \ldots, N_x\}$). The discretized space consists of $N_x + 1$ grid points. Here, $i = 0$ and $i = N_x$ correspond to the spatial locations at the left and right boundaries. The following sections introduce the two analog computing methods based on this discretized spatial grid.

3.2.1 Continuous-time in Laplace Domain (CTLD) Method

In the continuous-time in Laplace domain (CTLD) method, the partial derivatives in the spatial dimension are approximated using discrete finite differences, while the LT is applied along the time dimension [29–31, 106–109]. This results in a spatially-discrete and time-continuous update equation. The CT solution is then computed by realizing the corresponding update equation using analog circuit elements. The design of an analog 2-D beam filter where the time-domain was realized in RC-active low-frequency op-amp circuits was proposed by Bruton [29–31]. Here, we extend the approach in [29–31] to the general case of solving PDEs. The step by step procedure of the CTLD method is described below.

Step 1: Approximate the spatial domain partial derivatives in (3.17) using finite difference approximations. The resulting SDTC expression is

$$\begin{aligned} &A\frac{\partial^2 W(i,t)}{\partial t^2} + B\left[\frac{W(i+1,t) - 2W(i,t) + W(i-1,t)}{\Delta x^2}\right] + C\frac{\partial W(i,t)}{\partial t} \\ &+ D\left[\frac{W(i,t) - W(i-1,t)}{2\Delta x}\right] + EW(i,t) + F = 0. \end{aligned} \tag{3.18}$$

The first- and second-order derivatives are approximated using the backward (with the first-order accuracy) and central (with second-order accuracy) differences, respectively. The most suitable finite difference method (forward, backward or central) can be selected based on the resulting mathematical model. In general, the method that leads to a realizable analog circuit implementation with the lowest hardware complexity will be selected. The accuracy of the computation can be improved by selecting a higher-order accurate finite difference scheme to approximate the derivatives. However, this will increase the hardware complexity of the AC.

Step 2: Apply the LT on (3.18) along the time dimension. The resulting mixed-domain expression is

$$A\left[s^2\bar{W}(i,s) - sW(i,0) - W'(i,0)\right] + \frac{B}{\Delta x^2}\left[\bar{W}(i+1,s) - 2\bar{W}(i,s) + \bar{W}(i-1,s)\right]$$
$$+ C\left[s\bar{W}(i,s) - W(i,0)\right] + \frac{D}{2\Delta x}\left[\bar{W}(i,s) - \bar{W}(i-1,s)\right] + E\bar{W}(i,s) + F = 0, \tag{3.19}$$

where s is the Laplace domain variable. Here, one independent variable of the solution is in the space domain (discrete), where as the other variable is in the Laplace domain (not in the time domain). Thus, (3.19) is a mixed-domain expression. $\bar{W}(i,s)$ is the Laplace domain representation of $W(i,t)$ [29–31]. $W(i,0)$ and $W'(i,0)$ define the initial conditions of the system. Also, note that this expression is discrete in space and continuous in time (the Laplace domain variable is continuous).

Step 3: Solve (3.19) for the mixed-domain solution $\bar{W}(i,s)$. This is called the SDTC update equation of the PDE, which is used to compute the CT solution of the PDE at spatial index i. For zero initial conditions, the solution $\bar{W}(i,s)$ can be expressed as

$$\bar{W}(i,s) = \frac{\alpha\bar{W}(i-1,s) + \beta\bar{W}(i+1,s) - F}{As^2 + Cs + \gamma}, \tag{3.20}$$

where, $\alpha = -\frac{B}{\Delta x^2} + \frac{D}{2\Delta x}$, $\beta = \frac{-B}{\Delta x^2}$, and $\gamma = -\frac{2B}{\Delta x^2} + \frac{D}{2\Delta x} + E$. The update equation can be used to implement an analog module that computes the solution at spatial locations $x = i\Delta x$.

Analog Circuits: Consider the internal module (IM) that computes the solution $W(i,t)$ at spatial index i ($W(i,s)$ is the LT of $W(i,t)$). Based on (3.20), the module requires $W(i-1,t)$ and $W(i+1,t)$ functions as the inputs, which correspond to the solutions at $i-1$ and $i+1$ spatial locations, respectively. The numerator polynomial $\alpha\bar{W}(i-1,s) + \beta\bar{W}(i+1,s) - F$ can be realized using an adder/subtractor circuit (scaling and summing), where as the $\frac{1}{As^2+Cs+\gamma}$ operator can be implemented using a passive LRC circuit or an active circuit realization method

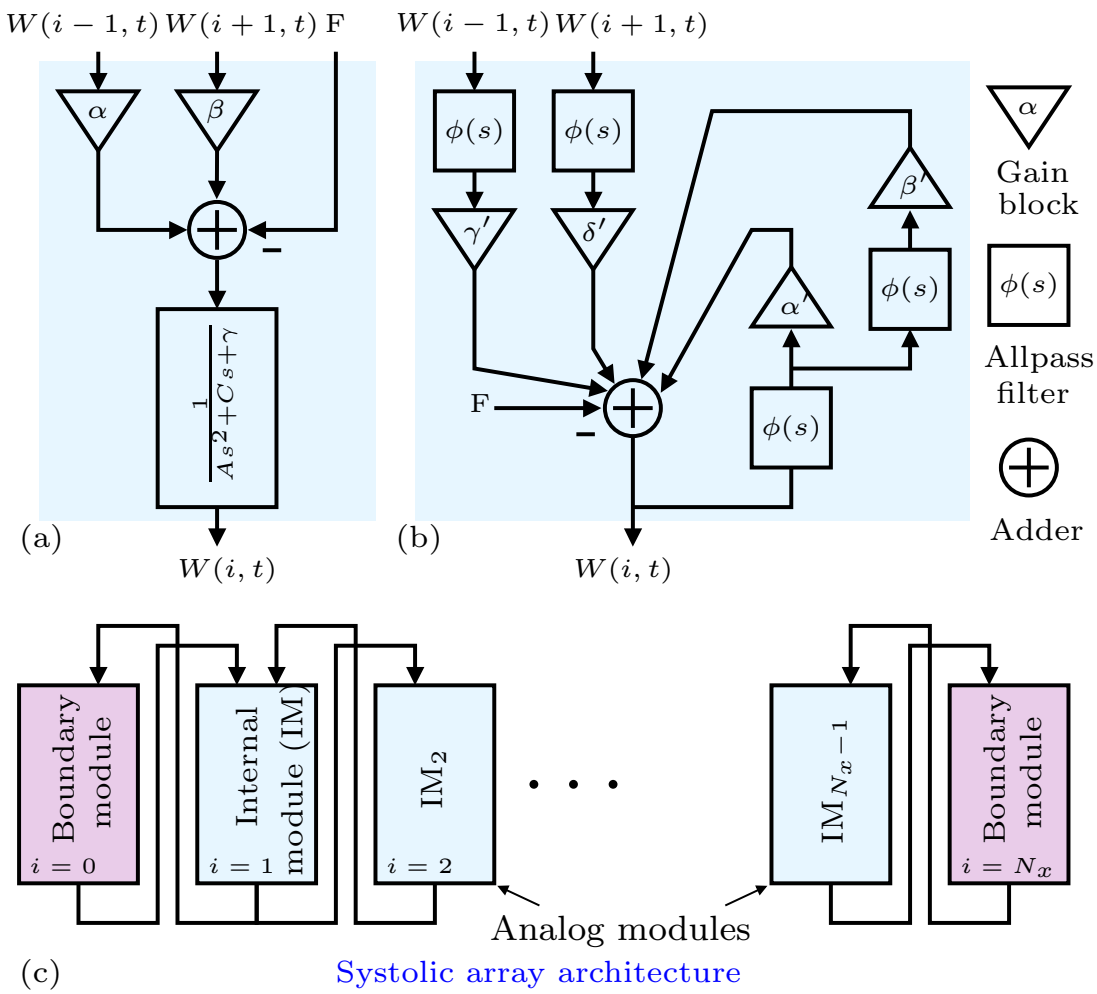

Figure 3.3: Block diagram of the internal module (IM) derived from (a) the direct LT method and (b) the all-pass filter method. (c) Systolic array architecture of the second order continuous-time PDE solver.

(e.g. Sallen-Key topology [118]). Fig. 4.1 (a) shows a block diagram of the corresponding IM. The continuous-time solution to the PDE over the whole spatial grid can be computed by interconnecting each module in an analog array architecture as shown in Fig. 4.1 (c). The proposed method can be extended to multiple dimensions and higher order PDEs.

At each boundary ($i = 0$ and $i = N_x$), different modules need to be implemented based on the governing equations at the boundary. As an example, consider the

left boundary $i = 0$ with a Dirichlet boundary. The Dirichlet boundary condition specifies the solution $W(0, t)$ at the boundary [20, 38, 119]. Thus, we do not need any additional hardware to implement the boundary module. A signal source that produces the corresponding solution $W(0, t)$ can be used to excite the IM at $i = 1$ (i.e. the $W(i + 1, t)$ input of the IM_1 is $W(0, t)$). However, for other boundary conditions such as the Neumann, and the Robin boundary conditions, a specific module needs to be designed based on the governing equations at the boundary. The CTLD method can be employed to derive the corresponding CT update equation at the boundary, which can then be used to design the boundary module. Several examples on modeling different boundary conditions using the CTLD method will be discussed in the succeeding chapters with their corresponding analog circuits.

3.2.2 All-pass Delay Approximation (APDA) Method

The second CT method of solving PDEs is proposed as a direct mapping from a fully-discrete (discrete in both space and time) FDTD method to a CT implementation. The corresponding CT update equations are realized by replacing unit sample delays in the digital prototype with an analog all-pass filter, thereby converting a digital prototype to a continuous-time analog version [110–115]. Here, the all-pass filter is approximating the unit sample delay. Thus, the method is named as all-pass filter approximation (APDA) method. Summing and scaling operations are implemented using adder/subtractor circuits (op-amps). Following is the step by step procedure of the APDA method for obtaining an SDTC update equation for a given PDE.

Step 1: Approximate the spatial-domain partial derivatives in (3.17) using the finite difference approximations. This step is similar to **Step 1** of the CTLD method. The

resulting SDTC expression is

$$A\frac{\partial^2 W(i,t)}{\partial t^2} + B\left[\frac{W(i+1,t) - 2W(i,t) + W(i-1,t)}{\Delta x^2}\right] + C\frac{\partial W(i,t)}{\partial t} + D\left[\frac{W(i,t) - W(i-1,t)}{2\Delta x}\right] + EW(i,t) + F = 0. \tag{3.21}$$

The first and second order derivatives are approximated using the backward (with first order accuracy) and central (with second order accuracy) differences, respectively. Here, the spatial variable x has been discretized.

Step 2: Approximate the time-domain partial derivatives in 3.21 using finite differences, but keeping the time variable continuous. The resulting expression is

$$A\left[\frac{W(i,t+\tau) - 2W(i,t) + W(i,t-\tau)}{\tau^2}\right] + B\left[\frac{W(i+1,t) - 2W(i,t) + W(i-1,t)}{\Delta x^2}\right] + C\left[\frac{W(i,t+\tau) - W(i,t-\tau)}{2\tau}\right] + D\left[\frac{W(i,t) - W(i-1,t)}{2\Delta x}\right] + EW(i,t) + F = 0. \tag{3.22}$$

Here, τ is the CT delay operator of the update equation. Both first and second order partial derivatives are approximated using centered finite differences with second order accuracy. As similar to the CTLD method, the most suitable finite difference method (forward, backward or central) can be selected based on the resulting mathematical model. Note that the expression given in (3.22) is still continuous in time even after the application of the finite differences (with the time variable t). This is the main difference between **Step 1** and **Step 2**.

Step 3: Solve (3.22) for $W(i,t+\tau)$ and evaluate the function at time t $(t+\tau \rightarrow t)$. This leads to the SDTC update equation

$$KW(i,t) = \alpha' W(i,t-\tau) + \beta' W(i,t-2\tau) + \gamma' W(i-1,t-\tau) + \delta' W(i+1,t-\tau) - F, \tag{3.23}$$

where $\alpha' = \frac{2A}{\tau^2} + \frac{2B}{\Delta x^2} - \frac{D}{2\Delta x} - E$, $\beta' = \frac{-A}{\tau^2} + \frac{C}{2\tau}$, $\gamma' = \frac{-B}{\Delta x^2} + \frac{D}{2\Delta x}$, $\delta' = \frac{-B}{\Delta x^2}$, and $K = \frac{A}{\tau^2} + \frac{C}{2\tau}$.

Step 4: Apply the LT on (3.23) with respect to the time variable t, which leads to the mixed-domain update equation

$$\begin{aligned} K\bar{W}(i,s) &= \alpha'\bar{W}(i,s)\,e^{-s\tau} + \beta'\bar{W}(i,s)\,e^{-2s\tau} \\ &+ \gamma'\bar{W}(i-1,s)\,e^{-s\tau} + \delta'\bar{W}(i+1,s)\,e^{-s\tau} - F. \end{aligned} \tag{3.24}$$

The resulting update equation can be used to design analog circuits that compute the continuous-time solution of the PDE.

Analog Circuits: The Laplace domain representation $e^{-s\tau}$ is approximated as $e^{-s\tau} \approx \left[\frac{1-\frac{s\tau}{2m}}{1+\frac{s\tau}{2m}}\right]^m$, and can be realized in an analog RC-active topology using a cascade of m all-pass filters. Typically, $m = 3$ is sufficient for approximation of $e^{-s\tau}$ [110–115]. Let $\phi(s) = \left(\frac{1-s\tau/2m}{1+s\tau/2m}\right)^m$ be the transfer function of the all-pass filter that approximates the continuous-time delay τ. Fig. 4.1 (c) shows the block diagram of the proposed IM, where the all-pass filter $\phi(s)$ is used as a building block. The summing and scaling operations can be implemented using adder/subtractor circuits and gain blocks. Note that this procedure is equivalent to a direct mapping from the digital FDTD implementation, where we replace the unit time delays with all-pass filters. The analog modules which correspond to each of the boundaries can be obtained by modeling the governing equations using the above mentioned method. The CT solution over the whole spatial grid can be computed by interconnecting internal and boundary modules in a systolic array architecture as shown in Fig. 4.1 (c).

3.2.3 FDTD Method vs APDA Method

FDTD algorithms are discrete in both space and time. Since the APDA method is derived from a purely discrete-time algorithm, there may be a confusion with the continuous-time nature of the APDA method. The following discus-

sion explains this argument and compares the behavior of the APDA method with the FDTD schemes.

Consider a general FDTD stepping algorithm as shown in Fig. 3.4 (a). The algorithm computes the solution $y(t)$ at time t by utilizing the solutions $y(t-\tau)$ and $y(t-2\tau)$ at time $t-\tau$ and $t-2\tau$, respectively, where τ is the temporal step size (see the FDTD stencil shown in Fig. 3.4 (a)). The algorithm can only access the solutions at discrete times $\{\ldots, t+2\tau, t+\tau, t, t-\tau, t-2\tau, \ldots\}$. Furthermore, the time marching is performed in a discrete manner. Thus, the next step of the discrete-time algorithm is to compute the solution $y(t+\tau)$ at time $t+\tau$ by utilizing the solutions $y(t)$ and $y(t-\tau)$ at time t and $t-\tau$, respectively.

Now consider the proposed APDA method. It also computes the solution $y(t)$ at time t by utilizing the solutions $y(t-\tau)$ and $y(t-2\tau)$ at time $t-\tau$ and $t-2\tau$, respectively (same FDTD stencil). However, the time marching is performed in a continuous manner. Thus, the solution $y(t+\delta t)$ at time $t+\delta t$ can be computed using the solutions $y(t+\delta t-\tau)$ and $y(t+\delta t-2\tau)$ at time $t+\delta t-\tau$ and $t+\delta t-2\tau$, respectively, where δt is a infinitesimally small time. This means the algorithm can access every point at the time dimension (i.e., can be considered as a continuous movement of the FDTD stencil along the time dimension as shown in Fig. 3.4(b)). In contrast, the stencil moves in a discrete manner in the discrete-time algorithm. Thus, the proposed APDA method is a CT algorithm.

Special Note: In a typical sampled discrete-time linear time invariant (LTI) system, such as an FIR or IIR digital filter, both the input spectrum as well as the filter response are periodic in 2π due to the phase-wrapping of the infinite-order all pass filter $e^{-j\omega T}$. When one replaces the digital filter with an analog version as in the APDA method, the input is not sampled any longer. However, the system response is still periodic in 2π because of phase wrapping of $e^{-j2\pi\tau}$ where τ is now a

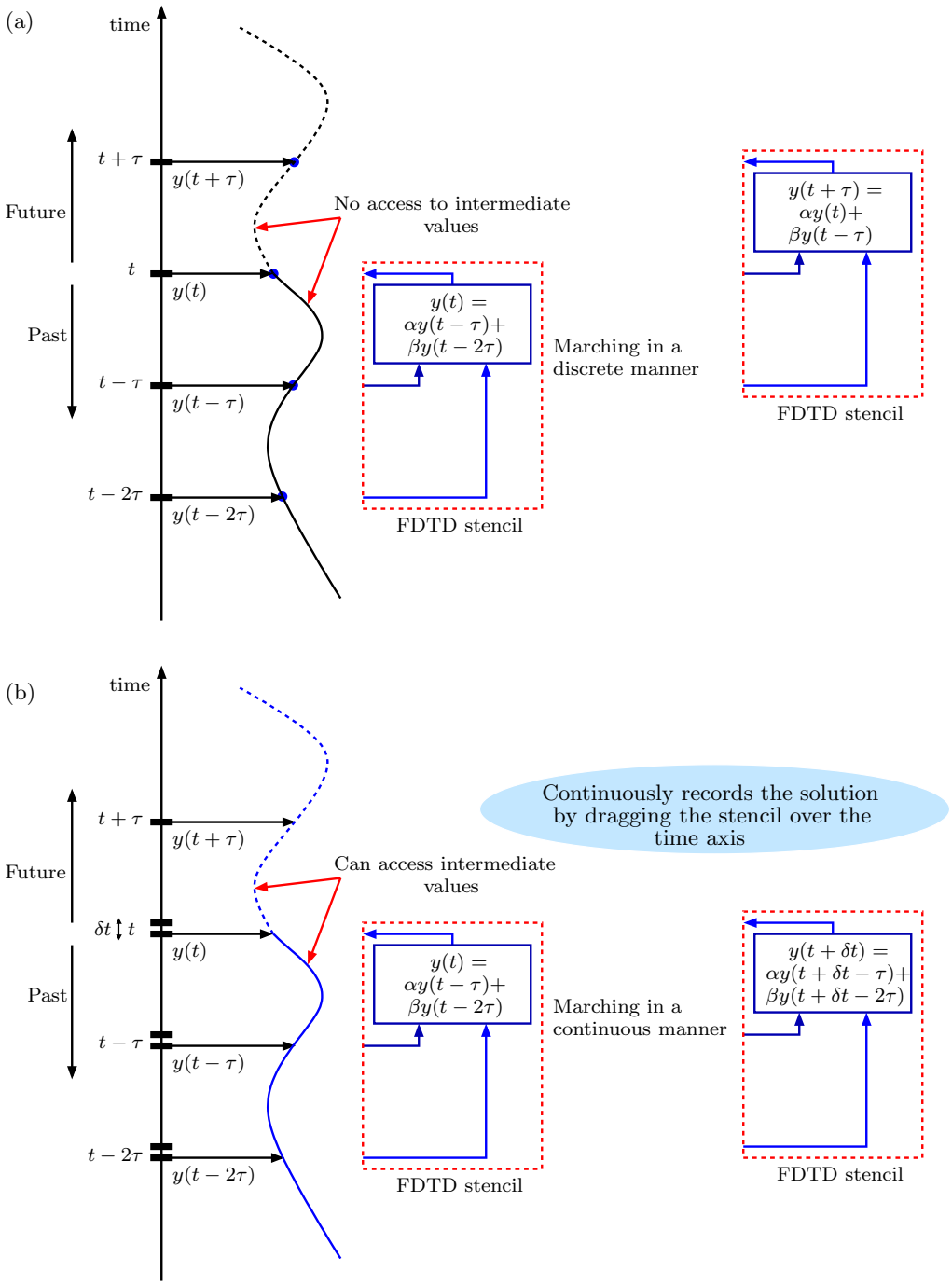

Figure 3.4: (a) The FDTD stepping algorithm. (b) The proposed APDA method.

continuous-time delay. In order for the system to correctly filter the input, it has to be strictly bandlimited, such that only the principal copy of the filter response is rel-

evant for the filtering. The strictly-bandlimited condition ensures that the wrapped copies of the system response do not have any corresponding spectral content in the input signal. By selecting a value of τ in the APDA method, the signals that are present within the circuit model are assumed to be band-limited to $\frac{1}{2\tau}$.

In the next two chapters, the CTLD and the APDA methods are employed to design ACs that compute the CT solution of Maxwell's and wave equations.

CHAPTER 4

CONTINUOUS-TIME SOLUTION OF MAXWELL'S EQUATIONS

This chapter employs CTLD and APDA methods to design ACs that compute the CT solutions of 1-D Maxwell's equations. Both ACs have been simulated using *ideal* analog circuits in Cadence Spectre for different boundary conditions such as the perfect electric conductor (PEC), the perfect magnetic conductor (PMC), and the absorbing boundary. The performance of the ACs have been quantified using i) mean squared difference between the AC results and FDTD simulation results, and ii) the noise to signal energy ratio. The chapter also presents mathematical models (based on the CTLD method) to design ACs that can solve 2-D Maxwell's equations.

Maxwell's equations are a set of coupled linear PDEs that describe the behavior of electric and magnetic fields at each point in space-time [20,23,38]. In the proposed ACs, the CT electric and magnetic fields (the dependent variables in Maxwell's equations) are represented using differences in electrical potentials (a voltage-mode analog circuit) and are defined at discrete spatial points. Electronic elements are then interconnected in such a way that they produce a defined set of time-varying voltages that are proportional to the electric and magnetic fields at each spatial point. More precisely, the variation of the voltages in the circuit is defined by Maxwell's equations. Since the independent variable of the coupled PDEs is time, the proportionality between the problem time scale and the computation time scale is unity.

Maxwell's equations in an isotropic source-free region of space can be expressed as [119, 120]

$$\nabla \times \mathbf{E} = -\mu \frac{\partial \mathbf{H}}{\partial t}, \qquad \nabla \times \mathbf{H} = \epsilon \frac{\partial \mathbf{E}}{\partial t}, \tag{4.1}$$

where $\mathbf{E} \equiv (E_x, E_y, E_z)$ and $\mathbf{H} \equiv (H_x, H_y, H_z)$ are the electric and magnetic field vectors, respectively. The field components E_k and H_k are functions of space and

time and are defined at each space-time point ($k \in \{x, y, z\}$). Here, ϵ and μ are the permittivity and permeability of the medium, respectively. For simplicity, ϵ and μ are assumed to be uniform over the simulation volume. Both methods are applied to the 1-D (one spatial dimension and the time dimension) Maxwell's equations. In 1-D, the transverse magnetic (TM) mode electromagnetic fields are described by

$$\frac{\partial E_z}{\partial t} = \frac{1}{\epsilon}\frac{\partial H_y}{\partial x}, \qquad \frac{\partial H_y}{\partial t} = \frac{1}{\mu}\frac{\partial E_z}{\partial x}, \tag{4.2}$$

and $E_x = 0$, $E_y = 0$, $H_z = 0$, and $H_x = 0$ [119, 120]. Note that both spatial and time variables (x and t) are continuous in nature.

4.1 CTLD Method for Solving Maxwell's Equations

Consider the coupled PDEs given in (4.2). In the CTLD method, the partial derivatives in the spatial dimension are approximated using discrete finite differences, while the LT is applied along the time dimension [29–31, 106–109]. This results in a spatially-discrete and time-continuous update equation. The CT solution is then computed by realizing the corresponding update equation using analog circuit elements.

Consider the spatial domain of $0 \leq x \leq L_x$. The coupled equations given in (4.2) are first discretized over the spatial dimension with a spatial grid size $\Delta x = \frac{L_x}{N_x}$. The electric field E_z is defined at integer grid points (i.e., $i \in \{0, 1, 2, \ldots, N_x\}$) whereas the magnetic field H_y is defined at half-integer points (i.e., $i + \frac{1}{2} \in \{\frac{1}{2}, \frac{3}{2}, \frac{5}{2}, \ldots, N_x - \frac{1}{2}\}$), where $i \in \mathbb{Z}$. The spatial points $i = 0$ and $i = N_x$ correspond to the left and right boundaries, respectively. Here, both boundaries are defined by electric fields. If needed, magnetic field-based boundaries may also be simulated by defining the left and right boundary at $i = \frac{1}{2}$ and $i = N_x - \frac{1}{2}$, respectively. The approximation of the spatial partial derivatives using finite differences leads to (while keeping the

time variable continuous)

$$\begin{aligned}\frac{\partial E_z(i,t)}{\partial t} &= \frac{1}{\epsilon}\frac{H_y(i+\frac{1}{2},t)-H_y(i-\frac{1}{2},t)}{\Delta x},\\ \frac{\partial H_y(i+\frac{1}{2},t)}{\partial t} &= \frac{1}{\mu}\frac{E_z(i+1,t)-E_z(i,t)}{\Delta x}.\end{aligned} \tag{4.3}$$

Here, each of the derivatives are approximated using second-order centered differences. However, more complex higher-order finite difference schemes (and time domain integration methods) can also be employed to improve the accuracy of the solutions. Use of a different approximation will result a different SDTC update equation, which eventually leads to an alternative analog circuit that solves the same PDE. The application of LT on (4.3) leads to the following mixed-domain expressions

$$\begin{aligned}s\bar{E}_z(i,s)-E_z(i,0) &= \frac{\bar{H}_y(i+\frac{1}{2},s)-\bar{H}_y(i-\frac{1}{2},s)}{\epsilon\Delta x},\\ s\bar{H}_y(i+\frac{1}{2},s)-H_y(i+\frac{1}{2},0) &= \frac{\bar{E}_z(i+1,s)-\bar{E}_z(i,s)}{\mu\Delta x}.\end{aligned} \tag{4.4}$$

Here, s is the Laplace domain variable, $H_y(i+\frac{1}{2},0)$ and $E_z(i,0)$ define the initial conditions of the coupled PDE, and $\bar{E}_z(i,s)$ and $\bar{H}_y(i+\frac{1}{2},s)$ are the Laplace transform domain representations of the field components $E_y(i,t)$ and $H_y(i+\frac{1}{2},t)$, respectively. For zero initial conditions, the SDTC update equations that solve E_z and H_y at mesh points i and $\left(i+\frac{1}{2}\right)$ can be expressed as

$$\begin{aligned}\bar{E}_z(i,s) &= \frac{\bar{H}_y\left(i+\frac{1}{2},s\right)-\bar{H}_y\left(i-\frac{1}{2},s\right)}{\epsilon\Delta x s},\\ \bar{H}_y\left(i+\frac{1}{2},s\right) &= \frac{\bar{E}_z(i+1,s)-\bar{E}_z(i,s)}{\mu\Delta x s},\end{aligned} \tag{4.5}$$

The resulting expressions can then be used to solve Maxwell's equations using analog circuits. Note that this is a staggered grid architecture similar to the standard Yee algorithm [41].

4.1.1 Circuit Theory

The SDTC update equations given in (4.5) have poles at $s = 0$. In order to move the poles away from origin, we approximate the corresponding system as

$$\begin{aligned} \bar{E}_z(i,s) &\approx \frac{\bar{H}_y\left(i+\frac{1}{2},s\right) - \bar{H}_y\left(i-\frac{1}{2},s\right)}{\alpha + \epsilon\Delta x s}, \\ \bar{H}_y\left(i+\frac{1}{2},s\right) &\approx \frac{\bar{E}_z(i+1,s) - \bar{E}_z(i,s)}{\beta + \mu\Delta x s}. \end{aligned} \tag{4.6}$$

The resulting system can accurately solve Maxwell's equations when the operating frequency of the solver $\omega \gg \omega_0 = \max\{\frac{\alpha}{\epsilon\Delta x}, \frac{\beta}{\mu\Delta x}\}$, where α and β define the minimum operating frequency ω_0 of the system. In other words, the update equations given in (4.6) accurately approximates (4.5) when $\omega \gg \omega_0$. In order to make the solver works for all frequencies, consider the following modification to (4.6).

$$\begin{aligned} \bar{E}_z(i,s) &= \frac{\bar{H}_y\left(i+\frac{1}{2},s\right) - \bar{H}_y\left(i-\frac{1}{2},s\right) + \bar{E}_z(i,s) - \bar{E}_z(i,s)}{\alpha + \epsilon\Delta x s}, \\ \bar{H}_y\left(i+\frac{1}{2},s\right) &= \frac{\bar{E}_z(i+1,s) - \bar{E}_z(i,s) + \bar{H}_y\left(i+\frac{1}{2},s\right) - \bar{H}_y\left(i+\frac{1}{2},s\right)}{\beta + \mu\Delta x s}, \end{aligned} \tag{4.7}$$

Here, a $\bar{E}_z\left(i,s\right)$ term is added and subtracted to the right-hand side of the first equation. Similarly, $\bar{H}_y\left(i+\frac{1}{2},s\right)$ is added and subtracted to the second equation. In both cases the subtracted term is then moved to the left-hand side, resulting in the following modified update equations:

$$\begin{aligned} \bar{E}_z(i,s) &= \frac{\bar{H}_y\left(i+\frac{1}{2},s\right) - \bar{H}_y\left(i-\frac{1}{2},s\right) + \bar{E}_z(i,s)}{1 + \alpha + \epsilon\Delta x s}, \\ \bar{H}_y\left(i+\frac{1}{2},s\right) &= \frac{\bar{E}_z(i+1,s) - \bar{E}_z(i,s) + \bar{H}_y\left(i+\frac{1}{2},s\right)}{1 + \beta + \mu\Delta x s}, \end{aligned} \tag{4.8}$$

which have poles at $s = -\frac{1+\alpha}{\epsilon\Delta x}$ and $s = -\frac{1+\beta}{\mu\Delta x}$ for the electric and magnetic update equations, respectively. Here, the poles are not at the origin even if $\alpha = 0$ and $\beta = 0$. Thus, we selected $\alpha = 0$ and $\beta = 0$ (throughout the chapter). However, both update equations now need feedback from the output to compute the solution. The modified update equations given in (4.8) (with $\alpha = 0$ and $\beta = 0$) can be used to

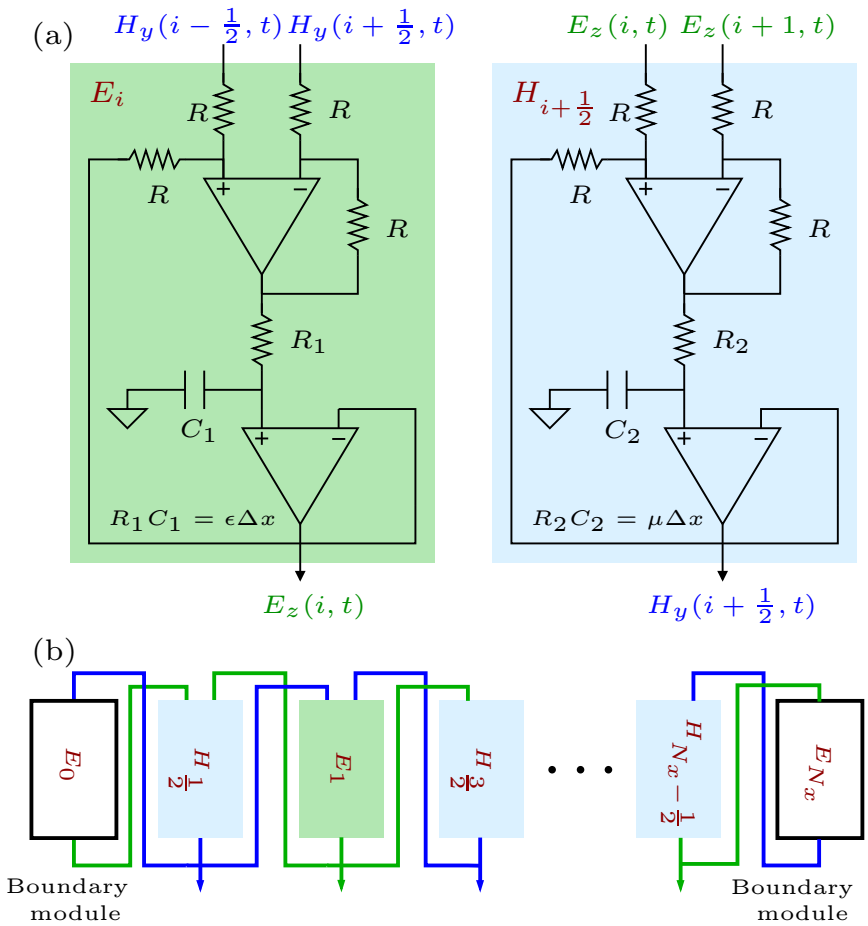

Figure 4.1: (a) Internal analog modules (RC-active circuits) that compute the electric (left) and magnetic (right) fields at internal spatial locations, which are derived based on the CTLD method. (b) Systolic array architecture of the interconnected internal modules to solve Maxwell's equations.

implement analog modules that compute the electric and magnetic field intensities at the internal spatial points $i \in \{\frac{1}{2}, 1, \frac{3}{2}, 2, \ldots, N_x - \frac{1}{2}\}$ (update equations given in (4.6) can also be used to design the internal module. However, this system only works for frequencies $\omega \gg \omega_0$). Two separate modules are required to compute the electric and magnetic fields. Fig. 4.1(a) shows the analog circuit architectures of the electric and magnetic fields internal modules (IMs) E_i and $H_{i+\frac{1}{2}}$. Consider the electric field IM E_i at grid point i. The magnetic field values at the neighboring spatial points ($i-\frac{1}{2}$ and $i+\frac{1}{2}$) are used as the inputs to the module. The operations

in the numerator of the electric field update equation given in (4.8) are realized using an op-amp. The $\frac{1}{1+\epsilon\Delta xs}$ operation (note that $\alpha = 0$) is implemented as a passive low pass filter by setting $R_1C_1 = \epsilon\Delta x$ (see Fig. 4.1 (a)). The output of the low pass filter is buffered prior to feeding the neighboring magnetic field modules. The buffered output is fed back to the summing op-amp. The IM that computes the magnetic field is implemented following the same architecture. Here, the resistor and the capacitor values are selected such that $R_2C_2 = \mu\Delta x$. The CT solution to Maxwell's equations over the internal simulation grid points can then be computed by interconnecting the IMs in a systolic array architecture, as shown in Fig. 4.1(b).

At each boundary, different modules need to be implemented based on the governing equations at the boundary. Consider a situation where we have a perfect electric conductor (PEC) at $x = L_x = N_x\Delta x$ (the right boundary). The component of the electric field parallel to a PEC boundary is zero. Thus, $E_z = 0$ at the boundary. The component of the magnetic field perpendicular to a PEC boundary is also zero. Thus, the PEC boundary can be realized by simply grounding the $E_z\left(i,t\right)$ input of the magnetic IM at spatial point $N_x - \frac{1}{2}$. The perfect magnetic conductor (PMC) boundary condition is the magnetic equivalent of the PEC boundary. The components of the parallel magnetic field and the perpendicular electric field at a PMC boundary are zero. Thus, the corresponding boundary module can be simulated by grounding the $H_y\left(i+\frac{1}{2}\right)$ input of the electric IM at spatial point $N_x - 1$. Here, the right boundary is defined at $x = \left(N_x - \frac{1}{2}\right)\Delta x$. Furthermore, if we need to excite the left boundary using a time varying function $f(t)$, this can be achieved by connecting a signal source that produces $f(t)$ to the $E_z(i-1,t)$ input of the $H_y\left(\frac{1}{2},t\right)$ magnetic field IM.

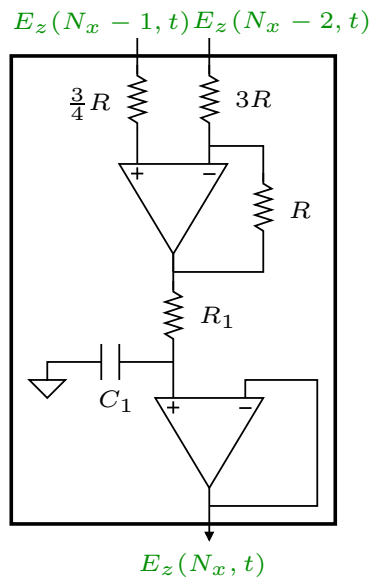

Figure 4.2: Analog circuit architecture of the radiation boundary module that is derived based on the CTLD method.

In order to simulate a radiation boundary at $i = N_x$ (the right side boundary), consider the 1-D Mur absorbing boundary condition [119, 120]

$$\frac{1}{c}\frac{\partial E_z(x,t)}{\partial t}|_{x=N_x\Delta x} + \frac{\partial E_z(x,t)}{\partial x}|_{x=N_x\Delta x} = 0. \tag{4.9}$$

Here $c = \frac{1}{\sqrt{\epsilon\mu}}$ is the wave propagation speed. Application of the finite difference approximation along the spatial dimension followed by the LT along the time dimension on (4.9) leads to a SDTC update equation

$$\bar{E}_z(N_x,s) = \frac{4\bar{E}_z(N_x-1,s) - \bar{E}_z(N_x-2,s)}{3\left(1+\frac{2}{3}\frac{\Delta x}{c}s\right)}. \tag{4.10}$$

Note that a second-order backward approximation is employed to discretize the spatial domain partial derivative. A backward approximation is suitable in this case since there is no reflection from the boundary, so the fields at the boundary can be computed only using fields inside it. Also, zero initial conditions are considered for the LT. The resulting update equation given in (4.10) is used to implement the radiation boundary module. Fig. 4.2 shows the corresponding circuit architecture.

Here, $R_1C_1 = \frac{2}{3}\frac{\Delta x}{c}$. Note that the inputs to the radiation boundary module are the electric fields from spatial points $N_x - 1$ and $N_x - 2$.

4.1.2 Ideal Cadence Simulations

The proposed AC that computes the CT solution of Maxwell's equations was evaluated using a circuit simulator (Cadence Spectre) assuming ideal elements. A spatial grid of 65 spatial points was considered for the simulation, which includes 33 and 32 electric and magnetic field-based spatial points, respectively (left and right boundaries are defined by the electric fields). The analog bandwidth $\mathrm{F}_{compute}$ of the system (for CT operations) was selected as 100 MHz ($\lambda_{min} = \frac{c}{\mathrm{F}_{compute}}$), and the spatial step size is $\Delta x = \frac{\lambda_{min}}{10}$ (oversampled by 5 compared to the Nyquist rate). The left boundary was excited using a Gaussian modulated cosine (GMC) electric field $E_m \cos\left(2\pi f\left(t - t_0\right)\right) e^{-k(t-t_0)^2}$, where f = 50MHz, E_m = 377 mV, and $t_0 = 0.2$ μs. Here, k sets the bandwidth of the signal. The characteristic impedance $\sqrt{\frac{\mu}{\epsilon}}$ and the velocity c of the medium are selected as 377 Ω and $c = 3 \times 10^8$ ms^{-1}, respectively. The right boundary was simulated as a PEC boundary. Figs. 4.3 (a) and (b) show the spatio-temporal variation of the electric and magnetic fields obtained from the proposed AC (the three-dimensional view and the top view). Zero initial conditions are considered for the simulations, and the maximum time step is set to 1 ps.

Similarly, Figs. 4.4 (a) and (b) show the electric and magnetic field variations in the space-time domain but with the right boundary replaced with an absorbing boundary.

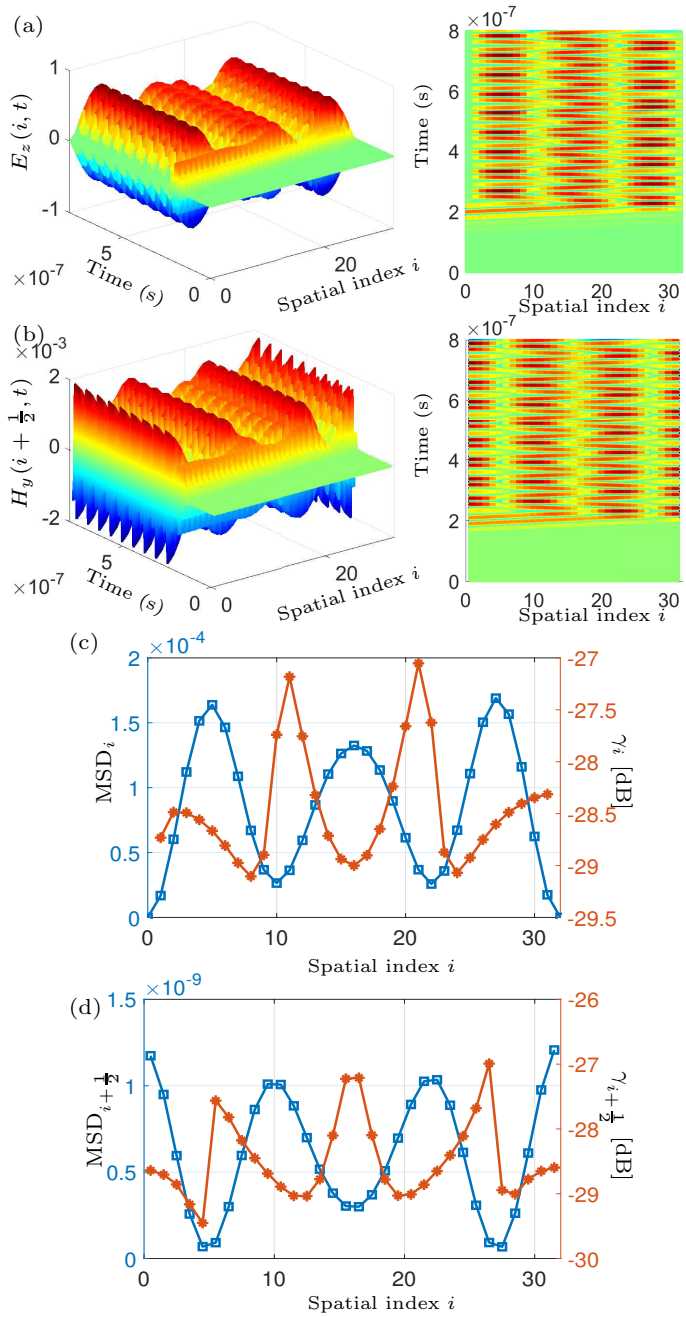

Figure 4.3: Ideal simulation results obtained from the CTLD solver (left: 3-D view, right: top view). The left boundary is excited using a GMC electric field at 50 MHz. The right boundary is simulated as a PEC. The spatio-temporal variation of (a) the electric field E_z and (b) the magnetic field H_y. The deviation of the continuous-time solution compared to the FDTD numerical solution (MSD and γ): (c) the electric and (d) the magnetic field.

4.1.3 Comparative Analysis

In order to quantify how much our continuous-time solution deviates from the standard FDTD solution, two metrics have been calculated: the mean squared difference (MSD) between the two solutions, and the noise energy to signal energy ratio γ (in dB). These metrics describe how much the computed CT solution deviates with respect to the FDTD solution, when the proposed ACs are realized using ideal analog circuits. For this purpose, an FDTD-based Maxwell's equation solver was implemented using MATLAB. In order to define the comparative metrics, consider the electric field solution E_z. Since the values of both metrics vary with the spatial location, MSD_i and γ_i are calculated for each spatial location i. MSD_i for the electric field solution is defined as

$$\mathrm{MSD}_i = \frac{1}{N_t} \sum_{n=0}^{N_t-1} \left[E_F\left(i, n\Delta T\right) - E_A\left(i, n\Delta T\right)\right]^2, \tag{4.11}$$

where $E_F\left(i, n\Delta T\right)$ and $E_A\left(i, n\Delta T\right)$ are the solutions of the FDTD and analog solvers, respectively, at spatial point i and time $t = n\Delta T$. Also, N_t is the total number of time samples and ΔT is the temporal step size of the FDTD simulation. In addition, the noise to signal energy ratio γ_i is expressed as

$$\gamma_i = 10 \log_{10} \frac{\sum_{n=0}^{N_t-1} \left[E_F\left(i, n\Delta T\right) - E_A\left(i, n\Delta T\right)\right]^2}{\sum_{n=0}^{N_t-1} E_F\left(i, n\Delta T\right)^2}. \tag{4.12}$$

Similar metrics corresponding to the magnetic field solutions ($\mathrm{MSD}_{i+\frac{1}{2}}$ and $\gamma_{i+\frac{1}{2}}$) at the half grid points can also be defined. Figs. 4.3(c) and (d) show the electric and magnetic field comparison results (MSD and γ), respectively, for the simulation with the PEC boundary at $x = N_x\Delta x$ (shown in Figs. 4.3(a) and (b)). The temporal step size ΔT is selected as 10 ps for the FDTD simulation. Both analog and FDTD simulations use the same spatial step sizes (10 spatial points per minimum

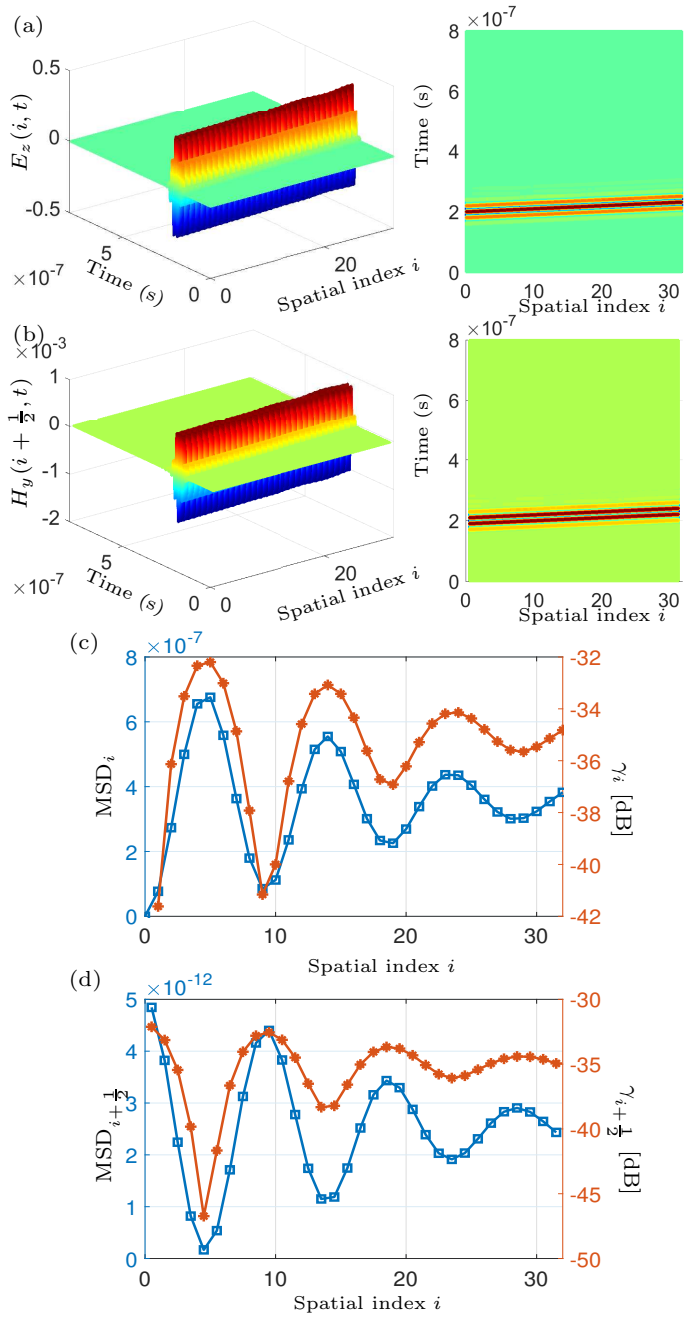

Figure 4.4: Ideal simulation results obtained from the CTLD Maxwell's equation solver (left: 3-D view, right: top view). The left boundary is excited using a GMC electric field at 50 MHz. The right boundary is simulated as a radiation boundary. The spatio-temporal variation of (a) the electric field and (b) the magnetic field. The deviation of the continuous-time solution compared to the FDTD solution (MSD and γ): (c) the electric and (d) the magnetic field.

wavelength), while $N_t = 30000$ for the comparison. Figs. 4.4(c) and (d) show the corresponding comparison results for the simulation with a radiation boundary on the right hand side (shown in Figs.4.4(a) and (b)). In both cases, the electric field MSD results are much larger than the magnetic fields. This is due to the amplitude difference of the fields: the simulated electric and magnetic field amplitudes are 377 mV (E_m) and ≈1 mV, respectively, which matches the expected characteristic impedance of the propagation medium ($\sqrt{\frac{\mu}{\epsilon}} = 377\ \Omega$). However, the values of γ are in the same range since they do not depend on the amplitudes of the fields. The proposed CTLD method is able to compute the solutions of Maxwell's equations with a difference less than -26 dB in terms of γ (assuming ideal circuit models).

4.1.4 Continuous-time Solution of 2-D Maxwell's Equations

The 2-D TM mode electromagnetic fields satisfy [119, 120]

$$\begin{aligned} \frac{\partial H_x}{\partial t} &= -\frac{1}{\mu}\frac{\partial E_z}{\partial y}, \\ \frac{\partial H_y}{\partial t} &= \frac{1}{\mu}\frac{\partial E_z}{\partial x}, \\ \frac{\partial E_z}{\partial t} &= \frac{1}{\epsilon}\frac{\partial H_y}{\partial x} - \frac{1}{\epsilon}\frac{\partial H_x}{\partial y}, \end{aligned} \tag{4.13}$$

where $E_x = 0$, $E_y = 0$, and $H_z = 0$. Spatial index j represents the grid points in y dimension (Δy is the corresponding step size). Consider a 2-D computational domain of $0 \leq x \leq L_x$ and $0 \leq y \leq L_y$ ($\Delta y = \frac{L_y}{N_y}$). The electric field E_z is defined at integer grid points for both dimensions (i.e., (i,j)). The H_x component is defined at integer and half integer grid points for x and y directions, respectively (i.e., $\left(i, j + \frac{1}{2}\right)$). Similarly, the H_y component is defined at half and integer grid points for x and y dimensions, respectively (i.e., $\left(i + \frac{1}{2}, j\right)$). The spatial points with $i = 0$, $j = 0$, $i = N_x$, and $j = N_y$ correspond to left, bottom, right, and top boundaries of the spatial grid, respectively. Application of the CTLD method on

(4.13) leads to

$$\bar{H}_x(i+\frac{1}{2},j+\frac{1}{2},s) = \frac{\bar{E}_z(i,j,s)-\bar{E}_z(i,j+1,s)}{\mu\Delta ys},$$
$$\bar{H}_y(i+\frac{1}{2},j+\frac{1}{2},s) = \frac{\bar{E}_z(i+1,j,s)-\bar{E}_z(i,j,s)}{\mu\Delta xs},$$
$$\bar{E}_z(i,j,s) = \frac{\bar{H}_y(i+\frac{1}{2},j,s)-\bar{H}_y(i-\frac{1}{2},j,s)}{\epsilon\Delta xs} - \frac{\bar{H}_x(i,j+\frac{1}{2},s)-\bar{H}_x(i,j-\frac{1}{2},s)}{\epsilon\Delta ys}, \tag{4.14}$$

which can then be used to solve the 2-D Maxwell's equations using analog computing methods. Here, zero initial conditions are considered for the LT. The internal analog modules that compute E_z, H_x, and H_y field components can be realized based on the expressions in (4.14). Consider the electric field IM. The magnetic field intensity values at the neighboring spatial points $(i-\frac{1}{2},j)$, $(i+\frac{1}{2},j)$, $(i,j-\frac{1}{2})$, $(i,j+\frac{1}{2})$ are used as the inputs to the module. The output $E_z(i,j,t)$ is the field intensity value at the $(i,j)^{\text{th}}$ spatial location. The addition and subtraction operations can be realized using op-amps. The $\frac{1}{\mu\Delta xs}$ and $\frac{1}{\epsilon\Delta xs}$ operations can be implemented using integrator circuits.

4.2 APDA Method for Solving Maxwell's Equations

The APDA method is derived based on the standard FDTD method (Yee algorithm). As in the Yee algorithm, the electric and magnetic fields are defined at the integer (i.e., $i, i+1, \ldots$) and half-integer (i.e., $i+\frac{1}{2}, i+\frac{3}{2}, \ldots$) spatial points, respectively, where $i \in \mathbb{Z}$. However, we only discretize the spatial variable, while keeping the time variable continuous; this is the main difference between the Yee algorithm and the proposed method. Thus, we first approximate both time and spatial derivatives in the 1-D TM mode Maxwell's equations (given in (4.2)) using second-order centered differences without explicitly discretizing the time variable. Rearranging the terms,

the update equations become [110–115]

$$E_z(i,t) = E_z(i,t-\tau) + \frac{\tau}{\epsilon\Delta x}\left[H_y\left(i+\frac{1}{2},t-\frac{\tau}{2}\right) - H_y\left(i-\frac{1}{2},t-\frac{\tau}{2}\right)\right],$$
$$H_y\left(i+\frac{1}{2},t-\frac{\tau}{2}\right) = H_y\left(i+\frac{1}{2},t-\frac{3\tau}{2}\right) + \frac{\tau}{\mu\Delta x}\left[E_z(i+1,t-\tau) - E_z(i,t-\tau)\right], \tag{4.15}$$

where τ (the temporal step size) is an *analog* variable and the time variable t is continuous. The updating of E_z and H_y are staggered in the time dimension by $\tau/2$. The stability condition for the Yee algorithm with second-order temporal differences is $\Delta T \leq \frac{\Delta x}{c}$ [32, 34, 41, 42]; the same condition applies to the proposed analog computing method, i.e., $\tau \leq \tau_{max} = \frac{\Delta x}{c}$. The application of the Laplace transformation on (4.15) leads to

$$\bar{E}_z(i,s) = \bar{E}_z(i,s)\,e^{-s\tau} + \frac{\tau}{\epsilon\Delta x}\left[\bar{H}_y\left(i+\frac{1}{2},s\right) - \bar{H}_y\left(i-\frac{1}{2},s\right)\right]e^{\frac{-s\tau}{2}},$$
$$\bar{H}_y\left(i+\frac{1}{2},s\right)e^{\frac{-s\tau}{2}} = \bar{H}_y\left(i+\frac{1}{2},s\right)e^{\frac{-3s\tau}{2}} + \frac{\tau}{\mu\Delta x}\left[\bar{E}_z(i+1,s) - \bar{E}_z(i,s)\right]e^{-s\tau}. \tag{4.16}$$

Similarly, the magnetic field update equation in (4.16) can be expressed as

$$\bar{H}_y\left(i+\frac{1}{2},s\right) = \bar{H}_y\left(i+\frac{1}{2},s\right)e^{-s\tau} + \frac{\tau}{\mu\Delta x}\left[\bar{E}_z(i+1,s) - \bar{E}_z(i,s)\right]e^{\frac{-s\tau}{2}}. \tag{4.17}$$

The resulting CT update equations (the electric and magnetic field update equations given in (4.16) and (4.17), respectively) are used to design analog circuits that can compute CT solutions to Maxwell's equations.

4.2.1 Circuit Theory

Two separate modules need to be implemented based on the two update equations: one produces the electric field at integer spatial points, while the other produces the magnetic field at the half spatial grid points. Fig. 4.5(a) shows the analog circuit

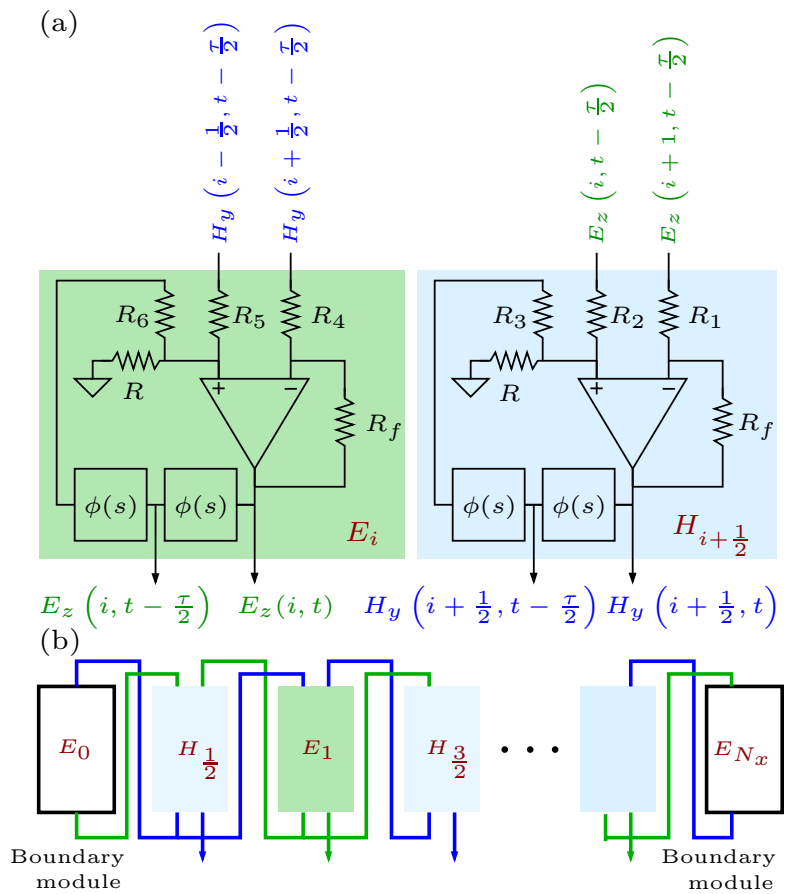

Figure 4.5: (a) The electric and magnetic field-based internal analog modules for solving Maxwell's equations using APDA method. (b) The systolic array architecture of the interconnected internal modules.

architectures for these modules. Consider the circuit on the left-hand side, which is the electric field IM that computes the solution E_z. It is fed by the outputs of magnetic field IMs at the neighboring spatial points ($i-\frac{1}{2}$ and $i+\frac{1}{2}$). It includes two continuous-time delays of value $\tau/2$, each of which has the Laplace domain representation $e^{-s\tau/2}$. It has been shown that the continuous-time delay $e^{-s\tau}$ can be approximated as $\left[\frac{1-\frac{s\tau}{2m}}{1+\frac{s\tau}{2m}}\right]^m$ with an accuracy that increases with m. Accurate time delays can be realized using a cascade of $m>1$ APFs ($m=3$ is required to approximate a true delay within 1% error) [110–116]. Thus, the delay $\tau/2$ is approximated as $e^{-s\tau/2}\approx\phi(s)$, where $\phi(s)=\left(\frac{1-s\tau/12}{1+s\tau/12}\right)^3$ is a third-order APF

with a group delay of $\tau/2$ (i.e., three cascaded first order systems each having an individual group delay of $\tau/6$). In addition, the same APF $\phi(s)$ is used to delay the adjacent magnetic fields by $\frac{\tau}{2}$ in time before they are fed into the electric field IM. A delay of τ is obtained by cascading two such APFs, as shown in Fig. 4.5(a), while summing and scaling operations are implemented using an op-amp. The magnetic field module that computes the solution for H_y can be designed by following the same procedure, and is shown on the right-hand side of Fig. 4.5(a). The complete solution of Maxwell's equations over the whole spatial domain is computed by interconnecting the IMs in a systolic array architecture as shown in Fig. 4.5(b). Note that this procedure is equivalent to a digital FDTD implementation, but with the unit time delays ΔT replaced by analog APFs that provide a group delay of τ.

Different boundary conditions can be simulated by modeling the governing equation at the boundary. A PEC at the right boundary can be simulated by grounding the $E_z(i+1, t-\frac{\tau}{2})$ input of the magnetic field IM at spatial point $i = N_x - \frac{1}{2}$ such that the simulation grid ends with an electric boundary at at $i = N_x$. Similarly, a PMC can be simulated by grounding the $H_y(i+1, t-\frac{\tau}{2})$ input of the electric field IM at spatial point $i = (N_x - 1)$ such that the simulation grid ends with a magnetic boundary at $i = N_x - \frac{1}{2}$. In order to realize a radiation boundary condition, we consider the Mur boundary condition given in (4.10). The approximation of the spatial and temporal derivatives using the second-order centered differences at spatial point $i = \left(N_x - \frac{1}{2}\right)$ (just inside the boundary) and time $t - \frac{\tau}{2}$ leads to

$$E_z\left(N_x - \frac{1}{2}, t\right) - E_z\left(N_x - \frac{1}{2}, t - \tau\right) + K\left[E_z\left(N_x, t - \frac{\tau}{2}\right) - E_z\left(N_x - 1, t - \frac{\tau}{2}\right)\right] = 0, \tag{4.18}$$

where $K = \frac{c\tau}{\Delta x} \leq 1$ [32, 34, 41, 42]. However, we do not define the electric field at half space-time points. Thus, a simple average, which is based on the electric fields at adjacent integer spacial points, is employed to approximate each term, such that

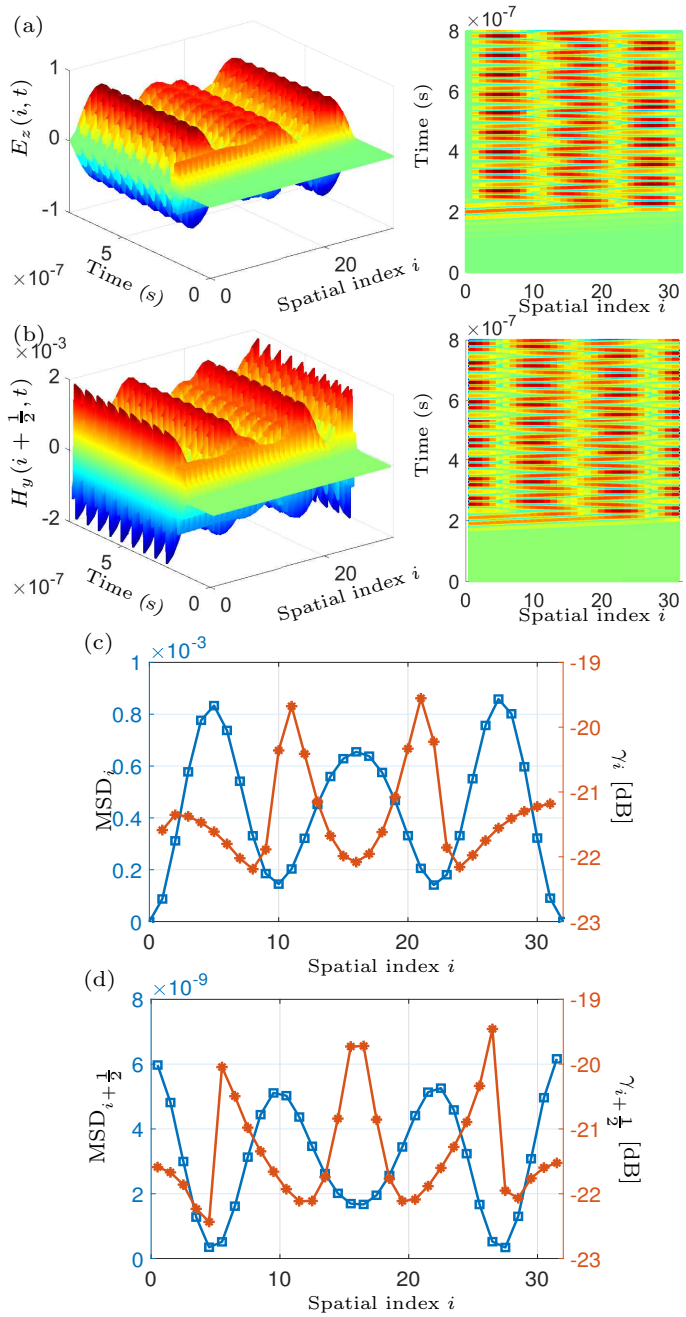

Figure 4.6: Ideal simulation results obtained from the APDA Maxwell's equations solver (left: 3-D view, right: top view): (a) the electric field E_z and (b) the magnetic field H_y. The right boundary was simulated as a PEC, while the left boundary was excited using a GMC electric field at 50 MHz. The deviation of the continuous-time solution compared to the FDTD numerical solution (MSD and γ) for (c) the electric field, and (d) the magnetic field.

$E_z\left(N_x - \frac{1}{2}, t\right) = \frac{E_z(N_x,t)+E_z(N_x-1,t)}{2}$. The resulting update equation is

$$E_z\left(N_x, t\right) = E_z\left(N_x - 1, t - \tau\right) + \left[\frac{1-K}{1+K}\right]\left[E_z\left(N_x, t-\tau\right) - E_z\left(N_x - 1, t\right)\right]. \tag{4.19}$$

The application of the LT on (4.19) leads to

$$\bar{E}_z\left(N_x, s\right) = \bar{E}_z\left(N_x - 1, s\right)e^{-s\tau} + \left[\frac{1-K}{1+K}\right]\left[\bar{E}_z\left(N_x, s\right)e^{-s\tau} - \bar{E}_z\left(N_x - 1, s\right)\right]. \tag{4.20}$$

The corresponding radiation boundary module can be realized using an architecture similar to the electric field IM shown in Fig. 4.5 (a).

4.2.2 Ideal Cadence Simulations and Comparative Analysis

The proposed APDA-based AC that solves Maxwell's equations was simulated using ideal circuit elements in Cadence Spectre. The IMs use active APFs realized as a cascade of three first-order sections [121]. A spatial grid of 65 points (including half spatial points) was selected for the simulation, with the left ($i = 0$) and right ($i = N_x$) boundaries defined using electric fields. The maximum frequency f_{max} was selected as 100 MHz, while the spatial step size $\Delta x = \frac{\lambda_{min}}{10}$ was similar to the CTLD-based simulation. Moreover, $\tau = 318$ ps was used to satisfy the stability condition $\tau \leq \tau_{max} = \frac{\Delta x}{c}$, resulting in a group delay of $\tau/2 = 159$ ps for each APF. Note that in our simulations, $\Delta x = 0.3$ m. Thus, $\tau_{max} = 1000$ ps.

Figs. 4.6(a) and (b) show the spatio-temporal variation of the electric and magnetic field solutions when a GMC electric field with a center frequency of 50 MHz (identical to that used in the CTLD-based simulations) is applied to the left boundary, while a PEC condition is applied to the right boundary. Figs. 4.6(c) and (d) use MSD and γ to quantify deviations between the APDA-based simulation results and those from standard FDTD simulations. Fig. 4.7 shows the corresponding results

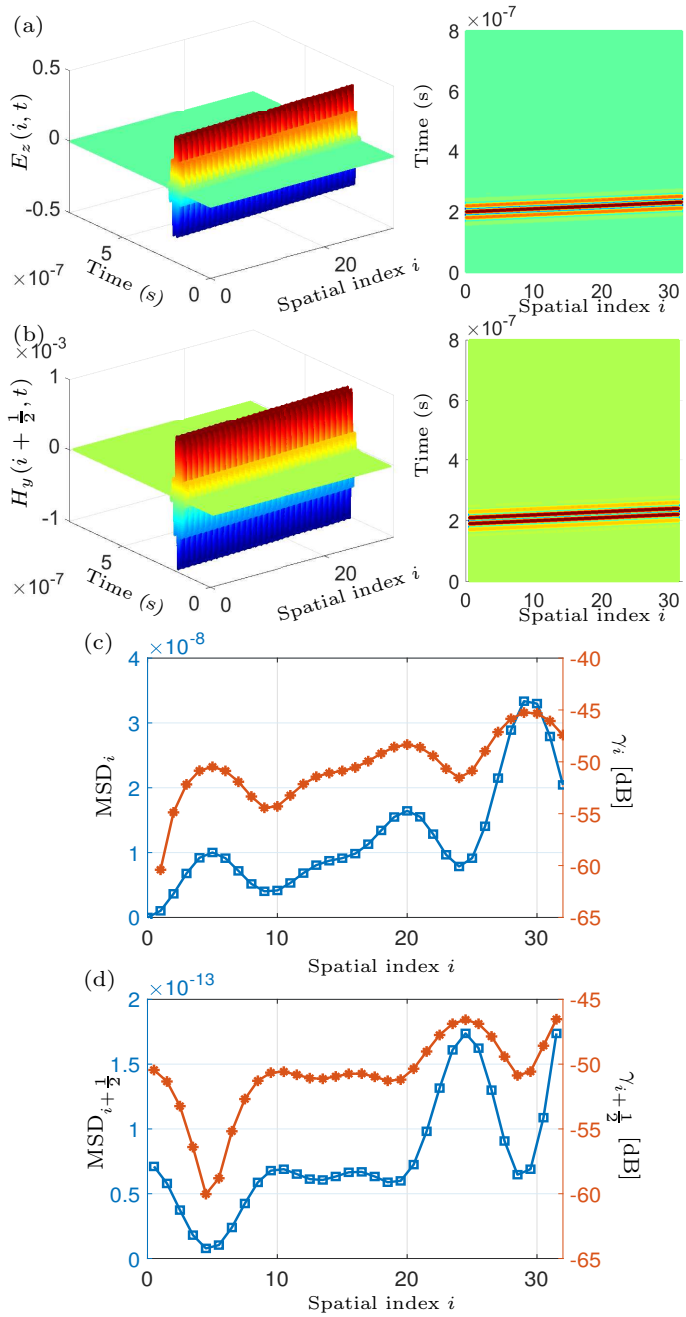

Figure 4.7: Ideal simulation results obtained from the APDA Maxwell's equations solver (left: 3-D view, right: top view): (a) the electric field E_z and (b) the magnetic field H_y. The right boundary was simulated as a radiation boundary, while the left boundary was excited using a GMC electric field at 50 MHz. The deviation of the continuous-time solution compared to the FDTD numerical solution (MSD and γ) for (c) the electric field, and (d) the magnetic field.

when the right boundary is simulated as a radiation boundary. The proposed APDA method is able to compute the solutions of Maxwell's equations with a difference less than -19 dB in terms of γ (assuming ideal circuit models).

CHAPTER 5

CONTINUOUS-TIME SOLUTION OF WAVE EQUATION

This chapter extends the CTLD and APDA algorithms towards designing ACs for solving the wave equation. 1-D and 2-D wave equations are considered. The ACs have been simulated using ideal circuits for different wave propagation scenarios and boundary conditions. The accuracy of their performances are estimated by comparing with the standard FDTD solutions. Electromagnetic wave propagation is described using the wave equation, which can be easily derived from the source-free Maxwell's equations given in (4.1). Since the field vectors **E** and **H** satisfy the same wave equation, a simplified analog circuit model can be used to compute the corresponding CT solution. In particular, consider the 1-D version, which has one spatial variable and the time variable t. If the function $E_z(x,t)$ represents the electric field intensity of a propagating wave at position x and time t, then $E_z(x,t)$ satisfies the PDE [119, 120]

$$\frac{1}{c^2}\frac{\partial^2 E_z(x,t)}{\partial t^2} = \frac{\partial^2 E_z(x,t)}{\partial x^2}. \tag{5.1}$$

Consider a spatial domain $0 \leq x \leq L_x$, which is discretized into $N_x + 1$ spatial points, where $\Delta x = \frac{L_x}{N_x}$ is the spatial step size. The electric field E_z is defined only at integer grid points i ($i \in \{0, 1, 2, \ldots, N_x\}$) and we do not need to define any variable at half grid points as the Maxwell's equation solver. Fig. 5.1 shows the spatial grid of the analog wave equation solver.

5.1 CTLD Method for Solving Wave Equation

Application of the finite difference approximation along the spatial dimension, followed by LT along the time dimension (i.e., the CTLD method) on (5.1), leads to a

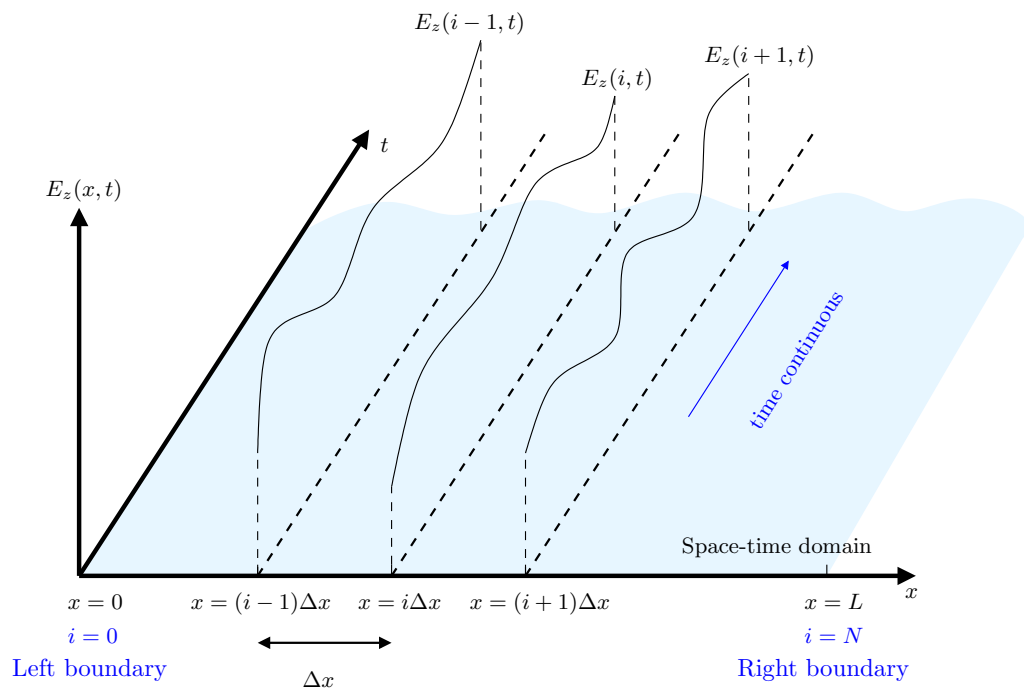

Figure 5.1: The space-time domain of the wave equation solver.

SDTC update equation [29–31, 106–109]

$$\bar{E}_z\left(i,s\right)=\frac{\bar{E}_z\left(i+1,s\right)+\bar{E}_z\left(i-1,s\right)}{2\left(As^2+1\right)}, \tag{5.2}$$

where $A=\frac{\Delta x^2}{2c^2}$ and spatial derivatives have been approximated using second-order centered differences [34]. Also, zero initial conditions are considered for the LT. Consider the IM that computes the solution $E_z\left(i,t\right)$ at spatial grid point i. According to (5.2), the IM is fed by its neighboring solutions $E_z\left(i-1,t\right)$ and $E_z\left(i+1,t\right)$. The summation operation in the numerator of the update equation can be realized using an op-amp, while the $\frac{1}{As^2+1}$ operation can be realized using a series inductor (L) and capacitor (C). Fig. 5.2(a) shows an analog implementation of the IM that results in $A=LC$. The Sallen-Key technique [118] or the Bruton transformation [106, 107] can be used to simplify this circuit by replacing the LC network with an equivalent active network [109]. Such a network can also be easily tuned, thus allowing A to be changed between IMs to simulate inhomogeneous media. Finally, the continuous-

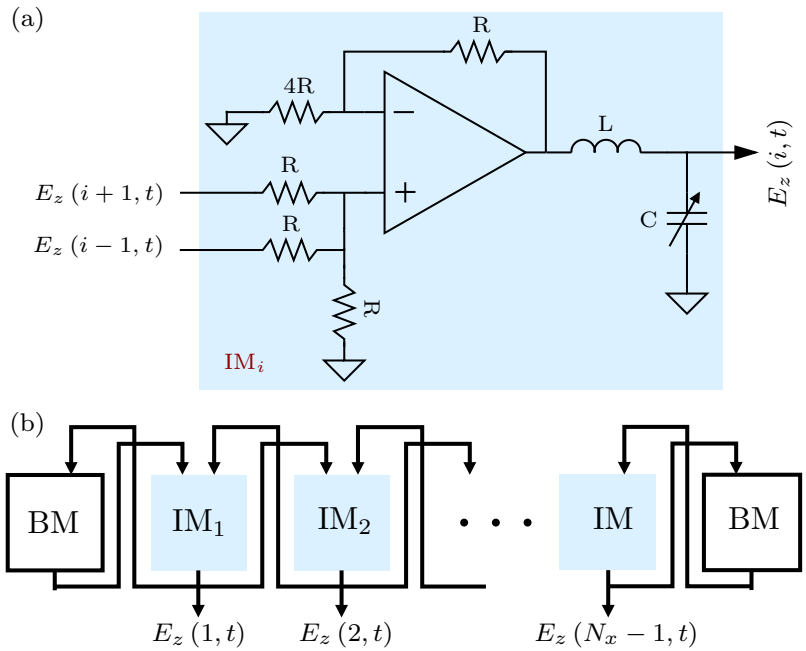

Figure 5.2: (a) An analog circuit that can be used as the internal module (IM) of the 1-D wave equation solver. (b) Systolic array architecture to compute the solution of the 1-D wave equation.

time solution over the whole spatial domain can be computed by interconnecting the IMs in a systolic array architecture, as shown in Fig. 5.2(b). However, note that the output of each IM is measured across the capacitor and so must be buffered before feeding the neighboring IMs.

At each boundary ($i = 0$ and $i = N_x$), different computing modules need to be implemented. For the proposed analog wave equation solver, we consider three different boundary modules, which are based on the Dirichlet, Neumann, and radiation boundary conditions. Consider the left boundary ($i = 0$). Since the Dirichlet condition specifies the solution $E_z(0, t)$ at the boundary, we do not need any additional hardware to implement it. In particular, a signal source $f(t)$ can directly excite the $E_z(i-1, t)$ input of IM_1 (at $i = 1$) with the desired solution. A hard reflector is a special case of the Dirichlet boundary condition. As an example, it can be imposed

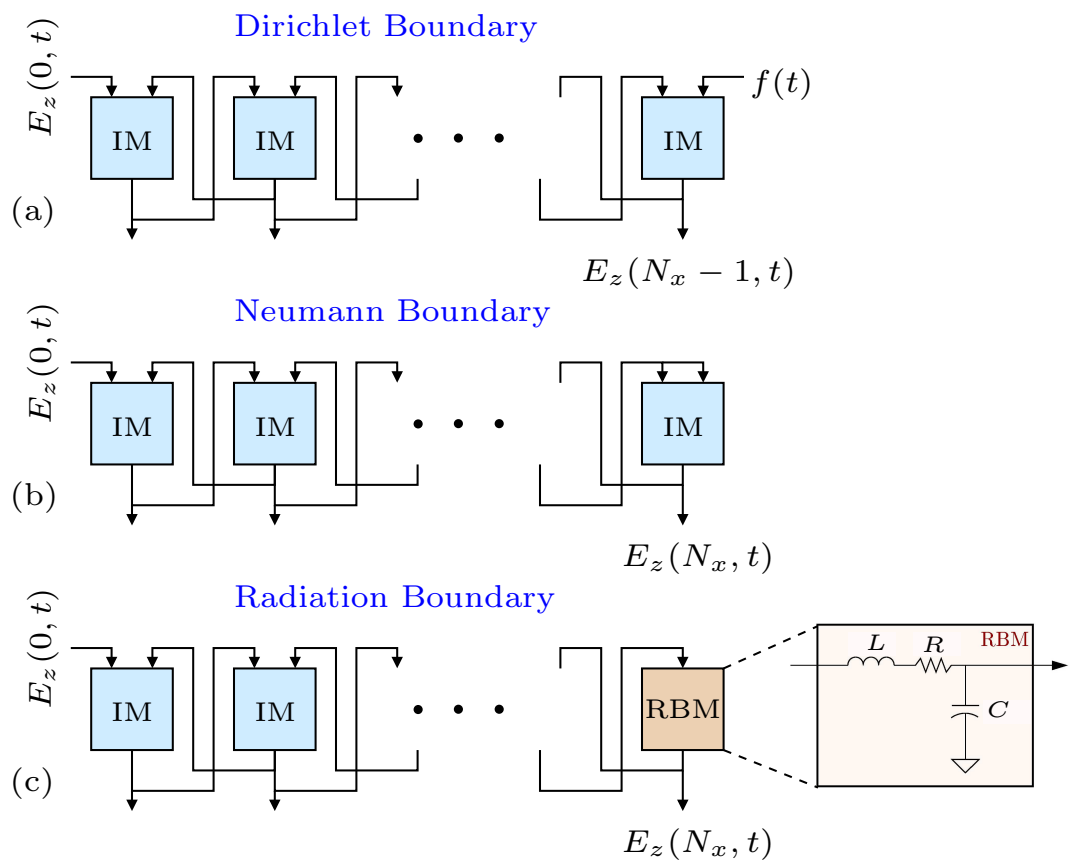

Figure 5.3: Realization of different boundary conditions at the right boundary: (a) Dirichlet, (b) Neumann, and (c) radiation boundaries.

on the right boundary by feeding a DC voltage source (i.e., $f(t) = \text{constant}$) to the $E_z(i+1,t)$ input of IM_{N_x-1} as shown in Fig. 5.3 (a).

The Neumann boundary condition specifies the value of the derivative of the solution at the boundary. Consider a situation where the field intensity $E_z(x,t)$ satisfies $\left.\frac{\partial E_z(x,t)}{\partial x}\right|_{x=N_x\Delta x} = 0$. Application of the CTLD method on the given Neumann condition leads to $\bar{E}_z(N_x+1,s) = \bar{E}_z(N_x-1,s)$. The SDTC update equation at $i = N_x$ can then be obtained as (by substituting in (5.2)).

$$\bar{E}_z(N_x,s) = \frac{\bar{E}_z(N_x-1,s)}{As^2+1}. \tag{5.3}$$

This function can be implemented by connecting the $E_z(i+1,t)$ and $E_z(i-1,t)$ inputs of an IM to each other as shown in Fig. 5.3 (b).

In order to obtain the SDTC update equation for the radiation boundary condition, consider the following Mur boundary condition [119, 120]

$$\frac{1}{c}\frac{\partial E_z(x,t)}{\partial t}|_{x=N_x\Delta x} + \frac{\partial E_z(x,t)}{\partial x}|_{x=N_x\Delta x} = 0. \tag{5.4}$$

Application of the CTLD method on (5.4) leads to

$$\bar{E}_z(N_x+1,s) = \bar{E}_z(N_x-1,s) - 2\frac{\Delta x}{c}s\bar{E}_z(N_x,s). \tag{5.5}$$

Here, a second order centered difference is employed to discretize the spatial domain partial derivative. The mixed domain expression at $i = N_x$ can then be obtained as (by substituting in (5.2))

$$\bar{E}_z(N_x,s) = \frac{2\bar{E}_z(N_x-1,s)}{\left(\frac{\Delta x}{c}\right)^2 s^2 + 2\frac{\Delta x}{c}s + 2}. \tag{5.6}$$

The radiation boundary module can be implemented by using a simple passive LRC circuit as shown in Fig. 5.3 (c) or its active implementation using, e.g., the Sallen-Key architecture [118]. The input to the module is the solution of the neighboring internal module IM_{N_x-1}. Figs. 5.3 (a-c) show the systolic array architectures when realizing Dirichlet, Neumann, and radiation boundary conditions, respectively, at the right boundary of the spatial domain.

5.1.1 Transfer Functions of the Analog Computer

In this section, the s-domain transfer functions of the AC that solves 1-D wave equation were obtained for each of the outputs, where input $E_z(0,t)$ was applied at $x = 0$. The resulting transfer functions can be used to analyze the stability of the AC. Consider a 1-D wave equation solver with seven spatial points (i. e.

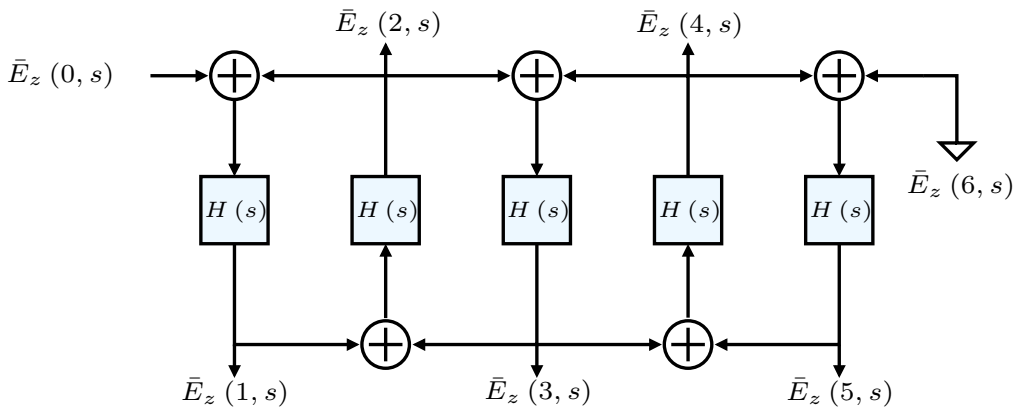

Figure 5.4: Signal flow graph of the analog 1-D wave equation solver when $N_x = 6$.

$N_x = 6$ and $i = 0, 1, 2, 3, 4, 5, 6)$. Spatial points $i = 0$ and $i = 6$ are the left and right boundaries, respectively. The left boundary is excited using a voltage source $E_z\,(0,t)$ and the right boundary is fixed at zero voltage (Dirichlet boundary condition). The signal flow graph of the analog solver is shown in Fig. 5.4. Here, $\bar{H}\,(s) = \frac{1}{2(As^2+1)}$, which is derived based on the update equation given in (5.2). The summation of the numerator is separated and represented using an adder in the figure. For the following calculations, A is selected as 1. $\bar{E}_z\,(0,s)$ is the Laplace domain representation of the input $E_z\,(0,t)$. The input-output relationships (s-domain transfer functions) of each output can then be obtained as

$$\begin{aligned}
\frac{\bar{E}_z\,(1,s)}{\bar{E}_z\,(0,s)} &= \frac{16s^8 + 64s^6 + 84s^4 + 40s^2 + 5}{2\,(s^2+1)\,(16s^8 + 64s^6 + 80s^4 + 32s^2 + 3)}, \\
\frac{\bar{E}_z\,(2,s)}{\bar{E}_z\,(0,s)} &= \frac{4s^4 + 2s^2 + 2}{(16s^8 + 64s^6 + 80s^4 + 32s^2 + 3)}, \\
\frac{\bar{E}_z\,(3,s)}{\bar{E}_z\,(0,s)} &= \frac{4s^4 + 8s^2 + 3}{2\,(s^2+1)\,(16s^8 + 64s^6 + 80s^4 + 32s^2 + 3)}, \\
\frac{\bar{E}_z\,(4,s)}{\bar{E}_z\,(0,s)} &= \frac{1}{(16s^8 + 64s^6 + 80s^4 + 32s^2 + 3)}, \\
\frac{\bar{E}_z\,(5,s)}{\bar{E}_z\,(0,s)} &= \frac{1}{2\,(s^2+1)\,(16s^8 + 64s^6 + 80s^4 + 32s^2 + 3)}.
\end{aligned} \tag{5.7}$$

When considering the denominator of each expression, it is clear that each of the individual systems, which correspond to each of the outputs, are stable and realizable using analog circuits.

5.1.2 Ideal Cadence Simulations

The proposed AC that solves the wave equation was simulated in Cadence Spectre using ideal circuit elements. Fig. 5.5 shows the results obtained for a spatial grid size of 65 ($N_x = 64$ and $i \in \{0, 1, 2, \ldots, 64\}$). The analog bandwidth $\mathrm{F}_{compute}$ of the system was selected as 100 MHz, resulting in a minimum wavelength of $\lambda_{min} = \frac{c}{\mathrm{F}_{compute}}$. The spatial step size is then set to $\Delta x = \frac{\lambda_{min}}{10}$. The left boundary is excited using a GMC field at 50 MHz (the same function used in earlier simulations). Fig. 5.5(a) shows the spatio-temporal results obtained when the right boundary is fixed at zero (thus implementing a hard boundary) by grounding the $E_z\,(i+1, t)$ input of IM_{63}. Both a 3-D view (left) and the top view (right) of the results are shown. Similarly, Figs. 5.5(b) and (c) show the results obtained with Neumann and radiation boundaries at the right boundary, respectively.

Figs. 5.6(a)-(c) compare the simulation errors of the three scenarios shown in Figs. 5.5(a) - (c), respectively. The proposed CTLD-based AC is capable of computing the 1-D wave equation with a difference less than -72 dB (defined using γ) when the circuits are assumed to be ideal.

5.1.3 Analog Computing for Two Media

In order to investigate the behavior of the proposed AC in different media, a situation with two media (two different wave propagation velocities) is considered. Note that the wave propagation velocity c of the medium is related with the value $A = \frac{\Delta x^2}{2c^2}$ of the internal modules. It can be changed by tuning the capacitor of the LC section of the IM (see Fig. 5.2(a)). The number of spatial points in the system is selected as 66. Spatial modules from 33 to 64 have half of the wave propagation speed as that of the spatial modules from 1 to 32 (i.e. internal modules 1 to 32 and 33 to 64 simulate

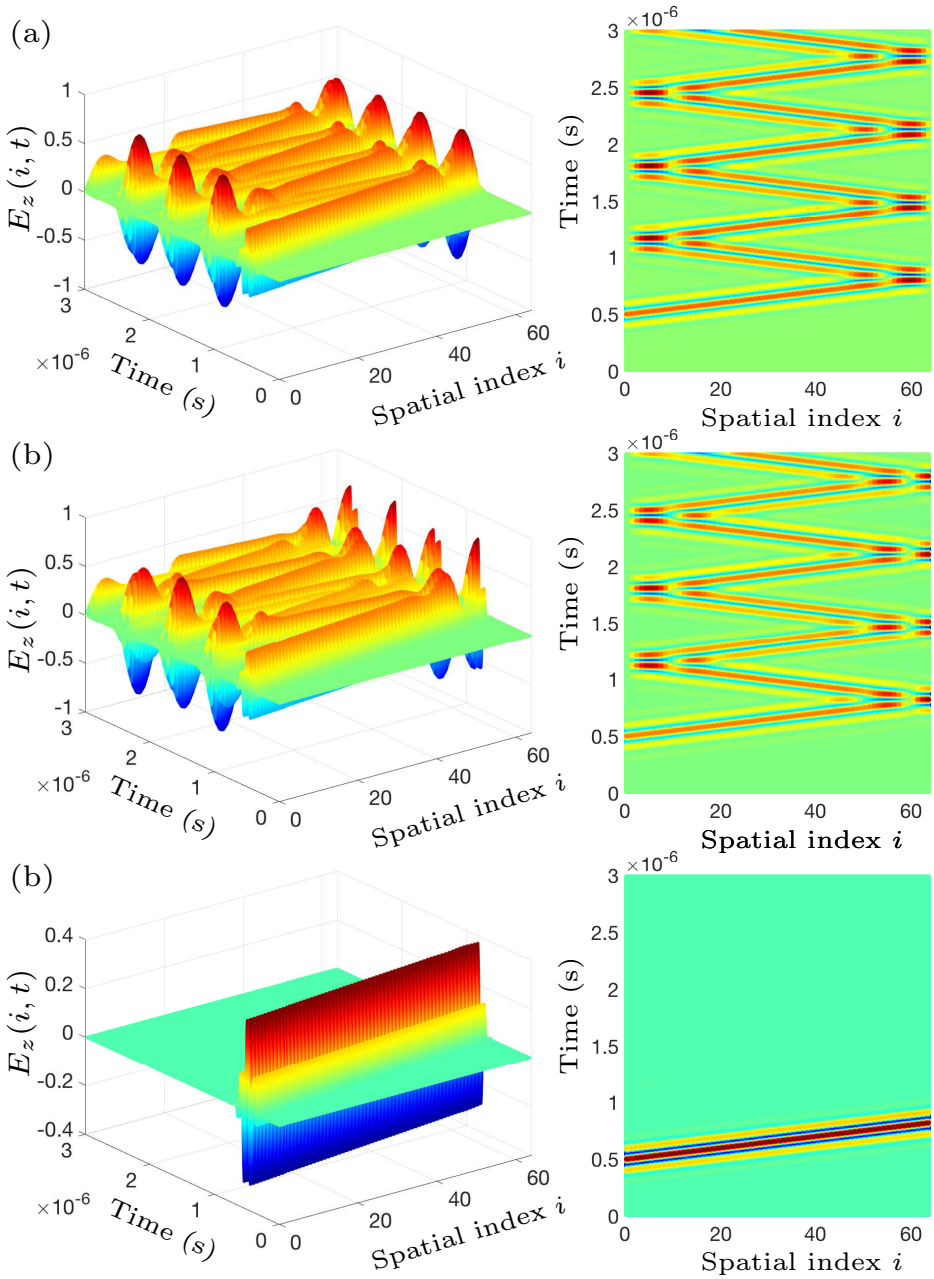

Figure 5.5: Simulation results obtained from the CTLD method-based 1-D analog wave equation solver (left: 3-D view, right: top view). The left boundary is excited using a GMC field at 50 MHz. The right boundary is realized as (a) a Dirichlet, (b) a Neumann, and (c) a radiation boundary.

wave propagation velocities c and $c/2$, respectively). Fig. 5.7 shows the Cadence results obtained for a Gaussian pulse with the radiation boundary condition at the

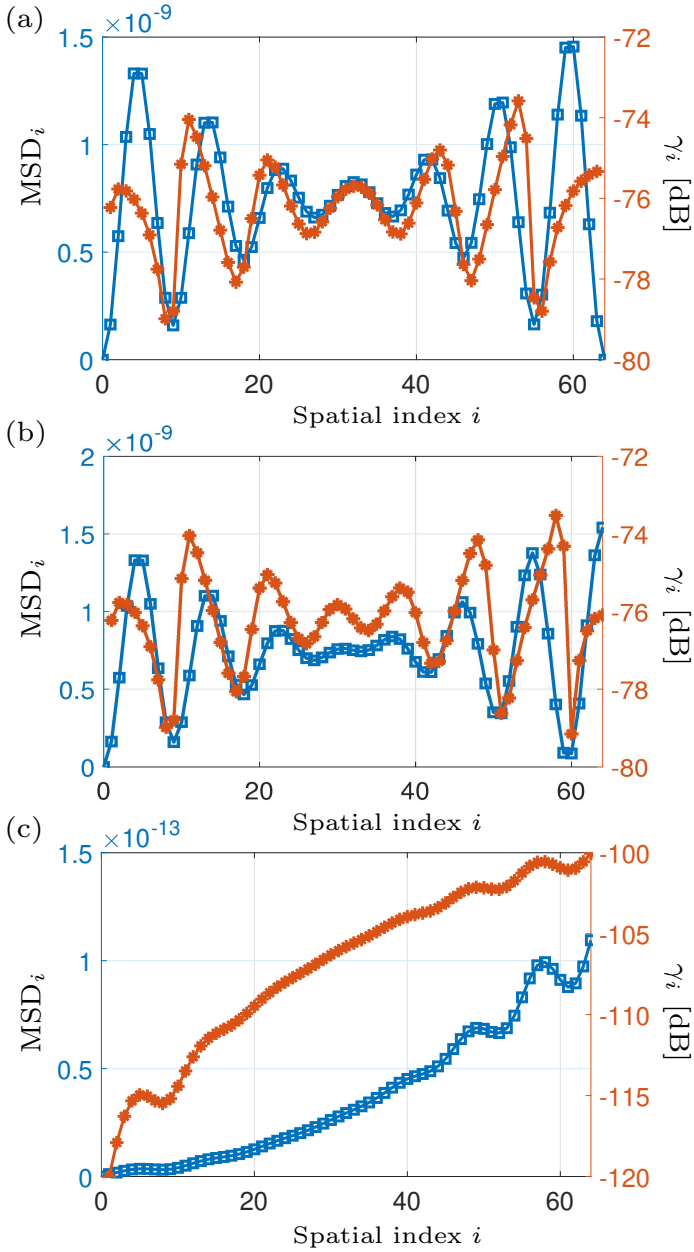

Figure 5.6: Comparison results (MSD_i and γ_i) corresponding to the three simulation scenarios shown in Figs. 5.5(a)-(c), respectively.

right boundary. The RLC values of the radiation boundary module are modified such that it accounts the velocity $c/2$ at the boundary. It is clear in Fig. 5.7 that

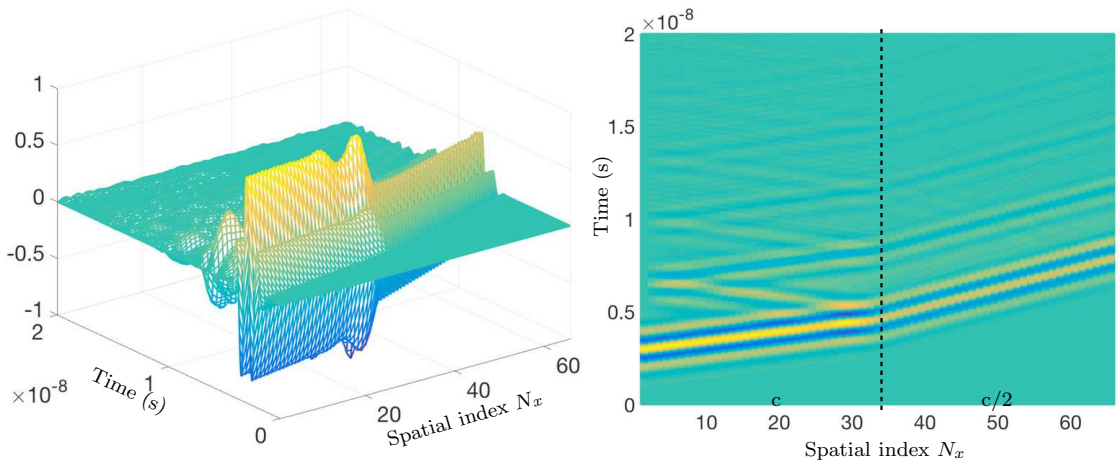

Figure 5.7: Behavior of the wave equation solver with two mediums. Cadence results obtained for a Gaussian pulse with the radiation boundary conditions (left: 3-D view, right: top view).

the proposed solver is able to provide information of the refraction at the medium boundary along with the partial refraction.

5.1.4 CT Solution of Damped Wave Equation

Mathematical models that we developed for solving the lossless wave equation can be extended to lossy mediums. The lossy (damped) wave equation can be expressed as,

$$\frac{1}{c^2}\frac{\partial^2 E_z\left(x,t\right)}{\partial t^2} + k\frac{\partial E_z\left(x,t\right)}{\partial t} = \frac{\partial^2 E_z\left(x,t\right)}{\partial x^2}, \tag{5.8}$$

where k is the damping coefficient of the medium. In general, losses in a medium can be modeled by a small number of odd-order terms added to the wave equation. Here, we only consider the simplest case where losses are approximated by a first order term (the resistive force is directly proportional to transverse velocity). Application of the direct LT method on (5.8) leads to the following mixed domain transfer

functions for internal spatial points (with zero initial conditions)

$$\bar{E}_z(i,s) = \frac{\bar{E}_z(i+1,s) + \bar{E}_z(i-1,s)}{\frac{\Delta x^2}{c^2}s^2 + k\Delta x^2 s + 2}. \tag{5.9}$$

The second order centered difference is employed to approximate the spatial partial derivative. The $\frac{1}{\frac{\Delta x^2}{c^2}s^2+k\Delta x^2 s+2}$ operation can be implemented using a LRC circuit. Scaling and summing operations can be realized using op-amp circuits.

5.1.5 Higher-Order Approximations

The accuracy of the CT solution can be improved by employing higher order approximations to the partial derivatives in the spatial dimension. As an example, we employ the fourth order accurate centered difference to approximate the $\frac{\partial^2 E_z}{\partial x^2}$ term in (5.1) as

$$\begin{aligned}\frac{\partial^2 E_z(i,t)}{\partial x^2} \approx & \frac{1}{12\Delta x^2}\left[-E_z(i+2,t) + 16E_z(i+1,t)\right. \\ & \left. -30E_z(i,t) + 16E_z(i-1,t) - E_z(i-2,t)\right].\end{aligned} \tag{5.10}$$

We then apply the Laplace transform with respect to the time variable t, which leads to a mixed domain transfer function

$$\begin{aligned}&\bar{E}_z(i,s) = \\ &\frac{1}{30\left(A_h s^2+1\right)}\left[-\bar{E}_z(i+2,s) + 16\bar{E}_z(i+1,s) + 16\bar{E}_z(i-1,s) - \bar{E}_z(i-2,s)\right],\end{aligned} \tag{5.11}$$

where $A_h = \frac{12\Delta x^2}{30c^2}$. The analog circuit that computes the solution can then be realized using op-amp circuits. The corresponding boundary modules need to be designed by approximating the spatial derivatives of the governing equations using higher order approximations.

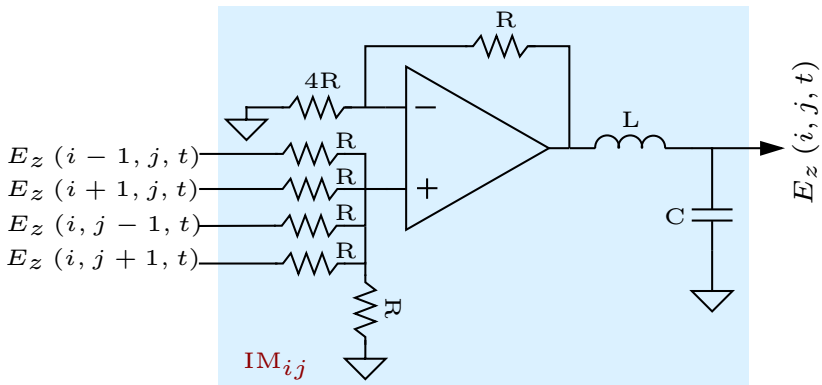

Figure 5.8: Analog circuit of the internal module that compute the solution of the 2-D wave equation.

5.1.6 CT Solution of 2-D Wave Equation

The analog computation methods that we developed to compute the CT solution for the 1-D wave equation can be extended to additional spatial dimensions. The general form of the 2-D wave equation is

$$\frac{1}{c^2}\frac{\partial^2 E_z\,(x,y,t)}{\partial t^2} = \frac{\partial^2 E_z\,(x,y,t)}{\partial x^2} + \frac{\partial^2 E_z\,(x,y,t)}{\partial y^2}, \tag{5.12}$$

which describes the field intensity $E_z\,(x,y,t)$ of a propagating wave at position (x,y) and time t. A rectangular 2-D spatial domain is considered, where $0 \leq x \leq L_x$ and $0 \leq y \leq L_y$. The spatial domain is first discretized into N_x+1 and N_y+1 spatial points along the x and y directions, respectively; the corresponding spatial step sizes are $\Delta x = \frac{L_x}{N_x}$ and $\Delta y = \frac{L_y}{N_y}$. The spatial indexes $i = 0,1,2,...,N_x$ and $j = 0,1,2,...,N_y$ represent the x and y directions, respectively. For simplicity, we consider the case where $\Delta x = \Delta y = \Delta d$ and apply the CTLD method to find a continuous-time solution. Applying finite differences to the spatial derivatives along both x and y, followed by the LT along t, leads to

$$\begin{aligned}\bar{E}_z\,(i,j,s) =& \frac{1}{4\,(A_{2D}s^2+1)}\left[\bar{E}_z\,(i+1,j,s) + \bar{E}_z\,(i-1,j,s)\right.\\ &\left.+\bar{E}_z\,(i,j+1,s) + \bar{E}_z\,(i,j-1,s)\right],\end{aligned} \tag{5.13}$$

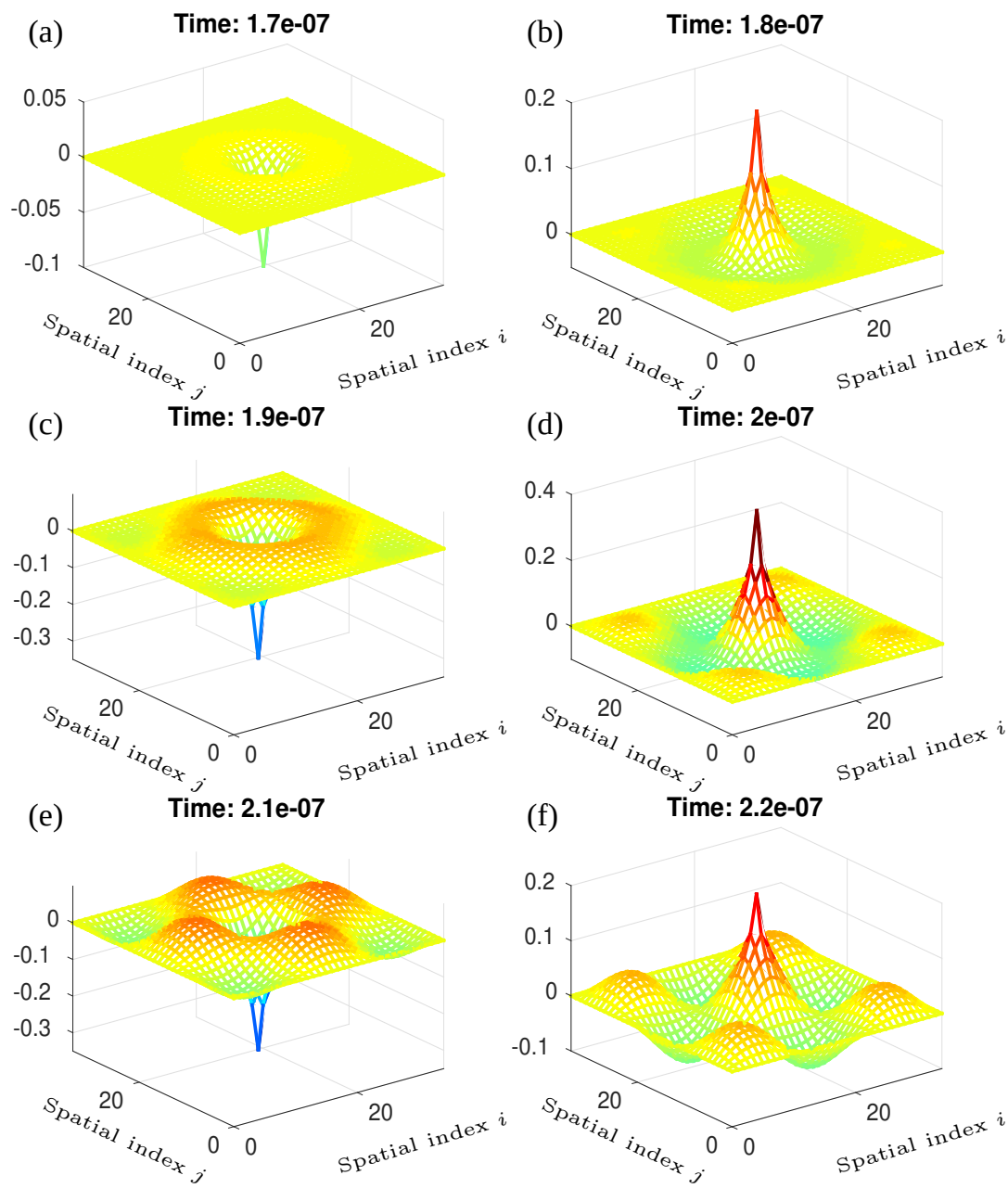

Figure 5.9: Simulation results obtained from the 2-D analog wave equation solver realized using the CTLD method. The spatial point (17,17) is excited using a GMC field at 50 MHz, while the four boundaries are fixed at zero.

where $A_{2D} = \frac{\Delta d^2}{4c^2}$, zero initial conditions are considered for the LT, and both spatial derivatives are approximated using second-order centered differences. The $\frac{1}{A_{2D}s^2+1}$ operation can be implemented using a LC circuit that can be made tunable in order

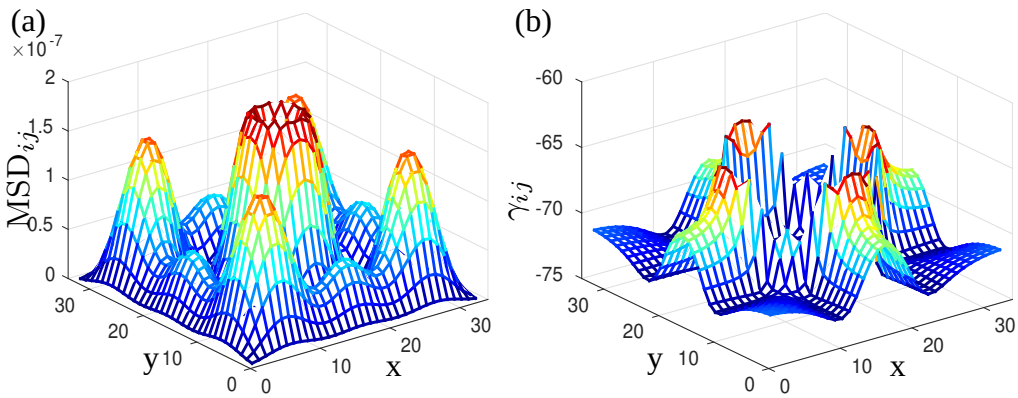

Figure 5.10: Comparison results (MSD_i and γ_i) corresponding to the simulation scenario shown in Fig. 5.9.

to simulate inhomogeneous media, while scaling and summing operations can again be realized using op-amps. Fig. 5.8 shows an analog circuit that can be used to solve (5.13) at internal spatial points. This IM is fed by the electric field solutions from the four neighboring modules. The CT 2-D solution to the wave equation can now be computed over the whole spatial grid by interconnecting IMs in a 2-D systolic array architecture.

As an example, a spatial grid of 35×35 spatial points ($N_x = 34$ and $N_y = 34$) was used to simulate the AC that computes the 2-D analog wave equation (using ideal circuit components). The boundaries were fixed at zero by grounding the corresponding inputs of IMs located just inside the boundary. A GMC electric field at 50 MHz is used to excite the $(17, 17)$ spatial point. Fig.5.9 shows the intensity patterns obtained in the x-y spatial domain at different times (time increases from (a) to (f)). Figs. 5.10 (a) and (b) show the corresponding comparison results (MSD_i and γ_i, respectively). The proposed AC is capable of computing the solution with a difference less than -60 dB (as defined using the comparison metric γ) when the circuits are assumed to be ideal.

Two example simulations of the AC are recorded as time varying signals. Captured data is used to create videos that show the wave propagation in the 2-D space-time domain. Followings are brief descriptions of each of the simulation setups along with the links to the videos.

Example 1: Spatial grid with 33×33 elements ($N_x = 32$ and $N_y = 32$) is selected for the simulation. Gaussian pulse is applied to $i = 16, j = 16$ spatial point. All boundaries are fixed at zero. Link to the video: https://www.dropbox.com/s/zcbke3zfp4t6yac/Video_1.mov?dl=0

Example 2: Spatial grid with 32×16 elements ($N_x = 31$ and $N_y = 15$) is selected for the simulation, where $i = 0, 1, 2, ..., N_x$ and $j = 0, 1, 2, ..., N_y$. Gaussian pulse is applied to left boundary. Right boundary is simulated as a open boundary. All other boundaries are simulated as soft reflectors. Link to the video: https://www.dropbox.com/s/prk9uds3bna8fwi/Video_2.mov?dl=0

5.2 APDA Method for Solving Wave Equation

The APDA method can also be employed to obtain a mathematical model that can solve the 1-D wave equation in the CT domain. Approximating both spatial and time derivatives in (5.1) using the second order centered finite differences (while keeping the time variable continuous) and applying the LT along the time dimension (with zero initial conditions) lead to

$$\begin{aligned} \bar{E}_z(i,s) &= K^2 \left[\bar{E}_z(i+1,s) + \bar{E}_z(i-1,s)\right] e^{-\tau s} \\ &+ 2\left(1-K^2\right) \bar{E}_z(i,s)\, e^{-\tau s} - \bar{E}_z(i,s)\, e^{-2\tau s}, \end{aligned} \tag{5.14}$$

where $K = \frac{c\tau}{\Delta x} \leq 1$. The resulted mixed domain transfer function is used to realize the internal analog module that computes the solution for internal locations. The Laplace domain representation $e^{-s\tau}$ is approximated using an all-pass filter $\phi'(s)$

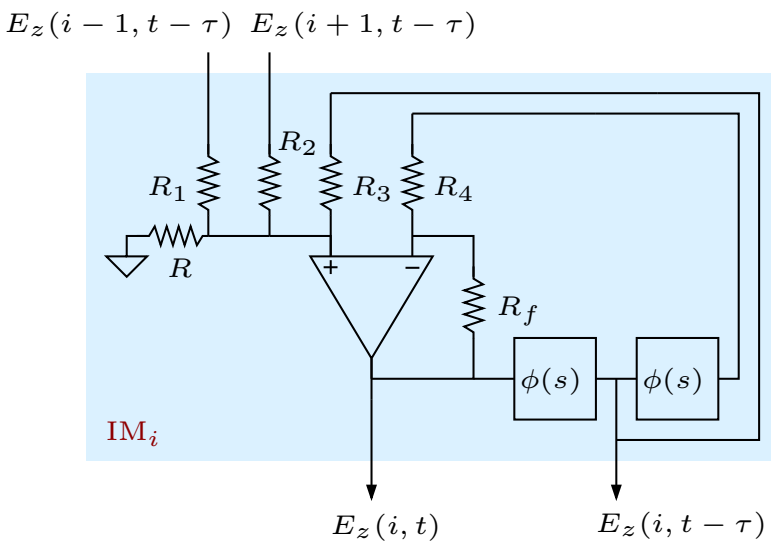

Figure 5.11: The analog circuit for realizing the internal module of the 1-D wave equation solver for the all-pass filter method.

with a group delay of τ. The analog circuit architecture of the all-pass filter-based IM is shown in Fig. 5.11. The complete CT solution to the 1-D wave equation (over the whole spatial grid) is obtained by interconnecting IMs in an analog array architecture.

A fixed boundary at $i = N_x$ can be simulated by fixing the $E_z(i+1, t-\tau)$ input of the IM at spatial point $i = N_x - 1$. A Neumann boundary at $i = N_x$ can be simulated by connecting the two inputs to the IM at $i = N_x$ (as similar to the CTLD method [29, 31]). Consider the radiation boundary condition given in (5.4). The application of the centered finite differences on both spatial and time dimensions and Laplace transformation along the time dimension lead to

$$K\bar{E}_z(N_x+1, s)\, e^{-\tau s} = K\bar{E}_z(N_x-1, s)\, e^{-\tau s} - \bar{E}_z(N_x, s) + \bar{E}_z(N_x, s)\, e^{-2\tau s}. \tag{5.15}$$

By substituting the resulting expression in (5.14), the mixed domain transfer function for the radiation boundary at $i = N_x$ can be obtained as

$$\begin{aligned}(1+K)\,\bar{E}_z\,(N_x,s) &= 2\left(1-K^2\right)\bar{E}_z\,(N_x,s)\,e^{-\tau s}\\ &+ (K-1)\,\bar{E}_z\,(N_x,s)\,e^{-2\tau s} + 2K^2\bar{E}_z\,(N_x-1,s)\,e^{-\tau s}.\end{aligned} \tag{5.16}$$

The corresponding radiation boundary module can be realized using an architecture similar to the IM shown in Fig. 5.11. However, the radiation boundary module only requires the input $E_z\,(i-1,t)$ for the computation.

5.2.1 Ideal Cadence Simulations

The APDA-based AC that solves the wave equation was simulated using ideal circuit elements. The analog bandwidth $\mathrm{F}_{compute}$ was selected as 100 MHz, and the spatial step size $\Delta x = \frac{\lambda_{min}}{10}$. The group delay of the APF τ was 1.59 ns, and the temporal step size ΔT of the FDTD simulation (used for comparison purposes) was also selected as 1.59 ns. Figs. 5.12(a)-(c) show the spatio-temporal solutions obtained from the proposed analog wave equation solver for a GMC excitation at the left boundary when the right boundary is set to Dirichlet, Neumann, or radiation boundary conditions, respectively. Fig. 5.13 shows how much the obtained results deviate from the standard FDTD simulations in terms of both MSD_i and γ_i. The proposed APDA method is able to compute the solution of the 1-D wave equation with a difference less than -24 dB (as defined using γ) when the circuits are assumed to be ideal.

5.3 CTLD Method vs APDA Method

Based on the comparison results of the proposed analog solvers (MSD and γ), it is clear that the CTLD method is more accurate than the APDA method for all

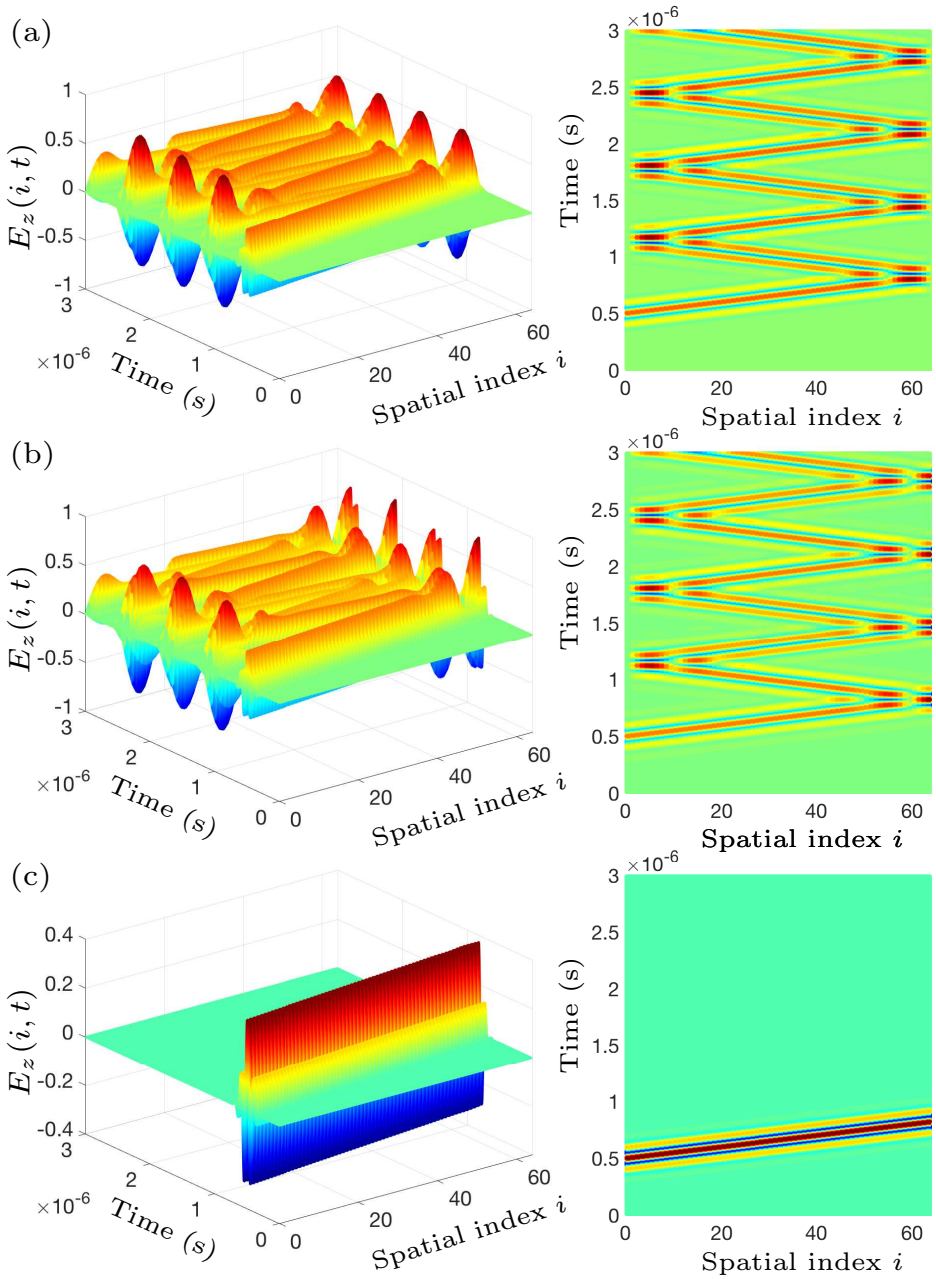

Figure 5.12: Ideal simulation results obtained from the APDA-based 1-D analog wave equation solver (left: 3-D view, right: top view). The left boundary is excited using a GMC field at 50 MHz, while the right boundary is simulated as (a) a Dirichlet, (b) a Neumann, and (c) a radiation boundary.

simulation scenarios. However, note that the performance of APDA-based solvers is mainly limited by the ability of the APFs to provide accurate gain and group delay at

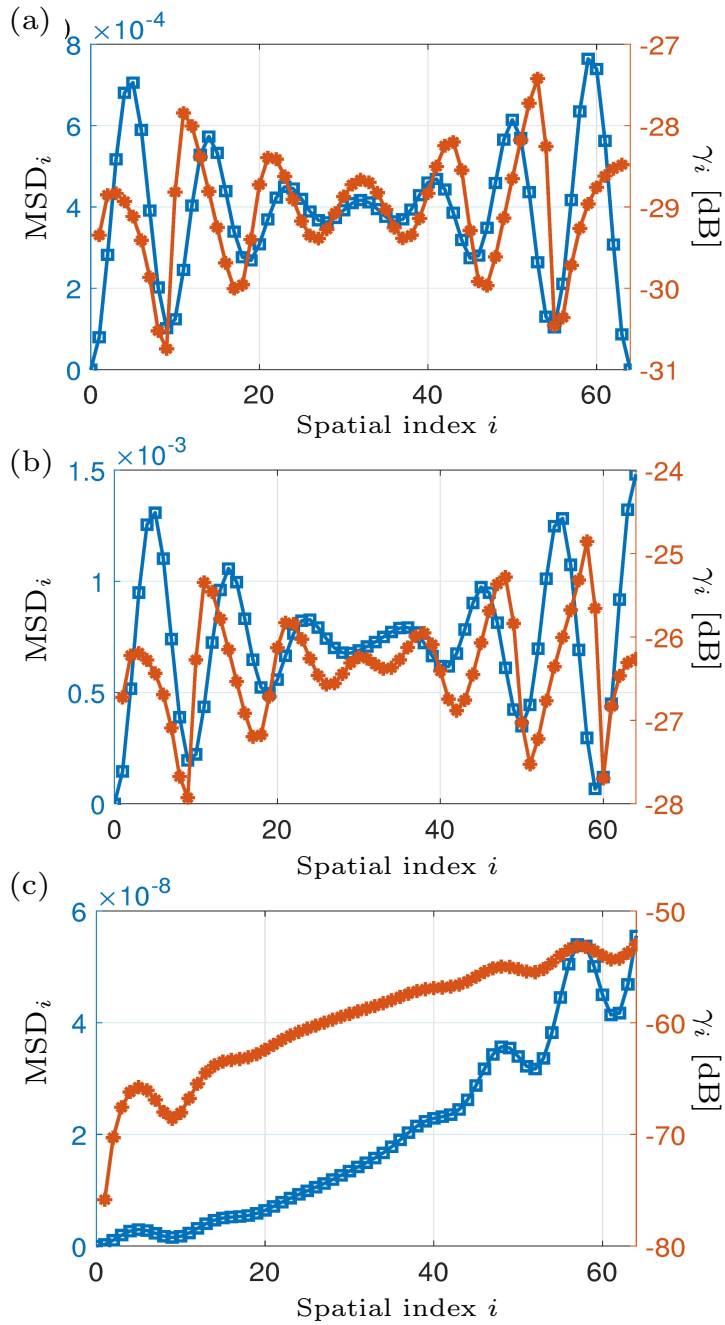

Figure 5.13: Comparison results (MSD_i and γ_i) correspond to the three simulation scenarios shown in Figs. 5.12(a) - (c), respectively.

higher frequencies. Thus, we can improve the performance of APDA solvers by using higher-order APFs to approximate continuous-time delays instead of the third-order

ones used in this dissertation. The APDA method also approximates both time and spatial domain derivatives using finite differences, whereas the CTLD method only discretizes the spatial variable to compute the continuous-time solution; this also influences the relative accuracy of the two classes of solvers. However, note that the APDA method is a direct mapping from an FDTD scheme in which we replace the unit time delays with analog APFs. Thus, the design phase of the APDA method is less complex than the CTLD method, even with different boundary conditions. Furthermore, the APDA method provides a more convenient way to compensate for the propagation delays introduced by practical circuit elements. Thus, it is more suitable for higher-bandwidth implementations for which these delays become more significant. The APDA method is also more general since it can also be used to solve non-linear PDEs, for which deriving a CTLD-based architecture is generally difficult.

CHAPTER 6

CONTINUOUS-TIME ALGORITHMS FOR SOLVING NONLINEAR CONSERVATIVE PDES

Nonlinear PDEs describe the behavior of a wide variety of complex physical phenomena with applications in fluid mechanics, electromagnetics, quantum mechanics, magnetohydrodynamics, etc. In this chapter, a CT mathematical model, which is suitable for designing analog circuits is proposed to solve non-linear coupled PDEs. The proposed model is an extension of the APDA method. Note that the APDA method is derived based on a standard FDTD method [110–115]. Thus, the investigation starts with an existing discrete-time discrete-space numerical method, which is capable of solving coupled non-linear PDEs. Such methods include standard and improved MacCormack's schemes, Lax-Wendroff scheme, and projective Riccati equation method [32, 117, 122, 123]. The proposed CT mathematical model is based on the standard MacCormack's numerical scheme. The discrete-time delay operators in the MacCormack's update equations are replaced with CT delays to obtain SDTC update equations that can be solved using analog computing methods. The CT delay is approximated using an analog all-pass filter (APF) [29–31, 109–111, 116]. Also, analog multipliers are used to realize the nonlinear terms in the PDEs, while other numerical operations in the update equations (e.g., scaling and summing) are realized using op-amps. The proposed architecture solves SDTC update equations simultaneously at each spatial grid point.

Note that this dissertation is only limited to the mathematical models of the non-linear PDE solver. Analog circuit implementations and simulation results are beyond the scope of this dissertation.

Developing a general-purpose analog computation platform that can solve all different types of nonlinear PDEs is an impossible task since even a fully-discrete

numerical method can only solve a specific class of nonlinear PDEs. Thus, the proposed analog computing architecture only considers conservative systems (an important class of physical systems with various real-world applications) that describe the variation of conserved physical quantities such as mass, velocity, energy, and density. Examples of conservative systems include electromagnetics, magnetohydrodynamics, optics, aerodynamics, plasma physics, etc. Furthermore, most of the hyperbolic PDEs, which have a variety of applications such as Maxwell's equations and shock tubes can be expressed in the conservative form. Multidimensional systems of non-linear conservation laws can be expressed as a system of coupled PDEs in the conservative or divergent form (flux-conservative form). In this formulation of physical law, the coefficient of the derivatives are either constant or, if variable, their derivatives do not appear anywhere in the equation [117]. In multidimensions, the conservative form of the non-linear PDEs can be expressed as

$$\frac{\partial \mathbf{\Phi}}{\partial t} + \nabla \cdot \mathbf{f}(\mathbf{\Phi}) = \mathbf{h}(\mathbf{\Phi}), \tag{6.1}$$

where $\mathbf{f}$ and $\mathbf{h}$ are the flux and the source terms of the conserved quantity $\mathbf{\Phi}$, respectively. Here,

$$\mathbf{\Phi} = \begin{bmatrix} u_1\left(x,y,z,t\right) \\ u_2\left(x,y,z,t\right) \\ \vdots \\ u_n\left(x,y,z,t\right) \end{bmatrix}, \qquad \mathbf{f}(\mathbf{\Phi}) = \begin{bmatrix} f_1\left(\mathbf{\Phi}\right) \\ f_2\left(\mathbf{\Phi}\right) \\ \vdots \\ f_n\left(\mathbf{\Phi}\right) \end{bmatrix}, \qquad \mathbf{h}(\mathbf{\Phi}) = \begin{bmatrix} h_1\left(\mathbf{\Phi}\right) \\ h_2\left(\mathbf{\Phi}\right) \\ \vdots \\ h_n\left(\mathbf{\Phi}\right) \end{bmatrix}.$$

In this chapter, the dependent variables of the PDEs are represented using lowercase letters because it is a common practice in the nonlinear computational community.

In particular, the derived mathematical model only considers a one-dimensional (1-D) system (space x and time t) with two conservative variables $u_1(x,t)$ and

$u_2(x,t)$. Thus, $\mathbf{f}$, $\mathbf{h}$, and $\mathbf{\Phi}$ terms can be expressed as

$$\mathbf{\Phi} = \begin{bmatrix} u_1(x,t) \\ u_2(x,t) \end{bmatrix}, \ \mathbf{f}(\mathbf{\Phi}) = \begin{bmatrix} f_1(\mathbf{\Phi}) \\ f_2(\mathbf{\Phi}) \end{bmatrix}, \ \mathbf{h}(\mathbf{\Phi}) = \begin{bmatrix} h_1(\mathbf{\Phi}) \\ h_2(\mathbf{\Phi}) \end{bmatrix}.$$

6.1 Continuous-time Mathematical Model

In order to obtain a SDTC update equation, the system of coupled PDEs given in (6.1) is first discretized into $N_x + 1$ spatial points while keeping the time variable continuous, where $\Delta x = \frac{L_x}{N_x}$ is the spatial step size. Here, the 1-D spatial domain is defined as $0 \leq x \leq L_x$. Then, the variables (solutions of the PDEs) $u_1(i,t) = u_1(i\Delta x, t)$ and $u_2(i,t) = u_2(i\Delta x, t)$ describe the variation of the conservative quantities at $x = i\Delta x$ ($i \in \{0, 1, \ldots, N_x\}$). The spatial indexes $i = 0$ and $i = N_x$ define the left and right boundaries, respectively. The next step is to select a numerical method which is capable of solving nonlinear PDEs defined by the conservative form. The most suitable numerical scheme to solve a particular problem depends on its vortex structure and may also vary from application to application. For example, some may require low order numerical schemes to reduce dissipation of the solution along the spatial dimension. Most of these algorithms are variations of the Lax–Wendroff scheme [124–126].

Here, the MacCormack numerical method, which is a second-order accurate scheme in both space and time is selected [124–126]. This method only consists of two computational steps. Fourth-order accurate schemes, such as the ones proposed in [127] and [128], are more accurate than the selected method. However, these methods dramatically increase the hardware complexity of the analog solvers compared to a two-step method. Moreover, the implicit numerical scheme proposed in [127] is not suited for analog circuit models since it results in tridiagonal matrices.

Thus, it is advantageous to use a two-step explicit scheme that trades off accuracy for less complexity.

The update equations of standard numerical methods are discretized both in space and time. However, the proposed method keeps the time variable t continuous to compute the solution using analog circuits. Let $\mathbf{\Phi}_i^t = \begin{bmatrix} u_1(i,t) \\ u_2(i,t) \end{bmatrix}$, $\mathbf{f_i^t} = \begin{bmatrix} f_1(\mathbf{\Phi}_i^t) \\ f_2(\mathbf{\Phi}_i^t) \end{bmatrix}$, and $\mathbf{h_i^t} = \begin{bmatrix} h_1(\mathbf{\Phi}_i^t) \\ h_2(\mathbf{\Phi}_i^t) \end{bmatrix}$. Here, the nonlinear functions f_1, f_2, h_1, and h_2 are in a particular form. Consider $f_1(u_1, u_2)$ as an example,

$$f_1(u_1, u_2) = \sum_{k_1=0}^{N_1} \sum_{k_2=0}^{N_2} \alpha_{k_1,k_2} u_1^{k_1} u_2^{k_2} \tag{6.2}$$

where α_{k_1,k_2} is the gain associated with the nonlinear term $u_1^{k_1} u_2^{k_2}$. The predictor and corrector steps of the continuous-time scheme can be derived using the MacCormack scheme. The corresponding equations are expressed as [124–126]

$$\begin{aligned} \mathbf{\Phi}_i^p &= \mathbf{\Phi}_i^{t-\tau} - \frac{\tau}{\Delta x}\left(\mathbf{f}_i^{t-\tau} - \mathbf{f}_{i-1}^{t-\tau}\right) + \tau \mathbf{h}_i^{t-\tau}, \\ \mathbf{\Phi}_i^c &= \mathbf{\Phi}_i^{t-\tau} - \frac{\tau}{\Delta x}\left(\mathbf{f}_{i+1}^p - \mathbf{f}_i^p\right) + \tau \mathbf{h}_i^p, \end{aligned} \tag{6.3}$$

where superscripts p and c refer to the predictor and corrector steps, respectively. The superscript $t-\tau$ corresponds to a τ-delayed version of the operator, where τ is a continuous-time delay operator of the numerical scheme. Then the conservative quantity $\mathbf{\Phi}_i^t$ at internal spatial points $i \in \{1, 2, \ldots, N_x - 1\}$ and time t, is computed using the predictor and corrector values as [124–126]

$$\mathbf{\Phi}_i^t = \frac{1}{2}\left[\mathbf{\Phi}_i^p + \mathbf{\Phi}_i^c\right]. \tag{6.4}$$

The SDTC update equations given in (6.3) and (6.4) are then used to design an analog module that computes the solution for internal spatial points (i.e. $i \in \{1, 2, \ldots, N_x - 1\}$). Note that each of the expressions is discrete in space and

continuous in time, such that they can be computed using analog computing methods. For each of the boundaries ($i = 0$ and $i = N_x$), different SDTC update equations need to be developed based on the corresponding boundary conditions and the FDTD scheme, which can then be used to design suitable analog circuits.

6.2 APDA Method for Solving Nonlinear PDEs

Consider the SDTC equations given in (6.3) and (6.4) which eventually solve problems governed by the conservation laws. Each equation involves addition, subtraction and scalar multiplications to compute the update equations. The nonlinear terms inside $\mathbf{f}$ and $\mathbf{h}$ operators (such as $u_1(i,t)^2$ and $u_1(i,t)\,u_2(i,t)$) demand multiplications. Apart from these arithmetic operations, a CT delay operator is required to produce the τ-delayed version of $\mathbf{\Phi}_i^t$. In the APDA method, a CT delay τ, which has a Laplace domain representation $e^{-s\tau}$, is approximated using an analog APF $\phi(s)$. It has been shown that the CT delay $e^{-s\tau}$ can be approximated as $\phi(s) \approx \left[\frac{1-\frac{s\tau}{2m}}{1+\frac{s\tau}{2m}}\right]^m$ with an accuracy that increases with m. Thus, accurate time delays can be realized using a cascade of $m > 1$ APFs ($m = 3$ is sufficient in most cases) [112–116]. It is clear that the proposed analog nonlinear PDE solver requires only analog adders/subtractors, programmable-gain amplifiers (PGAs), APFs, and multipliers as its building blocks, all of which can be realized in integrated circuit form. Also note that the APDA method is a direct mapping from an FDTD scheme in which we replace the unit time delays with analog APFs. All other operations are realized using analog circuits.

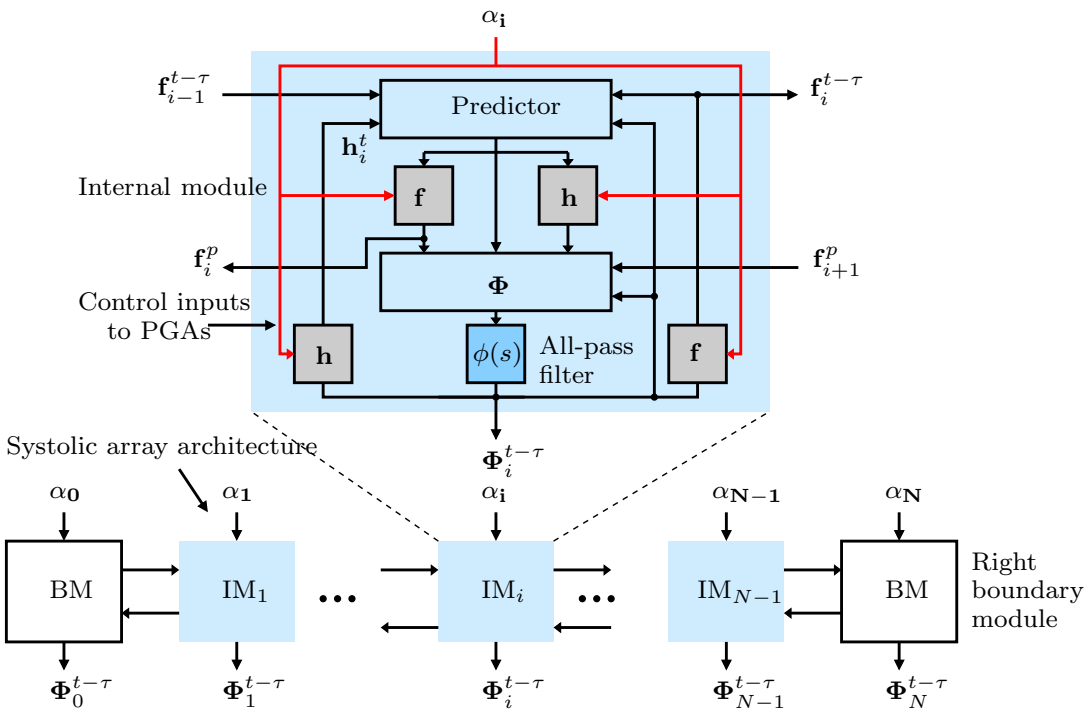

Figure 6.1: The proposed analog computing architecture that solves nonlinear PDEs defined by conservative systems.

6.3 Analog Computing Architecture

The conservative quantity $\mathbf{\Phi}_i^t$ given in (6.4) is computed based on a systematic calculation (using (6.3)), which can directly map into an analog circuit architecture. Fig. 6.1(a) shows a signal flow graph of the proposed analog computing architecture. Note that each signal path represents a vector of values. Consider the internal module IM_i that computes the CT solution of the nonlinear PDE at $x = i\Delta x$. The output of subsystem $\mathbf{\Phi}$, which produces $\mathbf{\Phi}_i^t$, is delayed using an analog APF $\phi(s)$ prior to feeding the $\mathbf{f}$ and $\mathbf{h}$ subsystems. The APF also produces the output of the internal module (IM). Note that the subsystem $\mathbf{\Phi}$ computes the corrector step given in (6.3) (i.e. $\mathbf{\Phi}_i^c$) and (6.4), simultaneously, whereas the predictor subsystem only computes the first equation in (6.3). The input vector $\alpha_\mathbf{i}$ contains all parameters needed to calculate the coefficients of $\mathbf{f}$ and $\mathbf{h}$ at $x = i\Delta x$ (i.e. the α_{k_1,k_2} values

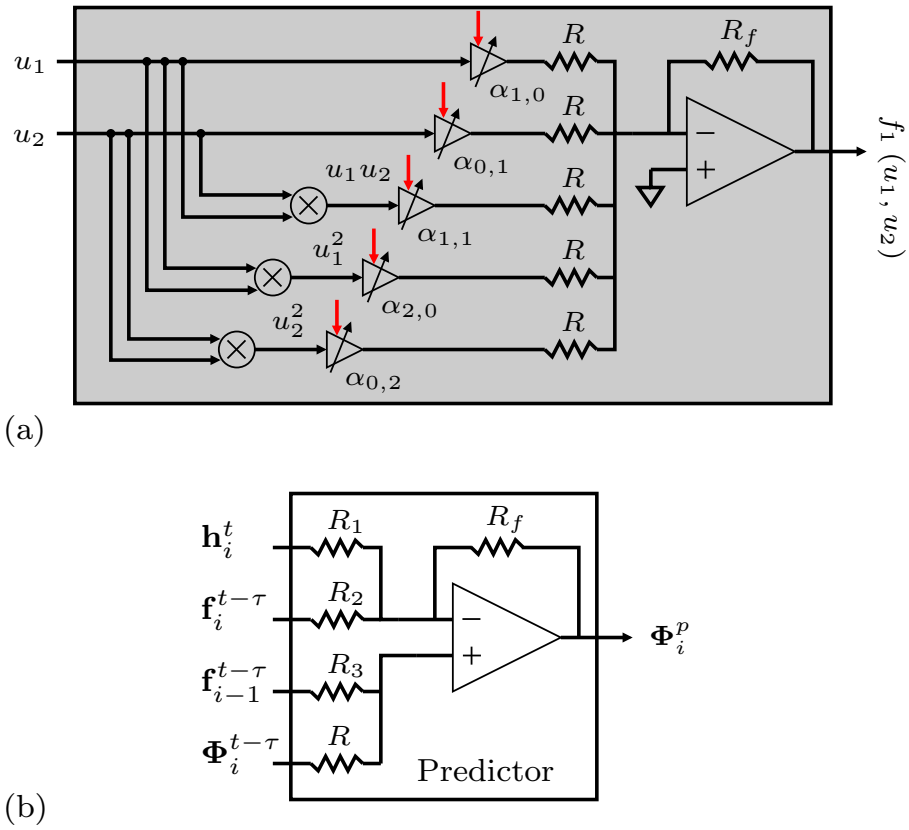

Figure 6.2: Block diagrams of analog circuits for realizing the computations (a) $f_1(u_1, u_2)$ and (b) $\mathbf{\Phi}_i^p$.

in (6.2)). Flux operators $\mathbf{f}_{i-1}^{t-\tau}$ and f_{i+1}^p are fed by the neighboring IMs at $x = (i-1)\Delta x$ and $x = (i+1)\Delta x$, respectively. Similarly, output flux operators $\mathbf{f}_i^{t-\tau}$ and $\mathbf{f}_i^p$ feed the neighboring modules to produce their outputs. Note that this systematic architecture does not vary from problem to problem. Functions $\mathbf{f}(\mathbf{\Phi})$ and $\mathbf{h}(\mathbf{\Phi})$ totally define the problem and introduce non-linearity to the system. In the proposed architecture, we limit the non-linearities in $\mathbf{f}(\mathbf{\Phi})$ and $\mathbf{h}(\mathbf{\Phi})$ to contain only u_1^2, u_2^2, and $u_1 u_2$ terms (non-linearities of order 2). Consider a system with f and h functions, which can be expressed in the following form ($f_1(u_1, u_2)$ is given here)

$$f_1(u_1, u_2) = \alpha_{1,0}\, u_1 + \alpha_{2,0}\, u_1^2 + \alpha_{0,1}\, u_2 + \alpha_{0,2}\, u_2^2 + \alpha_{1,1}\, u_1 u_2. \tag{6.5}$$

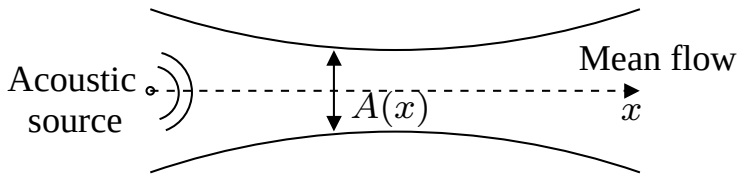

Figure 6.3: Cross section of a variable area duct.

Fig. 6.2(a) shows a block diagram of an analog circuit that realizes the computation of $f_1(u_1, u_2)$. Here, the multiplication operation is realized using analog multipliers. Input vector α_i is used to program the gains of the PGAs. Normalization of a given system limits the α_{k_1,k_2} values to be in the range of $[-1,\ 1]$, which can be implemented using PGAs (which should support negative gains). Thus, the realization of analog circuits to compute $f_1(u_1, u_2)$, $f_2(u_1, u_2)$, $h_1(u_1, u_2)$, and $h_2(u_1, u_2)$ involves multipliers, op-amps and PGAs. Fig. 6.2(b) shows the analog circuit architecture that implements the predictor subsystem, which uses op-amps. Similarly, we can implement a circuit for subsystem Φ. The resulting internal analog modules are finally interconnected in a systolic array architecture to compute the solution over the whole spatial grid (as shown in Fig. 6.1). Here, internal modules calculate the solution at each spatial point simultaneously. For each boundary ($i = 0$ and $i = N_x$) different boundary modules (BMs) need to be implemented based on the governing equations at the boundaries and the corresponding FDTD scheme.

6.4 Example Systems

Two example physical systems that can be solved using the proposed CT computational model are briefly introduced below.

6.4.1 Acoustic Shocks in a Shock Tube

In compressible gas dynamics, shock tube problems are realistic as they portray a converging diverging nozzle modeled after the inlet of a jet engine. Here, we focus our attention to acoustic shocks. Consider a flow of sound waves propagating in a shock tube (variable area duct) with length L as shown in Fig. 6.3, where $A(x)$ represents the geometry of the shock tube. The acoustic waves are generated by an incident plane wave on the left of the duct and leave the right end without reflecting. The total flow field is governed by the following equations (which are in the conservative form):

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial \bar{\rho}\bar{u}}{\partial x} = -\bar{\rho}\bar{u}\frac{A'(x)}{A(x)}, \qquad \frac{\partial \bar{u}}{\partial t} + \frac{\partial\left(\frac{\bar{u}^2}{2} + \frac{\gamma}{\gamma-1}c^2\bar{\rho}^{\gamma-1}\right)}{\partial x} = 0. \tag{6.6}$$

Here, the dependent variables $\bar{\rho}$ and $\bar{u}$ are the total density and velocity, respectively. Corresponding acoustic equations are obtained by splitting the variables into two parts: mean flow variable and acoustic flow variable. In order to partition the flow quantities into mean and acoustic parts, consider the following substitution for the flow variables

$$\bar{u} = u_s + u, \qquad \bar{\rho} = \rho_s + \rho, \tag{6.7}$$

where subscript s denotes the mean values (which are known) and they are assumed to satisfy the steady state condition. We can then rearrange Eq. 6.6 in the conservative form given in Eq. 6.1

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial\left(u_s\rho + \rho_s u + u\rho\right)}{\partial x} &= -\left(u_s\rho + \rho_s u + u\rho\right)\frac{A'(x)}{A(x)}, \\ \frac{\partial u}{\partial t} + \frac{\partial\left[u_s u + \frac{u^2}{2} + c^2\left(\frac{\rho}{\rho_s} + \frac{\gamma-2}{\gamma}\left(\frac{\rho}{\rho_s}\right)^2\right)\right]}{\partial x} &= 0, \end{aligned} \tag{6.8}$$

and the vectors $f(\Phi)$ and $h(\Phi)$ can be expressed as

$$\mathbf{f}(\mathbf{\Phi}) = \begin{bmatrix} u_s\rho + \rho_s u + u\rho \\ u_s u + \frac{u^2}{2} + c^2\left(\frac{\rho}{\rho_s} + \frac{\gamma-2}{\gamma}\left(\frac{\rho}{\rho_s}\right)^2\right) \end{bmatrix} \qquad \mathbf{h}(\mathbf{\Phi}) = \begin{bmatrix} -\left(u_s\rho + \rho_s u + u\rho\right)\frac{A'(x)}{A(x)} \\ 0 \end{bmatrix}. \tag{6.9}$$

It is clear that the system described by Eqs. 6.8 and 6.9 is in the form that is given in Eq. 6.1 and can be solved using the proposed continuous-time model.

6.4.2 1-D Wave Equation

Suppose that the function $w(x,t)$ provides the field intensity of a wave at position x and time t (for the 1-D case). Then, $w(x,t)$ satisfies the following coupled first-order differential equations,

$$\frac{\partial w}{\partial t} - c\frac{\partial v}{\partial x} = 0, \qquad \frac{\partial v}{\partial t} + c\frac{\partial w}{\partial x} = 0. \tag{6.10}$$

Thus $\mathbf{\Phi}, \mathbf{f}(\mathbf{\Phi})$, and $\mathbf{h}(\mathbf{\Phi})$ can be obtained as

$$\mathbf{\Phi} = \begin{bmatrix} w(x,t) \\ v(x,t) \end{bmatrix}, \qquad \mathbf{f}(\mathbf{\Phi}) = \begin{bmatrix} -c\ v(x,t) \\ c\ w(x,t) \end{bmatrix}, \qquad \mathbf{h}(\mathbf{\Phi}) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}. \tag{6.11}$$

The two variable model described in this chapter can be used to solve this system, which can be considered as a new approach for solving the 1-D wave equation.

Boundary Condition

The characteristic-based method can be used to obtain boundary conditions. Since Eq. 6.10 is already in linear PDE form, it can be expressed as [128]

$$\frac{\partial \mathbf{\Phi}}{\partial t} + \underbrace{\begin{bmatrix} 0 & -c \\ c & 0 \end{bmatrix}}_{\mathbf{M}} \frac{\partial \mathbf{\Phi}}{\partial x} = 0. \tag{6.12}$$

Eigenvalues of the matrix $\mathbf{M}$ are $\lambda_1 = c$ and $\lambda_2 = -c$. Corresponding eigenvectors are $e_1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$ and $e_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$. Thus, $S = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$ and $\mathbf{v} = S^{-1}\mathbf{\Phi} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \frac{w-v}{2} \\ \frac{w+v}{2} \end{bmatrix}$. Eq. 6.10 can then be expressed as

$$\frac{\partial}{\partial t}\left(\frac{w+v}{2}\right) + c\frac{\partial}{\partial x}\left(\frac{w+v}{2}\right) = 0, \qquad \frac{\partial}{\partial t}\left(\frac{w-v}{2}\right) - c\frac{\partial}{\partial x}\left(\frac{w-v}{2}\right) = 0. \tag{6.13}$$

CHAPTER 7

LOW FREQUENCY PROTOTYPE ANALOG COMPUTER FOR SOLVING THE WAVE EQUATION

In this chapter, the mathematical models and ideal ACs developed in Chapter 5 are extended to a non-ideal analog circuit implementation. Design challenges that one may face during the implementation phase of the non-ideal analog solver and their analysis would be beneficial when extending these methods towards the analog CMOS realizations. A low-frequency prototype AC that solves the 1-D wave equation has been designed, simulated, and implemented using discrete op-amp ICs. Active circuit realization techniques have been introduced to implement the passive LC section of the internal module shown in Fig. 7.1, mainly because of the difficulties in realizing high-quality factor (high-Q) inductors at low frequencies. The boundary modules are also designed using active circuits. The Bruton transformation is applied to the LC circuits (internal and boundary modules) to avoid the use of inductors [31, 106, 107, 129]. This introduces a circuit element called frequency-dependent negative resistance (FDNR) to the system, which can also be realized using active circuits (op-amps) [31, 106, 107, 129].

The AD8056 op-amp IC, which uses standard voltage-feedback topology and has a 300 MHz of -3 dB bandwidth is utilized to design the circuit. SPICE data of the op-amp circuit is used to simulate the analog solver in the OrCAD PSPICE software. This allows validating the circuit behavior before building the circuit. The performance of the op-amp based analog solver is quantified by comparing the results with a MATLAB FDTD solver. The MSD between the two simulations (MATLAB-based FDTD and OrCAD PSPICE) and the noise to signal energy ratio (γ) are used to quantify the accuracy. The OrCAD models are then extended to a

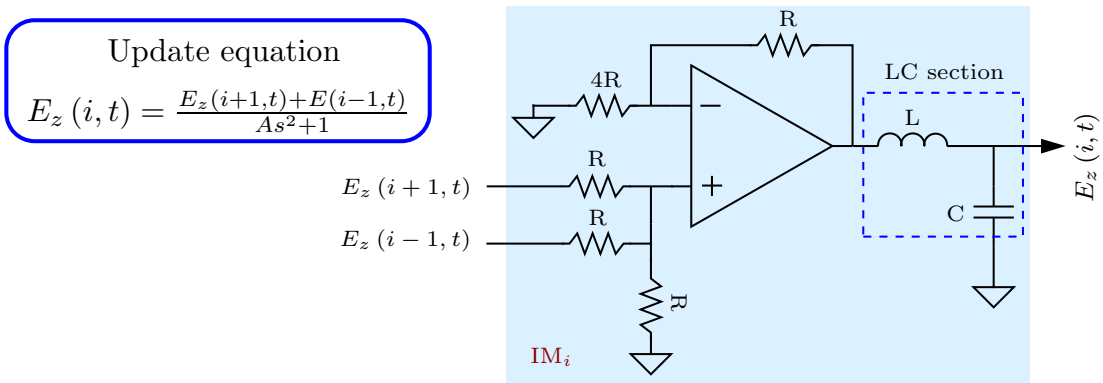

Figure 7.1: An analog circuit that can be used as the internal module (IM) of the 1-D wave equation solver.

printed circuit board (PCB) level implementation as an experimental verification of the continuous-time wave equation solver.

7.1 Active Circuit Implementation

The analog bandwidth of the AC, $\mathrm{F}_{compute}$, was selected as 250 kHz (minimum wavelength $\lambda_{min} = \frac{c}{\mathrm{F}_{compute}}$, where c is the wave speed). The distance between two spatial points Δx is set to be $\frac{\lambda_{min}}{2}$. Consider the update equation of the internal module given in (5.1) (also in Fig. 7.1). The numerator of the update equation is realized using an op-amp as shown in Fig. 7.1. In Chapter 5, the $\frac{1}{As^2+1}$ operation was implemented using an LC circuit (for ideal simulations). However, the realization of a high-Q inductor at low frequencies is a difficult task. Thus, an op-amp based active circuit is employed to realize the passive LC circuit. The selection of $\mathrm{F}_{compute}$ leads to $A = \frac{1}{8\mathrm{F}^2_{compute}} = 2 \times 10^{-12}$.

Consider the passive LC circuit shown in Fig. 7.2 (a). The Bruton transformation, which involves an impedance scaling of the circuit elements, was utilized to transform the LC section into a circuit that does not require any inductors [31, 106–108, 129, 130]. In the Bruton transformation, a circuit element with

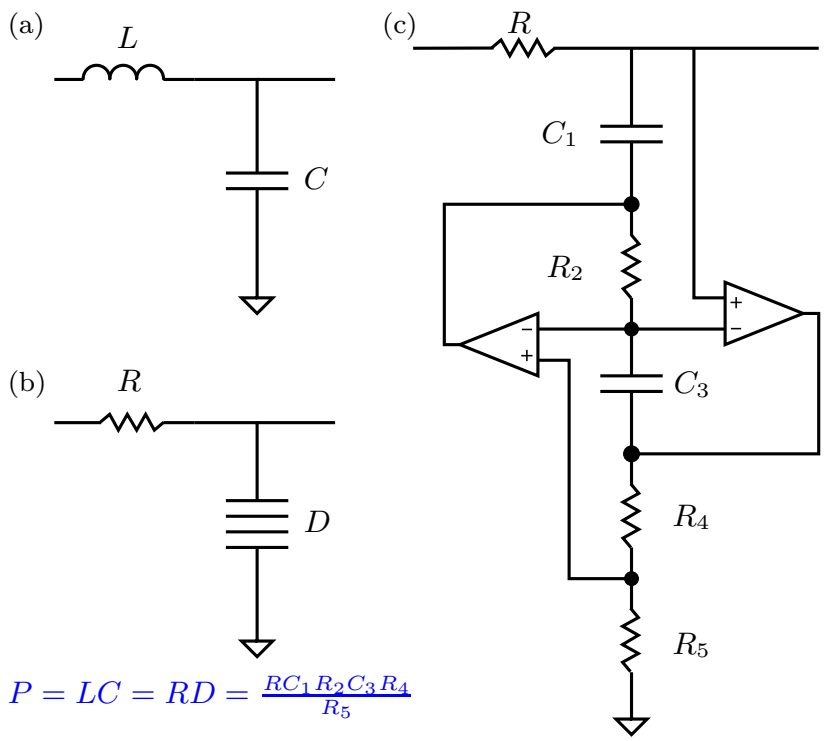

Figure 7.2: (a) Ideal LC passive circuit. (b) Circuit obtained from the Bruton transformation (c) Corrosponding FDNR based implementation.

the impedance $Z(s)$ is transformed into a circuit with an impedance $Z(s)/s$. This implies

- Inductor transforms to a resistor: $Ls \rightarrow L$.
- Resistor transforms to a capacitor: $R \rightarrow \frac{R}{s}$.
- Capacitor transforms to an FDNR: $\frac{1}{Cs} \rightarrow \frac{1}{Ds^2}$.

The impedance transformation does not affect the voltage and current transfers and provides the original transfer function after the transformation. The elimination of the inductors in favor of resistors is the key purpose of the Bruton transformation [31, 106–108, 129, 130]. Thus, the passive LC circuit of the internal module (see Fig. 7.2(a)) can be replaced by a circuit with a resistor and an FDNR element as shown in Fig. 7.2(b), which can be realized using op-amps as shown in Fig. 7.2(c).

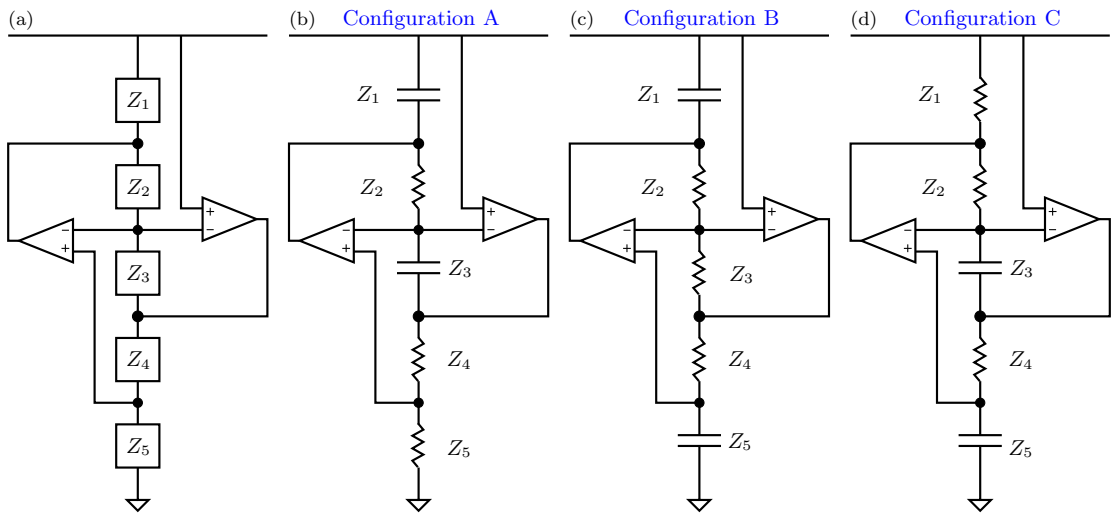

Figure 7.3: (a) Block diagram of a general impedance converter. (b-d) Different configurations (by changing the positions of the two capacitors) for realizing the FDNR elements.

7.1.1 Frequency Dependent Negative Resistance (FDNR)

The FDNR element has a steady-state impedance of $Z(s) = \frac{1}{-D\omega^2}$, which implies a frequency-dependent, negative, real resistance. Here, D is the value of the FDNR element. Such a circuit can be designed using the general impedance converter method and realized using active circuits. Fig. 7.3(a) shows the block diagram of a general impedance converter (GIC). The input impedance of the circuit can then be expressed as

$$Z = \frac{Z_1 Z_3 Z_5}{Z_2 Z_4}. \tag{7.1}$$

By substituting capacitors and resistors at appropriate locations, different types of impedances can be synthesized. Note that the lower end of the circuit must be grounded. An FDNR element can be realized by substituting capacitors to two of the Z_1, Z_3 and Z_5 elements, whereas others are replaced by resistors [106,107]. Thus, depending on the capacitor placement, there are three different configurations to the FDNR element, as shown in Fig. 7.3 (b-d). Each configuration consists of two op-

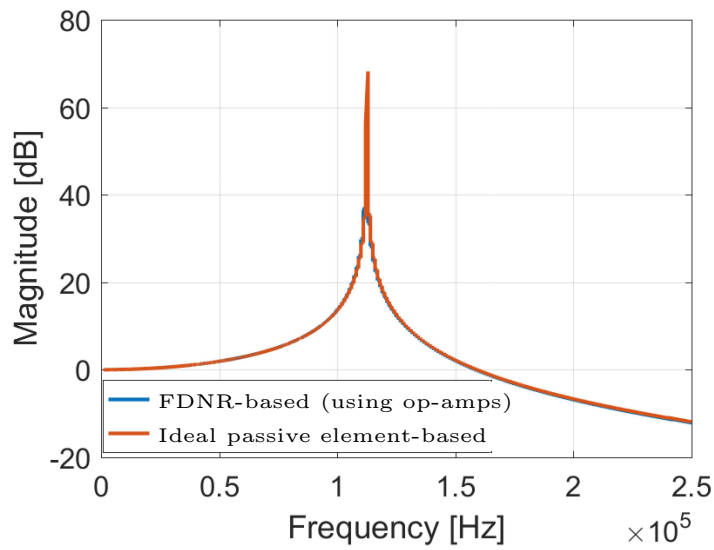

Figure 7.4: Frequency response comparison between the ideal and FDNR-based circuits.

amps, two capacitors, and three resistors. In theory, there is no difference between the three configurations. However, configuration A is the only circuit that provides a return path for the amplifier bias currents. Thus, configuration A is utilized to implement the analog 1-D wave equation solver. Here, the value D of the FDNR element (which has an impedance $Z(s) = \frac{1}{-D\omega^2}$) can be expressed as [106, 107]

$$D = \frac{C_1 R_2 C_3 R_4}{R_5}. \tag{7.2}$$

Fig. 7.4 compares the frequency responses of the non-ideal FDNR element-based circuit (which is based on op-amps) with the passive LC circuit that is implemented using an ideal inductor and a capacitor. AD8056 op-amp is used to implement the FDNR elements. Z_1 and Z_3 capacitor values are selected as 10 nC. Resistor values $Z_2 = 100\ \Omega$, $Z_4 = 100\ \Omega$, and $Z_5 = 50\ \Omega$ are used to obtain the appropriate A value. The FDNR-based circuit provides a better approximation to the ideal LC circuit.

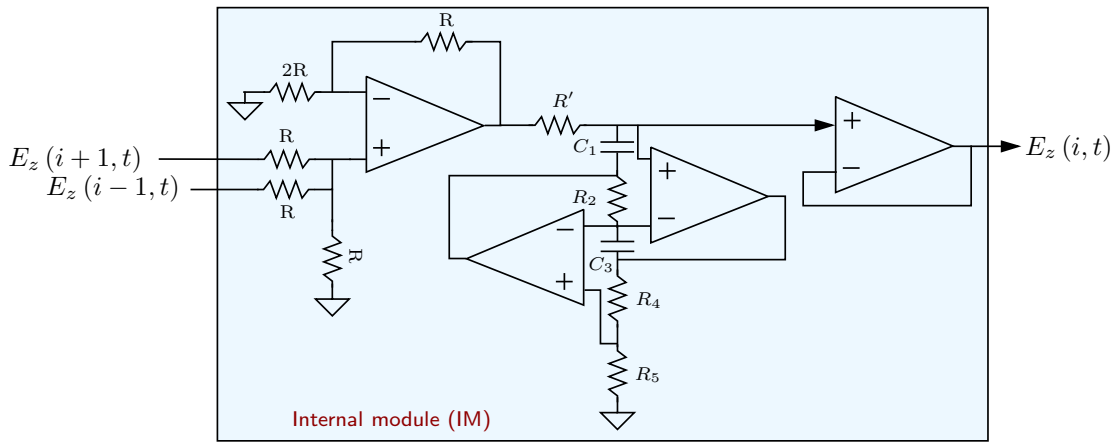

Figure 7.5: FDNR element based implementation of the internal module of the low frequency 1-D wave equation solver.

7.1.2 Analog Computing Modules

Fig. 7.5 shows the complete circuit of the internal module which is realized using the Bruton transform and FDNR elements. The output of the internal module is buffered prior to feeding the adjacent elements. Here, a unity gain buffer is realized using the same op-amp as shown in Fig 7.5. Thus, four op-amps are required to realize the internal module of the low-frequency analog wave equation solver. The following subsection describes the procedure for implementing the radiation boundary module using FDNR elements.

The radiation boundary module (RBM) can be implemented using an LRC circuit as shown in Fig. 7.6 (a). In order to eliminate the use of the inductor, we apply the Bruton transformation to the LRC circuit and the resulting circuit is shown in Fig. 7.6(b) [31,106–108,129,130]. Here, the circuit consists of a capacitor, a resistor, and an FDNR element as shown in Fig. 7.6(c). Fig. 7.6(d) compares the frequency responses of the ideal LRC circuit and the non-ideal FDNR-based RBM circuit, which shows a good approximation.

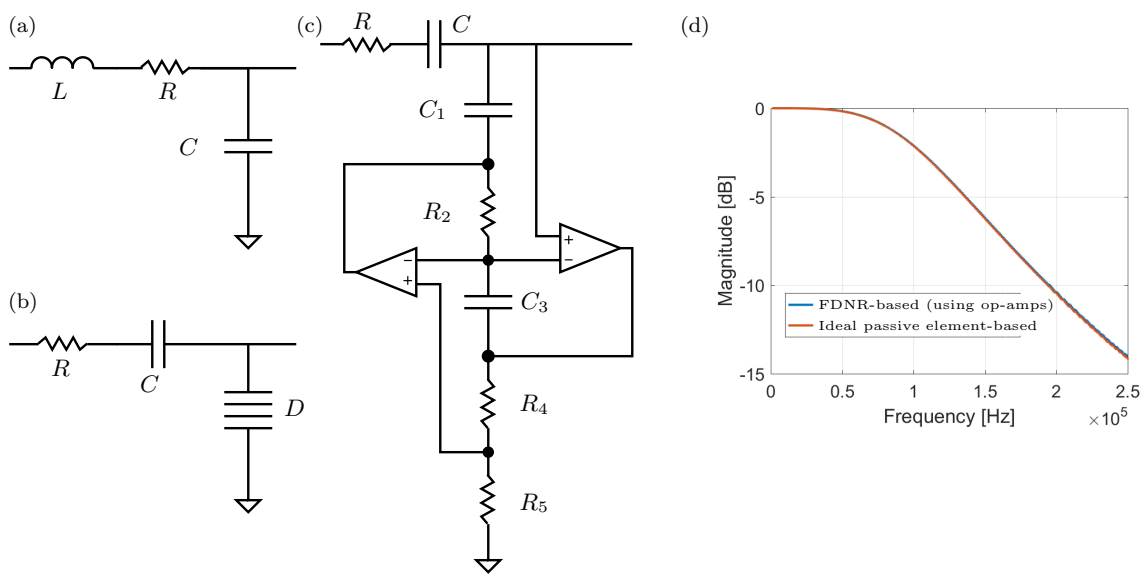

Figure 7.6: (a) Circuit diagram of a reflection free module which is implemented using passive elements. (b) Circuit obtained from the Bruton transformation (c) Corresponding FDNR element-based implementation (d) Frequency response comparison between the ideal passive element- and FDNR-based circuits.

However, this circuit creates a negative dc shift, which increases with the time as shown in Fig. 7.7 (a) (corresponding results are based on a time-domain simulation with a sinusoidal input). This is due to the nonidealities (dc offset) of the op-amps circuits. In order to remove the dc component of the output signal, a first-order RC high pass filter with the cutoff frequency of 1 kHz is utilized (see Fig. 7.7(b)). Fig. 7.7 (c) shows the results from the modified circuit and it is clear that the output of the new circuit follows the ideal output. As similar to the internal module, a unity gain buffer is utilized at the output of the FDNR element to prevent loading.

7.2 SPICE Simulation Results

The internal and radiation modules are then interconnected in a 1-D systolic array architecture to simulate different wave propagation scenarios. The corresponding AC consists of 17 spatial points ($N_x = 16$). The radiation boundary condition is

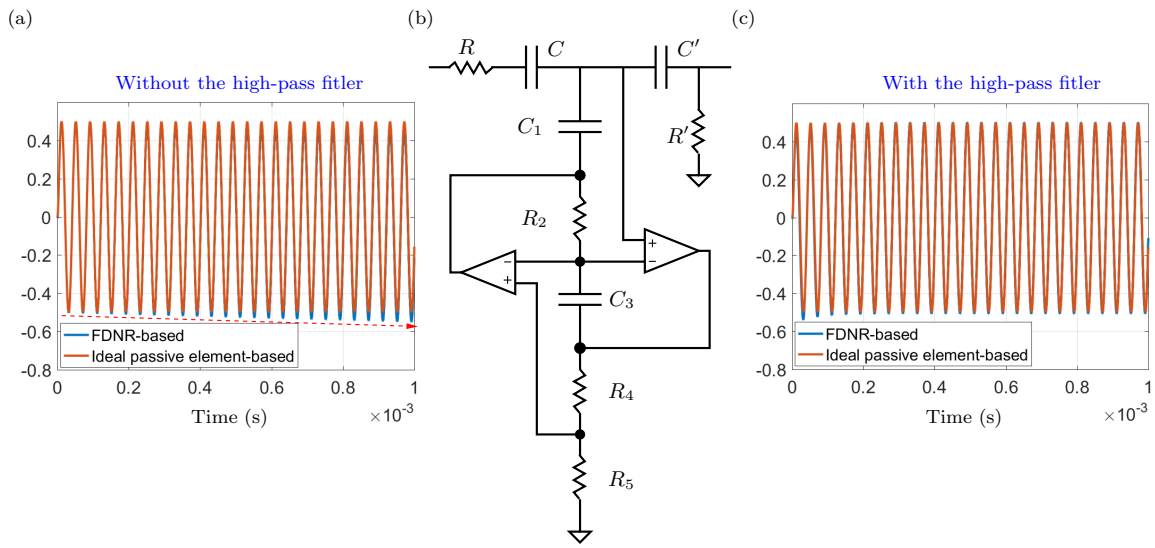

Figure 7.7: (a) Time domain response of the original radiation boundary module. (b) Modified circuit to remove the negative DC shift. (c) Time domain response with the high-pass filter at the output of the radiation boundary module.

considered for simulations. All simulation results are based on non-ideal SPICE models of op-amps (using OrCAD PSPICE software). The following examples show the results obtained from the low-frequency analog wave equation solver for different input signals. MATLAB FDTD simulations are used to compare the results and quantify the accuracy of the computations (using MSD and γ).

Simulation 1: A Gaussian modulated cosine pulse is applied to the left boundary, where $E_z(0,t) = [\cos 2\pi f(t-\tau)]\, e^{-k(t-\tau)^2}$. Here, f is the center frequency of the wide band signal, and is selected as 50 kHz. τ is selected as 50×10^{-5}s. Constant k is used to control the fractional bandwidth of the signal and selected as 10^9 for simulations. The right boundary is simulated using a RBM. Fig. 7.8 (a) shows the space-time domain results obtained from the MATLAB FDTD simulation (both 3-D and top view). The corresponding results obtained from the SPICE simulations (using non-ideal SPICE models) are shown in Fig. 7.8(b).

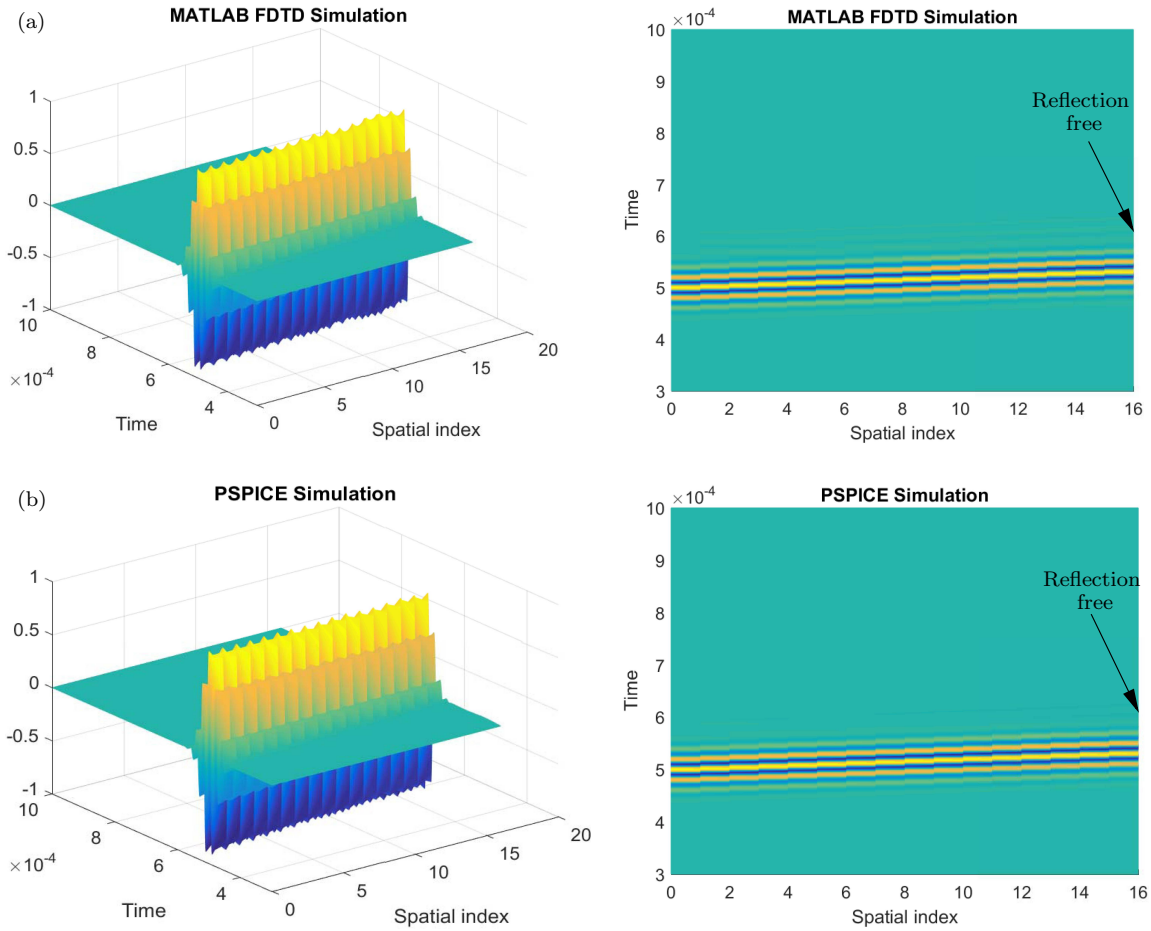

Figure 7.8: Obtained results for a Gaussian pulse with a center frequency 50 kHz (a) MATLAB FDTD simulation (left- 3-D view, right- top view), (b) proposed low-frequency analog wave equation solver (using PSPICE simulation).

Fig. 7.9 summarizes the comparison results associated with simulation 1. The SPICE results are compared with a MATLAB FDTD solver. Fig. 7.9 (a) shows the variation of the absolute difference between the two simulation results in the space-time domain. Figs. 7.9 (b-f) show the MATLAB and PSPICE simulation outputs obtained at different spatial points ($i = 0$, $i = 4$, $i = 8$, $i = 12$, and $i = 16$). The variation of the absolute difference between the MATLAB and the PSPICE simulations is also plotted. The MSD and γ are calculated for each spatial point.

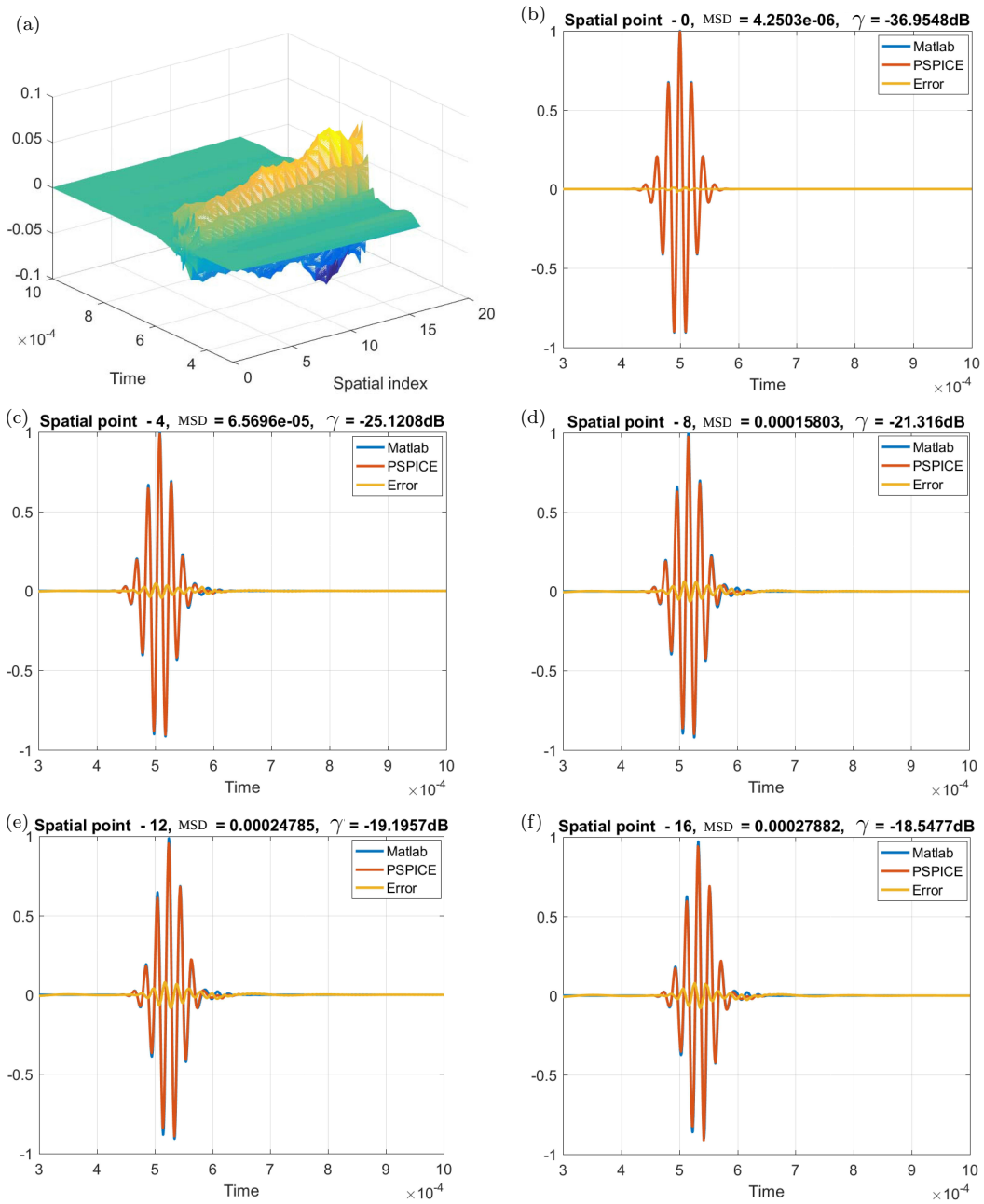

Figure 7.9: (a) Absolute difference between the PSPICE and the MATLAB simulation results shown in Fig. 7.8. (b-f) Signal outputs at different spatial points ($i = 0$, $i = 4$, $i = 8$, $i = 12$, and $i = 16$), along with the absolute difference (error). The mean square difference (MSD) and the noise energy to signal energy ratio (γ) are provided for each spatial location.

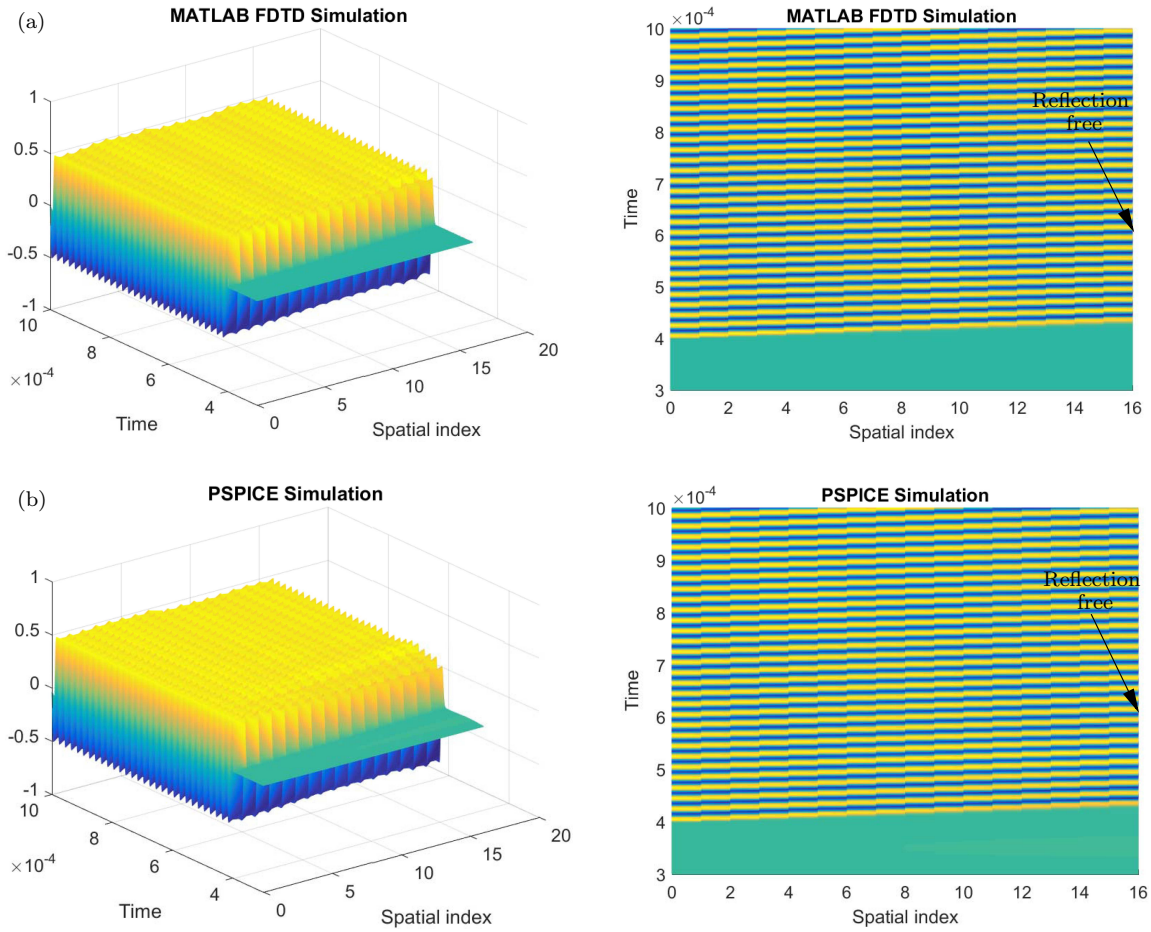

Figure 7.10: Obtained results for a Gaussian pulse with a center frequency 50 kHz (a) MATLAB FDTD simulation (left- 3-D view, right- top view), (b) proposed low-frequency analog wave equation solver (using PSPICE simulation).

Simulation 2: The left boundary is excited using a sinusoidal wave, where $E_z(0,t) = \cos 2\pi f t$. Here, f is selected as 50 kHz. The right boundary is simulated as a radiation boundary condition. Corresponding MATLAB and PSPICE simulation results are shown in Fig. 7.10 (both 3-D and top views). Fig. 7.11 shows the comparison results associated with simulation 2. Signal outputs at different spatial points are plotted along with the absolute difference between the two simulations.

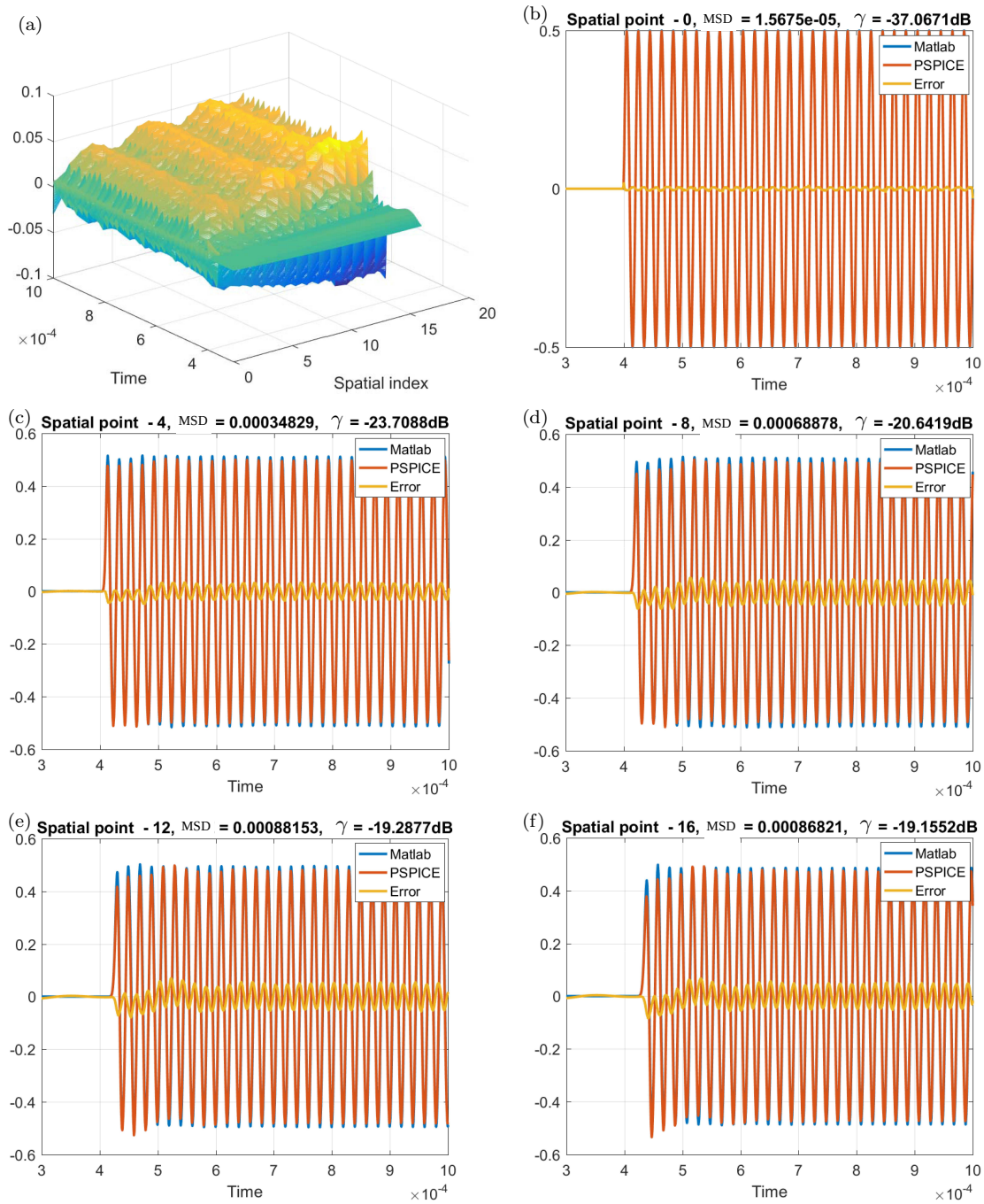

Figure 7.11: (a) Absolute difference between the PSPICE and the MATLAB simulation results shown in Fig. 7.10. (b-f) Signal outputs at different spatial points ($i = 0$, $i = 4$, $i = 8$, $i = 12$, and $i = 16$), along with the absolute difference. The MSD and the γ are provided for each spatial location.

7.3 Printed Circuit Board Level Implementation

The OrCAD models were then designed into a PCB level implementation as an experimental verification of the mathematical models and circuits developed in the previous chapters. Surface mount components had been used to populate the PCB (Rogers 4003C laminates). EAGLE PCB design software was used to layout the design. The circuit board was fabricated using an LPKF plotter model ProtoMat S103 PCB milling machine. Eight- and sixteen-spatial point ACs that solve the 1-D wave equation were implemented and tested. An RS232 connection was used so that multiple 8-spatial point boards could be easily connected to form longer chains. SMA connectors were used to capture signals from each of the spatial point outputs. The radiation boundary module was implemented as a separate circuit, which can be connected to the main PCB board using an RS232 connection.

Figs. 7.12 (a) and (b) show the populated 8-spatial point PCB ((a) top and (b) bottom views) and Figs. 7.12 (c) and (d) show the populated RBM board ((c) top and (d) bottom views). Fig. 7.12 (e) shows a 16-spatial point analog wave equation solver, which is implemented using two 8-spatial point boards. The RBM board is connected to one of the boundaries of the solver.

7.4 Experimental Verification and Measurement Results

Fig. 7.13 shows the experiment setup that is used to capture measurements from the analog wave equation solver. The left boundary of the solver was excited using different analog signals. A signal generator was used to generate input signals and supplied to the circuit through a 50 Ω cable. All the computations are performed in the continuous-time domain by using the analog circuit that inputs and outputs signals via 50 Ω transmission lines. The national instrument data acquisition unit,

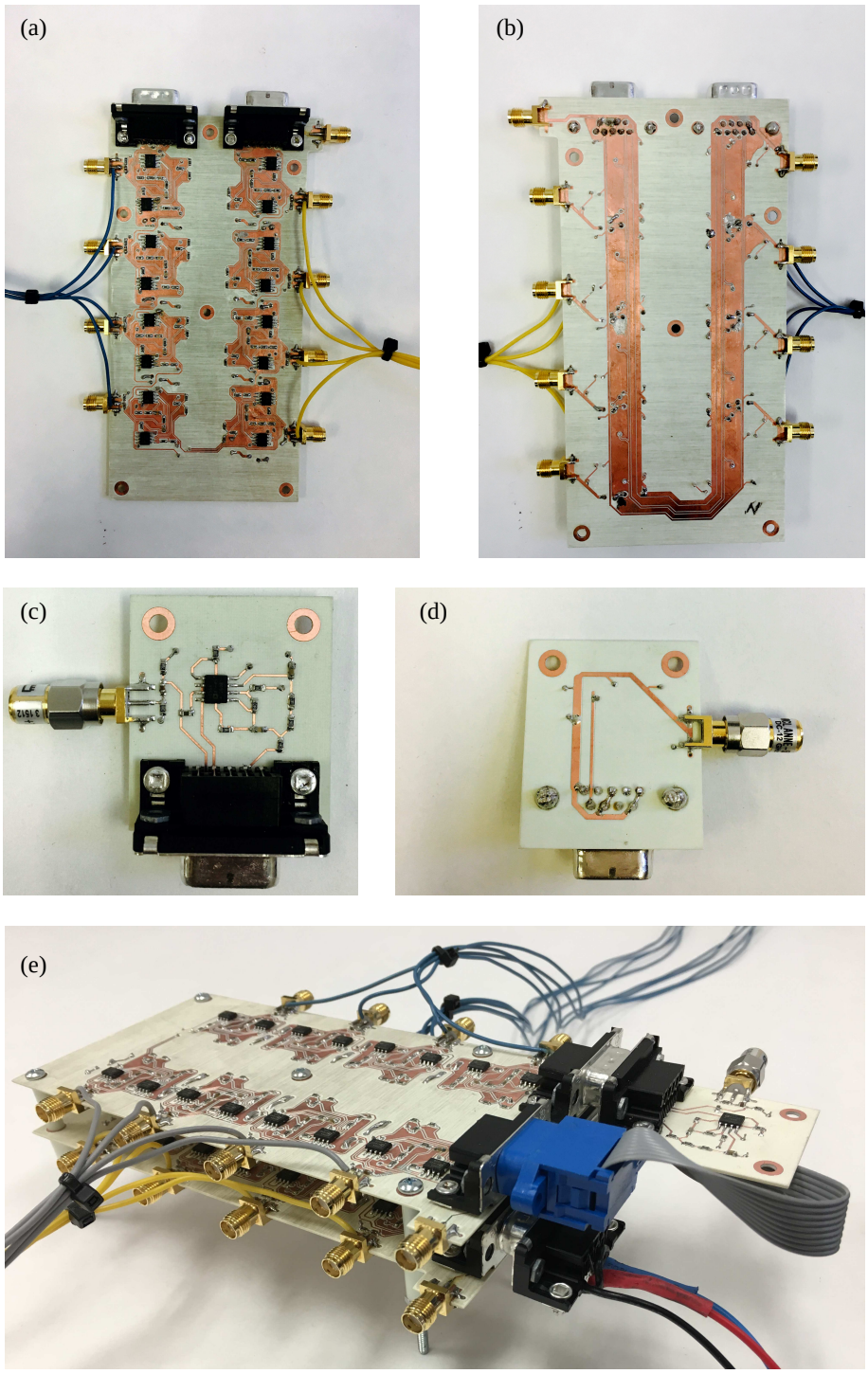

Figure 7.12: (a) Top and (b) bottom views of a populated 8-spatial point board. (c) Top and (d) bottom views of a populated radiation boundary module. (e) An interconnected 16-element analog wave equation solver.

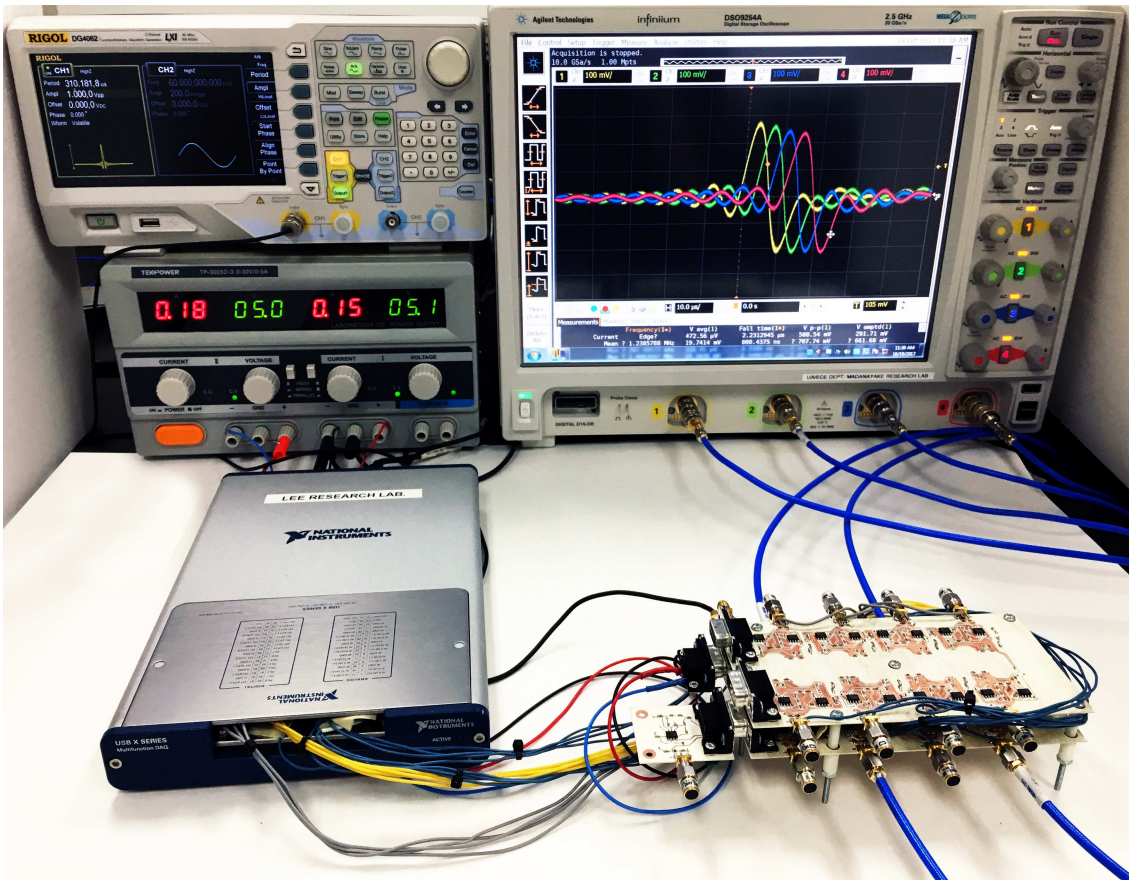

Figure 7.13: Experiment setup to collect data. A signal generator is used to generate the excitation signals. All the computations are performed in the continuous-time domain by using the analog circuit that inputs and outputs signals via 50 Ω transmission lines. A National instrument data acquisition unit is used to capture signals from the circuit.

which consists of several ADCs, was used to capture signals from the circuit (computed outputs of the wave equation solver). In Fig. 7.13, the circuit is excited using a Gaussian pulse at the left boundary. Here, four different outputs are connected to the oscilloscope.

Figs. 7.14 and 7.15 show the measurements obtained from the low-frequency analog wave equation solver. The 16-spatial point version of the circuit is used to collect the measurements. The sampling rate of the NI ADC is selected as 500 kHz. The left boundary is excited using sinusoidal signals with different frequencies. Fig. 7.14

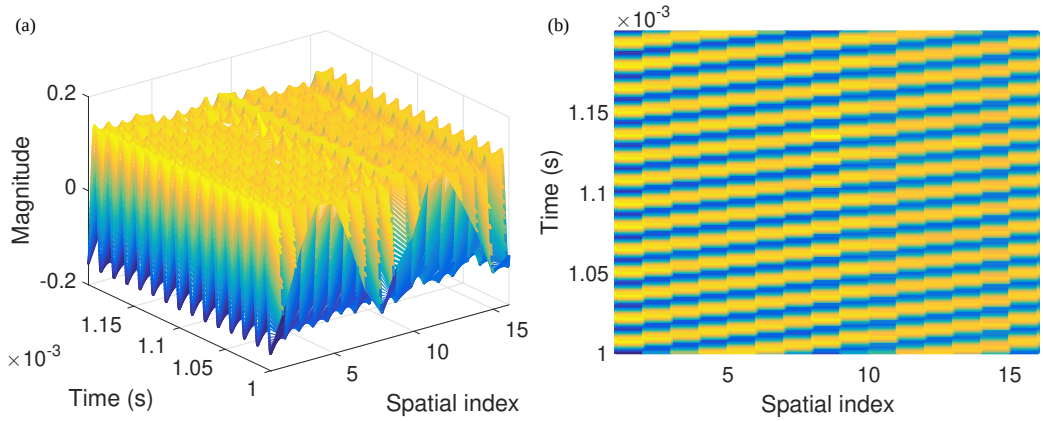

Figure 7.14: The space-time measurement results obtained from the 16-spatial point analog wave equation solver: (a) 3-D view and (b) top view. The left boundary is excited using a sinusoidal signal with a center frequency of 70 kHz.

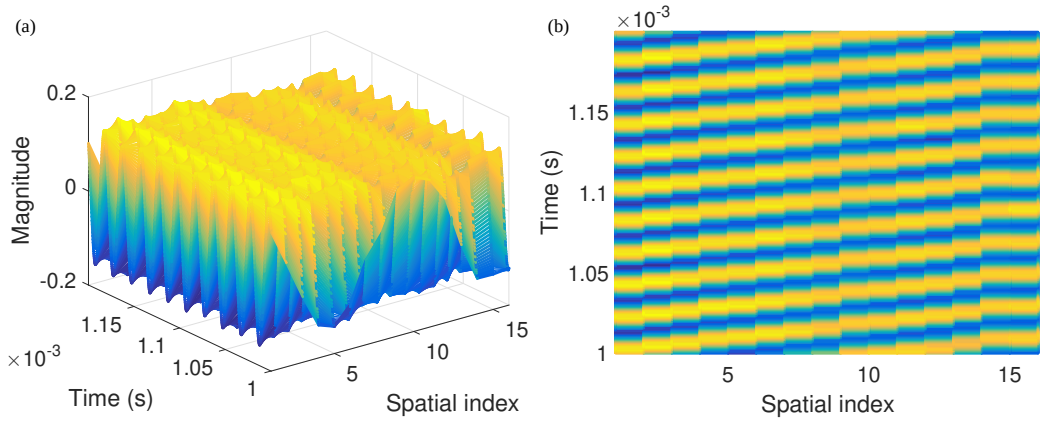

Figure 7.15: The space-time measurement results obtained from the 16-spatial point analog wave equation solver: (a) 3-D view and (b) top view. The left boundary is excited using a sinusoidal signal with a center frequency of 50 kHz.

and 7.15 show the results when the wave equation solver is excited using a sinusoidal input with a center frequency of 70 kHz and 50 kHz, respectively. The results clearly show the wave propagation in the space-time domain (with a constant velocity) and the effect of the radiation boundary (reflection free).

CHAPTER 8

CMOS ANALOG COMPUTER FOR SOLVING THE WAVE EQUATION

The proposed analog computing methods that compute the CT solutions of the 1-D wave equation and Maxwell's equations were extended to analog CMOS designs. This chapter focuses on the analog CMOS implementation of the AC that solves the 1-D wave equation. Mainly, the APDA method was considered. The key challenges toward CMOS implementations of the proposed solvers were identified and discussed with possible solutions. Here, the AC was first simulated with a dominant-pole model, which better approximates the non-ideal (CMOS) circuit behavior. A propagation delay compensation technique, which is required for high-speed designs, was introduced based on the APDA method. The main building blocks of the CMOS wave equation solver were then identified and separately designed, simulated and improved to achieve better performance. Finally, a full-scale AC that solves the wave equation was designed and simulated using CMOS transistor-based circuits and was laid out to realize a chip-level implementation. Also, an on-chip digital calibration circuit was introduced to compensate for process, voltage, and temperature (PVT) variations and random mismatches of the analog CMOS circuits.

The circuit designs employed the Taiwan Semiconductor Manufacturing Company (TSMC) analog CMOS library with the 180 nm process. Low-voltage (operating voltage of 1.8 V) MOS transistor models (substrate-based), which have an operating frequency of 4 GHz (both NFET and PFET), are used to design the circuit. The TSMC library offers fully characterized simulation models with noise, matching, and parasitic parameters and supports up to six metal levels (Levels 3 to 6). A comprehensive set of passive devices including resistors (diffusion resistors, poly resistors, high resistive poly resistors, high precision poly resistors) and capac-

itors (VNCAP, MIM cap, MOS-Varactors) are available in the library to support various applications. Full-scale Berkeley short-channel IGFET model-4 (BSMI4) models that take into account full process parameters were used in the design and simulations. Cadence integrated circuit design software is used to design the proposed analog solvers.

8.1 Key Challenges Towards CMOS Implementations

The key challenge of realizing large-scale analog networks such as the proposed PDE solvers is to minimize the effects of circuit non-idealities such as noise, nonlinearity, component mismatch, and parasitic elements (mostly device and interconnect capacitances) on the computation. Note that the simulation results shown in Chapter 4 and Chapter 5 are based on ideal circuit elements, e.g., op-amps with infinite gain-bandwidth product, infinite input impedance, zero output impedance, and no noise or offset. However, practical op-amps have non-ideal characteristics including finite gain-bandwidth, voltage offsets, input bias currents, noise and drift, finite input and output impedances, finite common-mode and power-supply rejection, and limited linear range. Each of these non-idealities affects the simulated PDE solution with a certain sensitivity that needs to be analyzed prior to a CMOS implementation. These analysis results can then be used to determine the performance requirements of the op-amps and other circuits. As an example, consider the op-amp bandwidth B_o, which is the key factor that limits the usable bandwidth B of the proposed PDE solvers. In general, B_o should be $> 10B$ to obtain acceptable gain and phase characteristics throughout the required bandwidth B, with $B_o > 50B$ preferred for applications that demand high accuracy.

The simulations in Chapter 4 and Chapter 5 have assumed zero propagation delays for each ideal circuit block, but in practice these often cause significant phase

shifts when B exceeds a few MHz. In this case, these delays need to be estimated and compensated to ensure that the solution is stable and has acceptable accuracy. The APDA method is convenient in this respect since it allows the delays from all the circuit blocks to be directly inserted into the continuous-time delay operator τ without affecting the SDTC update equation. Specifically, the actual group delay τ_a of the CMOS APFs can be set to $\tau_a = \tau - \tau_p$, where τ_p is the propagation delay introduced by other circuit elements. It is also possible to compensate the CTLD-based methods in a similar manner, but in this case the RC values within the IMs need to be modified. Note that these RC values are calculated based on the coefficients of the SDTC update equation, where the coefficients are defined by the physical quantities such as μ, ϵ and c. The use of empirical values for these parameters may lead to impractical capacitor and resistor values (e.g., very large capacitor values). Thus, the physical parameters can be scaled down/up to obtain practical RC values that can be realized using analog CMOS. Similarly, if the selection of Δx and c leads to an unrealizable τ_{max} in the APDA method, these parameters can be scaled up or down to obtain a practical value for τ_{max}.

Element-level parasitic components and PVT variations will play a major role in limiting the accuracy of the proposed continuous-time CMOS ACs. Precision circuit design methods must be used to maximize the linear range and minimize systematic mismatches. For example, capacitor ratios can be used to control the gains of the op-amp stages instead of resistor ratios, since the former are more accurate. Expected errors in simulated variables due to component mismatches can be analyzed using Monte Carlo simulations, followed by device sizing and optimized layouts (e.g., common-centroid designs) to obtain acceptable error levels. In general, mismatch can be reduced by increasing the element sizes, but at the cost of increased power consumption [131]. If necessary, the effects of mismatches

can be further reduced by using well-known active techniques like auto-zeroing (for offset cancellation) or dynamic element matching (for mismatch shaping). The effects of parasitic elements can be minimized by estimating them using post-layout RC extraction and back-annotation, and iteratively modifying the design until performance goals are reached.

On-chip digital calibration techniques can also be employed to compensate for PVT variations and random mismatches that degrade the analog circuit performance. For example, binary-weighted capacitor and resistor DACs can be used instead of single circuit elements in the analog solvers to dynamically change R and C values such that matching is improved and PVT variations are minimized. However, the introduction of such reconfigurable hardware will increase the area of the circuit. Scalability of the proposed architectures to larger computational domains and higher dimensions is also challenging for analog IC implementations. However, this can be addressed by taking a highly modular approach to circuit realization, in which a high-resolution problem is partitioned into multiple lower-resolution problems. In this approach, fundamental modules need to be identified and designed in such a way that the higher dimensions and larger computation domains can be achieved by interconnecting multiple modules in analog systolic array architectures.

8.1.1 Dominant-pole Model of the Op-amps

To better portray the behavior under non-ideal circuit conditions, a simulation that approximates the analog circuit elements with more realistic behaviors is performed. Consider the APDA-based AC that solves the 1-D wave equation that was described in Chapter 5. The circuit architecture shown in Fig. 8.1 was used to implement the analog circuits that eventually compute the CT update equations given in (5.14)

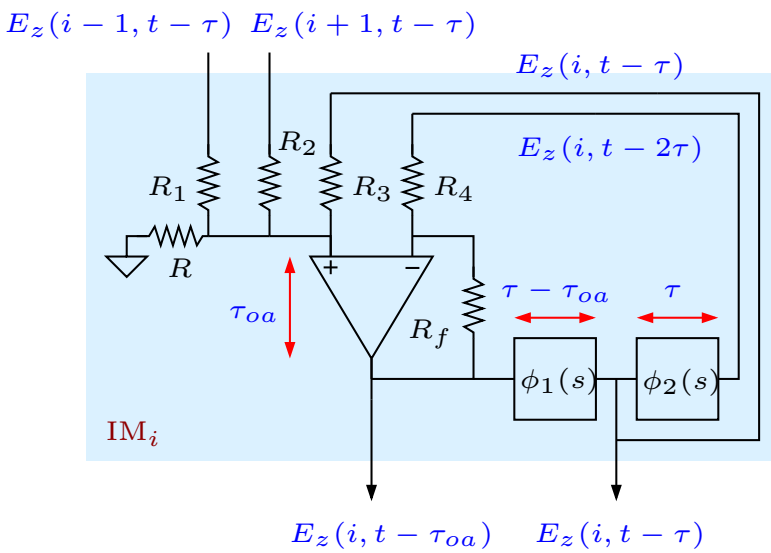

Figure 8.1: The analog circuit architecture used to implement the internal and radiation modules of the APDA-based 1-D wave equation solver (the radiation boundary module only needs one external input). The propagation delay of the op-amp, which is introduced by its dominant pole response, is denoted as τ_{oa}.

and (5.16), which correspond to the internal module and the radiation boundary module, respectively (note that the radiation boundary module only needs one external input). The APFs were simulated using a cascade of three first-order APFs, which provide a better approximation (both in time and frequency domains) to a practical APF.

The op-amp, which is used to perform summing and scaling operations in the update equations was approximated with a dominant-pole response. It has been assumed that the op-amps have a first-order low-pass response to demonstrate the viability of the method for realistic components. Thus, the approximate s-domain transfer function $A(s)$ of an op-amp gain can be expressed as $A(s) = \frac{A}{1+\frac{s}{\omega_0}}$, where A and ω_0 are the DC gain and the dominant pole frequency, respectively. The corresponding unity-gain frequency and gain-bandwidth product (GBW) are approximately $A\omega_0$. By contrast, for an ideal op-amp, the gain is infinite and independent of

frequency. Also it should be noted that the bandwidth of a dominian-pole op-amp in an inverting gain configuration is $\frac{A\omega_0}{K+1}$, where K is the gain of the configuration [107]. The corresponding bandwidth in a non-inverting configuration is $\frac{A\omega_0}{K}$. However, in the ideal case, both inverting and non-inverting circuits have an infinite bandwidth.

In the simulations, the pole frequency was selected as $f_0 = 30$ kHz $\left(f_0 = \frac{\omega_0}{2\pi}\right)$ and the DC gain was selected as 90 dB, resulting in a reasonable GBW ≈ 1 GHz. The selected DC gain and the GBW are comparable with the op-amp that we designed using the 180 nm CMOS technology.

8.1.2 Propagation Delay Compensation Technique

The finite bandwidth of the op-amp introduces a signal propagation delay from its input to output (which is inevitable for all practical elements). This delay affects the continuous-time computation and needs to be compensated by the use of an additional delay in the forward paths. A method that can address this concern has been introduced in this dissertation. In order to explain the proposed propagation delay (PD) compensation technique, consider Fig. 8.1. Let us consider that $E_z(i, t-\tau)$, $E_z(i, t-2\tau)$, $E_z(i-1, t-\tau)$, and $E_z(i+1, t-\tau)$ inputs are available at the input of the op-amp. The op-amp should be able to compute the solution $E_z(i, t)$ at the output instantly (which is the case for an ideal op-amp). However, the actual op-amp requires an amount of time τ_{oa} to produce the solution. Thus, the output is $E_z(i, t-\tau_{oa})$, where τ_{oa} is the propagation delay of the op-amp (assumed to be constant over the expected bandwidth of the solver). Also note that, the delay τ_{oa} can vary with the gain configuration and the gain-bandwidth product of the op-amp. The group delay τ_1 of the all-pass filter $\phi_1(s)$ is selected such that $\tau_1 = \tau - \tau_{oa}$. This selection ensures the $\tau-$delayed version of the electric field at

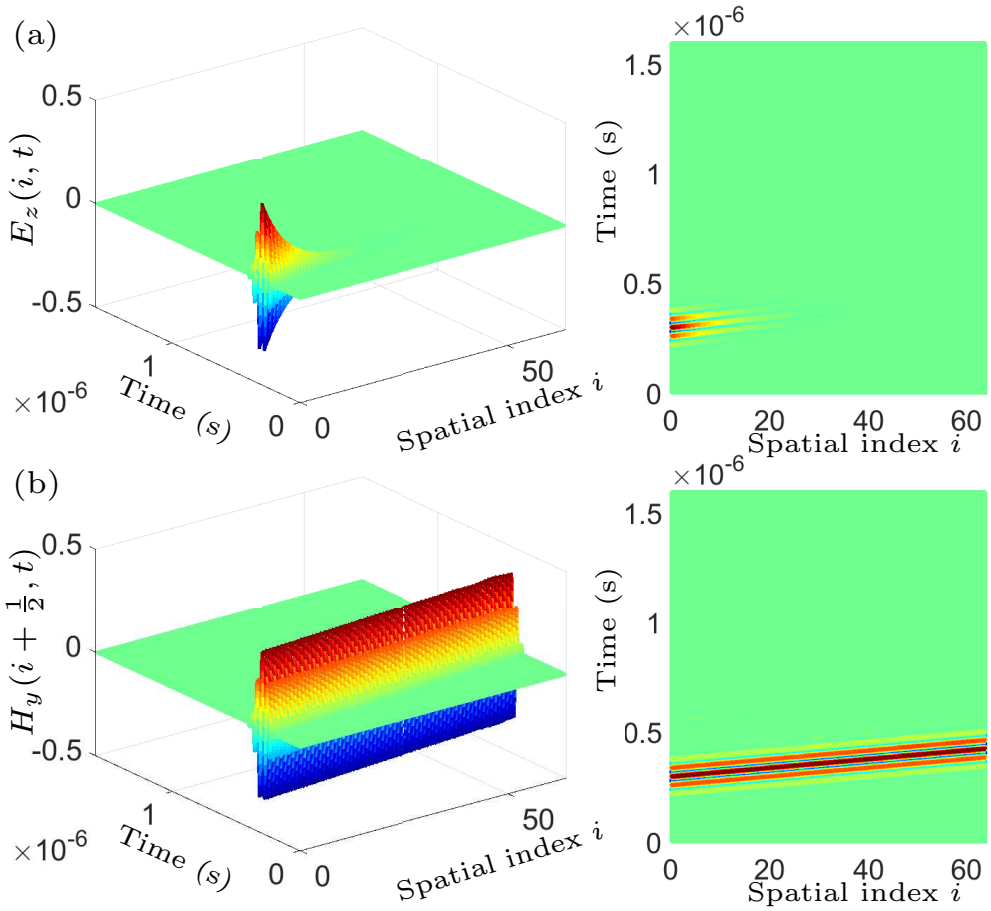

Figure 8.2: Performance of the propagation delay compensation technique. The summing op-amp is approximated with a dominant pole response. The all-pass filters are simulated as a cascade of three first-order all-pass filters. Thus, the simulated circuit closely approximates the realistic behavior of the practical circuit elements. The left boundary is excited using a 25 MHz GMC electric field, whereas the right boundary is simulated as a radiation boundary. Simulation results obtained (a) without and (b) with the propagation delay compensation technique.

the output of the first all-pass filter (which is $E_z(i, t-\tau)$), which can then be used for the accurate computation. Here, the propagation delay of the op-amp is embedded inside the continuous-time delay operator τ. The group delay of the second all-pass filter $\phi_2(s)$, which provides the $2\tau-$delayed version of the electric field, is maintained at the nominal value τ.

This compensation technique, which can be used to compensate for any propagation delays in the practical circuit, is used during the IC implementation to compensate for propagation delays of the op-amps and the buffers. Fig. 8.2(a) shows how the wave equation solver computes the solution when the op-amps are approximated with a dominant pole response in the absence of the propagation delay compensation technique. Here, a radiation boundary is simulated at the right boundary. It demonstrates the effectiveness of the propagation delay on continuous-time computation and the solution. Even though the wave propagation can be seen in the space-time domain plot, the amplitude of the propagating wave rapidly decays with the spatial index, which should not be the case for behavior of a loss-less propagation. Fig. 8.2(b) shows the results after applying the propagation delay compensation technique as discussed here.

Note that gain compensation may be required even after the application of the PDC technique since the gains of the circuits are also frequency-dependent and drop with the frequency increases. This can be achieved by employing variable gains for the op-amp circuits. Also, the all-pass filter gains can be made programmable to calibrate the gains.

8.1.3 Non-ideal Simulation Results

Figs. 8.3(a-c) show the space-time simulation results obtained for three different wave propagation scenarios after the PD compensation technique is applied. For all three simulations, the left boundary is excited using a 25 MHz GMC electric field. The right boundary is simulated as either a fixed, soft, or radiation boundary, respectively. The propagation delays from the summing op-amps used in the internal module and the radiation boundary module are measured as 286 ps and

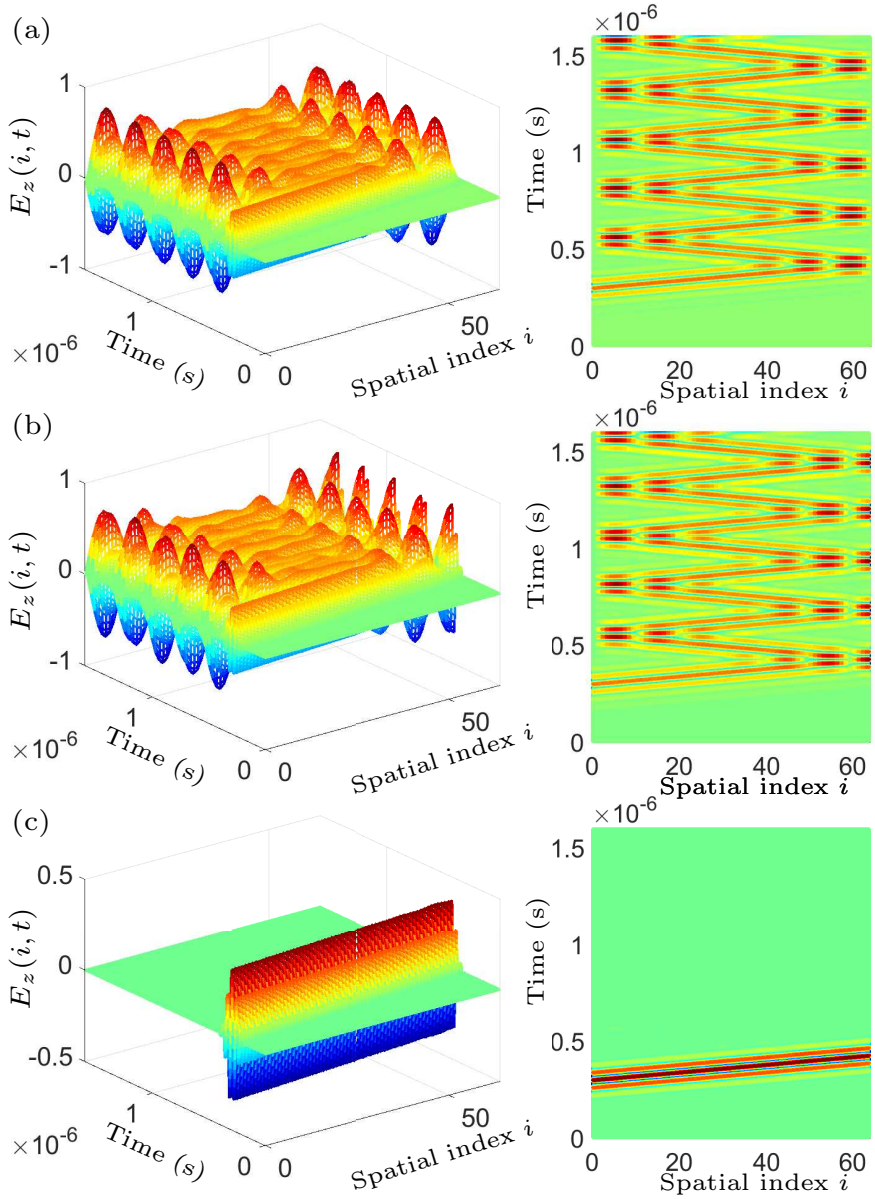

Figure 8.3: Simulation results obtained from the APDA method-based 1-D analog wave equation solver (left: 3-D view, right: top view). The summing op-amp is approximated with a dominant pole response. The all-pass filters are simulated as a cascade of three first-order all-pass filters. The left boundary is excited using a GMC field at 25 MHz. The right boundary is realized as (a) a Dirichlet boundary, (b) a Neumann boundary, and (c) a radiation boundary.

175 ps, respectively. Even though both modules use the same op-amp (with same gain-bandwidth product), they experience different delays due to the different gain

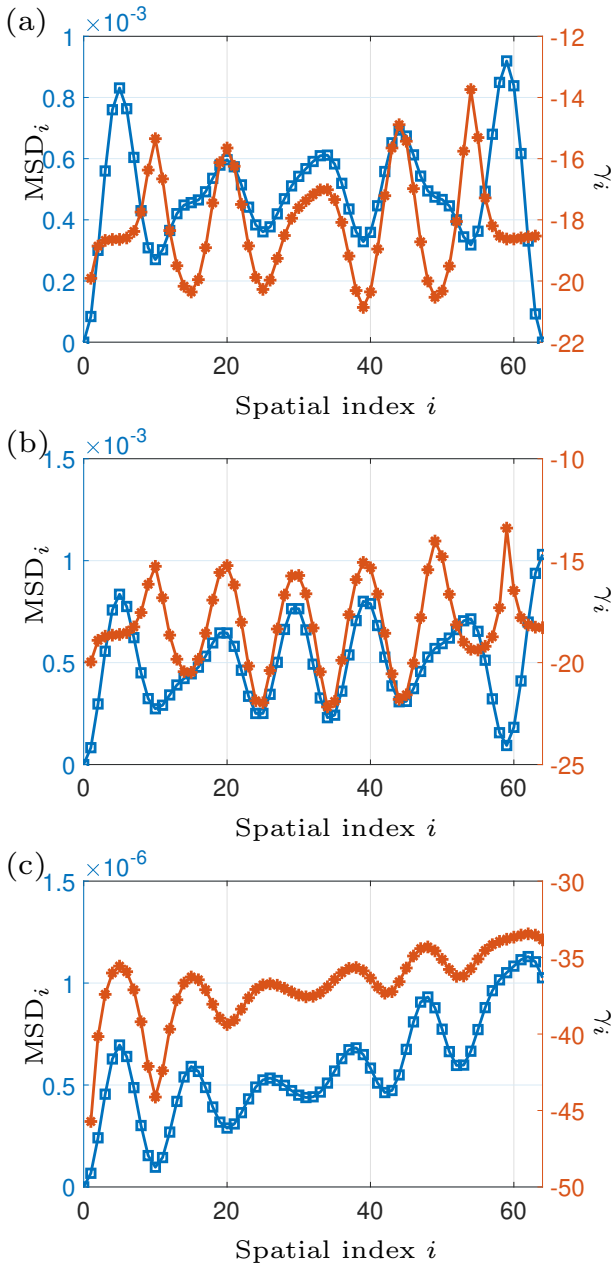

Figure 8.4: (a-c) Comparison results (MSD_i and γ_i) corresponding to the three simulation scenarios shown in Fig. 8.3.

configurations (which are defined by the update equations). The analog bandwidth of the AC $\text{F}_{compute}$ was selected as 50 MHz, and the spatial step size $\Delta x = \frac{\lambda_{min}}{10}$. Here, τ is selected as 1273 ps, which satisfies the stability condition $\tau \leq \frac{\Delta x}{c}$. The

group delays of the all-pass filters $\phi_1(s)$ are selected as 987 ps and 1.098 ns for the internal and the radiation modules, respectively. Thus, in the real circuit, we require a variable all-pass filter as $\phi_1(s)$ that can finely tune its group delay around a nominal value τ. Furthermore, the gains of the op-amp need to be calibrated for the accurate computation of the continuous-time update equations (due to PVT variations). The results are compared with a FDTD simulation and Figs. 8.4(a-b) show the corresponding results (MSD and noise to signal energy ratio). Here, the time domain step size ΔT of the FDTD simulation is equal to τ. It is clear that the proposed analog computing methods along with compensation techniques can be used to compute the solution of the wave equation, even with non-ideal circuits, with an error (in terms of noise to signal energy ratio) that is less than -13 dB.

8.2 180 nm CMOS AC that Solves the Wave Equation

The APDA-based AC that solves the 1-D wave equation has been extended towards a CMOS chip-level implementation. The challenges described in the previous section have been addressed and discussed in the following subsections. The circuit designs employed the TSMC analog CMOS library with the 180 nm process.

8.2.1 Top-level Design Parameters

The top-level block diagram of the AC is shown in Fig. 8.5. The electric field E_z is defined at discrete grid points $x = i\Delta x$ ($i \in \{0, 1, 2, \ldots, N\}$), where Δx is the spatial step size. Spatial indexes $i = 0$ and $i = N$ correspond to the left and right boundaries, respectively. The AC produces time-varying voltages that correspond to the CT electric fields at each spatial point, which are governed by the wave equation (and the boundary conditions). It consists of 18 analog modules (M1 to

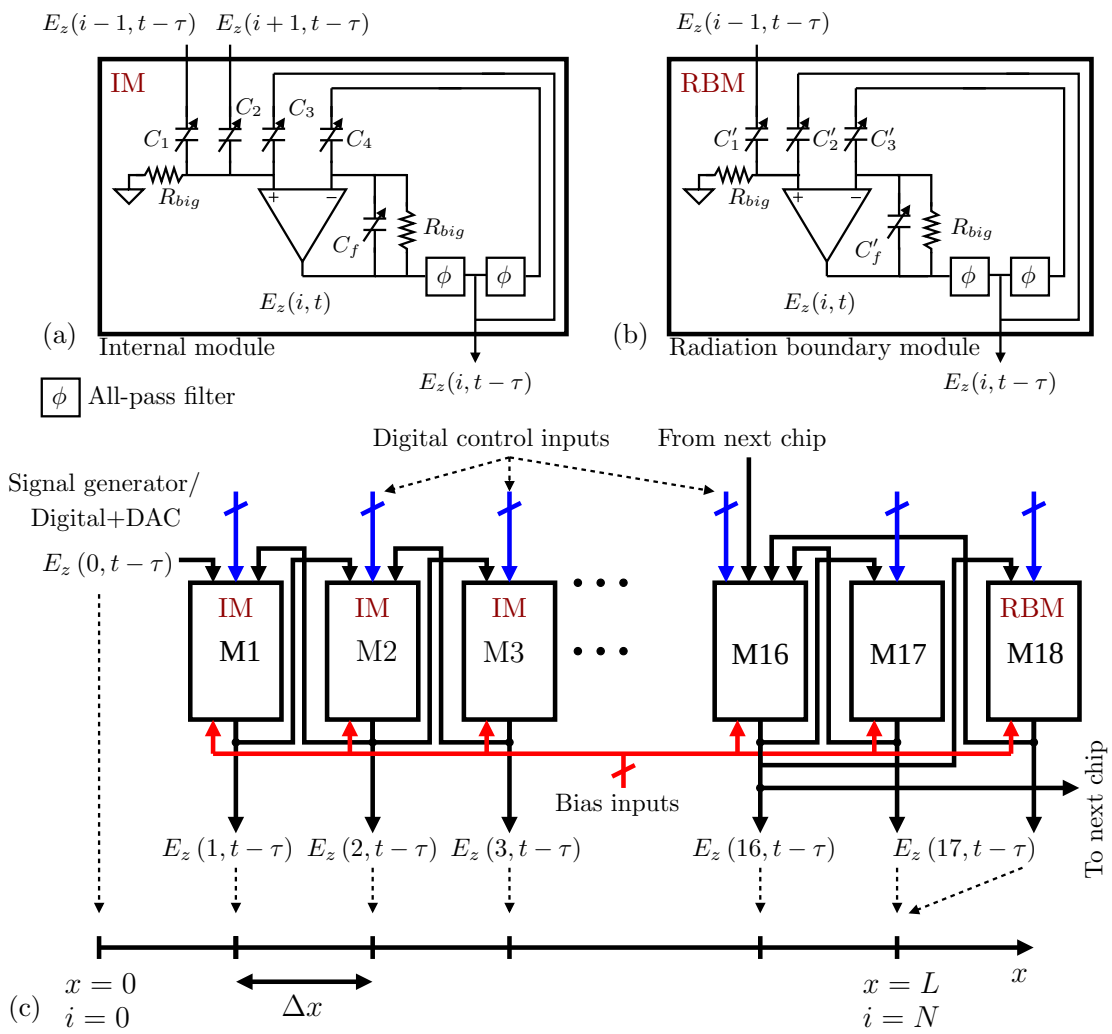

Figure 8.5: The top-level block diagram of the analog CMOS wave equation solver.

M18), where each module corresponds to a particular spatial point. The first 16 modules (M1 to M16) are designed as internal modules (IMs) (see the schematic of an IM shown in Fig. 8.5). These modules correspond to spatial indexes $i = 1$ to $i = 16$. The input to M1 is considered as the solution at spatial index $i = 0$. This is also the excitation signal at the left boundary, which can be supplied to the circuit using a signal generator or a digital circuit plus DAC.

The proposed AC is capable of implementing three different boundary conditions at the right boundary. Module 17 is used to realize the Dirichlet and Neumann boundary conditions. Note that these two boundaries can be implemented using an IM. Thus, the architecture of M17 is similar to the IM, but with additional control circuitry to configure the required boundary. Module 18 is used to realize the radiation boundary condition. The schematic of the radiation boundary module is shown in Fig. 8.5. Module 16 connects the boundary modules (M17 and M18) with the main circuit. Here, a control input at M16 determines the right-side boundary condition. The selection of a particular boundary disconnects other boundaries from the computing circuit. The configuration of either radiation or Neumann boundary condition simulates a spatial grid of 18 spatial points ($N = 17$), whereas for the Dirichlet boundary, $N = 18$ (19 spatial points). In the Dirichlet condition, the boundary is set by grounding the $E_z\left(i+1, t-\tau\right)$ input of the IM. Thus, $E_z\left(i+1, t-\tau\right)$ input of M17 is considered as the solution at $i = 18$. In addition to the three boundary conditions, M16 can be excited using an external input signal. Thus, the AC can be extended along the spatial dimension to simulate larger spatial grid sizes by cascading several chips. This can be achieved by connecting module M16 on a given chip to M1 on the next chip. Here, the boundary modules of the first chip are disconnected from the computing circuit.

The expected analog bandwidth (or $F_{compute}$) for the CT computation of the AC is set to 50 MHz at design time. This results a minimum wavelength of $\lambda_{min} = \frac{c}{\mathrm{F}_{compute}}$. The spatial step size is set to $\Delta x = \frac{\lambda_{min}}{8}$, where the spatial oversampling factor is 4. The equivalent temporal over-sampling factor for FDTD in a DC is selected as 6.25 to satisfy the stability condition $\tau \leq \tau_{max} = \frac{\Delta x}{c} = 2.5$ ns which leads to the CT delays of $\tau = 1.6$ ns in the AC. Thus, the equivalent temporal update rate in the AC is 625 MHz. Here, c is the wave speed. The proposed analog

solver is able to simulate different wave propagation speeds. Note that the gains of the op-amp depend on parameter $K = \frac{c\tau}{\Delta x}$. Thus, the corresponding gain values are configured using a digital control input to vary the propagation speed of the medium (each module can be programmed individually).

8.2.2 Primary Building Blocks

To start with, the primary building blocks of the analog solver have been identified and designed using Cadence circuit design software. The op-amp, which was used for scaling and summing operations is designed, simulated and improved to achieve the required performance. It was also used as a buffer in the circuit (unity gain configuration). As described in Section 8.1, the gain-bandwidth product of the op-amp should be > 500 MHz to obtain an acceptable accuracy from the analog solver (that has a bandwidth of 50 MHz). In general, the phase margin of the op-amp should be $> 60°$ to achieve stability. Also, in order to maintain an acceptable accuracy, the open-loop voltage gain A_{OL} of an op-amp should be high; this can be verified by using the basic feedback theory. The expression for a closed-loop gain A_{CL} with a forward gain of α and a feedback gain of β is $A_{CL} = \frac{\alpha}{1+\alpha\beta}$. In an op-amp $\alpha = A_{OL}$ and the loop-gain $\alpha\beta = A_{OL}\beta$. The term $\frac{1}{\beta}$ is the ideal closed-loop gain of the op-amp in a non-inverting configuration. By rearranging the terms,

$$A_{CL} = \frac{1}{\beta}\left(\frac{1}{1+\frac{1}{A_{OL}\beta}}\right). \tag{8.1}$$

Thus, the percentage of error can be calculated using

$$\text{Error\%} = \left[1 - \left(\frac{1}{1+\frac{1}{A_{OL}\beta}}\right)\right] \times 100 \approx \frac{1}{A_{OL}\beta} \times 100. \tag{8.2}$$

It is clear that the error of the gain increases as the loop-gain decreases in frequency. This is the reason for selecting the gain bandwidth product of the op-amp $B_o > 10B$.

The error term can also be made smaller by having an op-amp with a larger open-loop gain A_{OL}. The expected open-loop DC gain is about 90 dB, which results ∼0.003% of error for the unity gain configuration at DC.

In the proposed system, capacitors were used instead of resistors to control the gains of the summing op-amp to improve the accuracy of the analog solver. This also minimizes the effect of the PVT variations on the continuous-time computation. The capacitor ratio then determines the gain of the op-amp. In general, the ratio of the capacitors can be realized more accurately than the ratio of the resistors in a silicon wafer. Also, much better matching can be achieved by making the capacitor area large (errors less than 0.2% can be achieved) [131]. In particular, four-bit capacitor DACs were used as capacitors, instead of a single capacitor. Thus, the capacitor DAC can produce 16 different capacitor values, which correspond to 16 gain configurations. The required capacitor value is selected using a digital control signal. The use of capacitor DACs improves the reconfigurability of the proposed wave equation solver. Furthermore, the DACs can be used during the calibration stage to compensate for the solver for PVT variations and device parasitics in the chip. However, the introduction of capacitors to the circuit blocks the DC paths to the op-amps. Thus, additional circuitry is required to maintain the required DC voltage levels.

An all-pass filter was designed based on a resistor-capacitor network. The all-pass filters are used to approximate the $e^{-s\tau}$ terms appears in (5.14) and (5.16) as $e^{-s\tau} \approx \left[\frac{1-\frac{s\tau}{2M}}{1+\frac{s\tau}{2M}}\right]^M$ and are realized in an analog RC topology using a cascade of M first-order all-pass filters. Voltage buffers are used in between each cascading stage to avoid loading effects. The bandwidth of the all-pass filter should be greater than 100 MHz to achieve a reasonable accuracy from the AC. A constant group delay of ∼1-2 ns is expected over the bandwidth of the solver. Also, the all-pass filters

are tunable in terms of the gain and the group delay. The tunability of the all-pass filters is used to compensate for the PVT variations of the analog solver. In addition, the variable group-delay is necessary to compensate for the propagation delays, as discussed in the preceding section. Each module consists of several capacitors and resistor DACs that need to be configured using a digital control signal. An on-chip digital serial peripheral interface (SPI) module is designed to communicate and program the chip. Note that each module can be programmed individually. The primary analog circuits are designed and simulated using Cadence. The following subsections describe the circuits in greater detail.

8.2.3 High-speed CMOS Op-amp

A two stage op-amp is designed and ported to TSMC 180 nm CMOS technology. Fig. 8.6 shows the schematic of the op-amp [132–134]. The first stage is a telescopic differential amplifier (diff-amp). The current source loads and the differential pairs of the diff-amp are cascoded to increase the gain from the first stage. The gain of the diff-amp A_d can be expressed as

$$A_d \approx g_{m1} \left(g_{m2} r_{o2}^2 || g_{m4} r_{o4}^2 \right), \tag{8.3}$$

where g_{mX} and r_{oX} are the transconductance and the output resistance of transistor MX (or MXT or MXB), respectively. Transistors M6, M5T, and M5B are used to bias the first stage. The second stage is the PMOS common source stage with a cascoded current source load. An indirect feedback technique is used for compensating the two-stage op-amp [132, 133]. This method results in a faster op-amp circuit and a smaller layout area. Also, the proposed circuit offers a better power supply rejection ratio (PSRR) at higher frequencies. The compensation capacitor C_c does not directly connect to the output of the diff-amp as most of the two-stage op-amp

circuits in the literature. Connecting a compensation capacitor between two high-impedance nodes introduces a right-hand plane (RHP) zero, which limits the speed of the op-amp. In the indirect compensation, the current is feedbacked indirectly to the output of the diff-amp as shown in Fig. 8.6. Here, the capacitor is connected to an internal low impedance node at X (source of M4B). This technique introduces a left-hand-plane (LHP) zero. The LHP zero is located at $-\frac{g_{m4B}}{C_c+C_A}$ and improves the phase margin at the unity gain frequency f_{un}, where C_A is the capacitance at low impedance node X [132, 133]. The unity gain frequency is located at around $\frac{g_{m1}}{2\pi C_c}$ [132, 133]. The low frequency open loop gain of the two stage op-amp is

$$A \approx g_{m1} g_{m8} \left(g_{m2} r_{o2}^2 || g_{m4} r_{o4}^2 \right) r_{o8}. \tag{8.4}$$

The channel lengths of all MOSFETs are selected as 360 nm, except for transistor M8 (second gain stage). The channel length of M8 is 180 nm. A smaller channel length is selected for M8 to move the second pole sufficiently away from the origin of the complex frequency plane (beyond the unity-gain frequency). Note that a large capacitor is required in the direct (or Miller) compensation technique to improve the phase margin. Thus, the compensation capacitor generally dominates the layout area of an op-amp. However, the compensation capacitor C_c of the proposed circuit is only about 160 fF. The use of the indirect compensation technique relaxes the capacitor requirement, which leads to a significantly smaller chip area.

Four bias voltages (Vb1, Vb2, Vb3, Vb4) are used to bias the op-amp. Their simulated values are Vb1 = 1100 mV, Vb2 = 575 mV, Vb3 = 1100 mV, and Vb4 = 715 mV. Slightly higher W/L ratios are selected for M8, M9T, and M9B to obtain a higher gain from the second amplification stage and to improve the phase margin (to move the second pole away from the unity gain frequency). The common centroid technique and other precision layout techniques are utilized to layout the

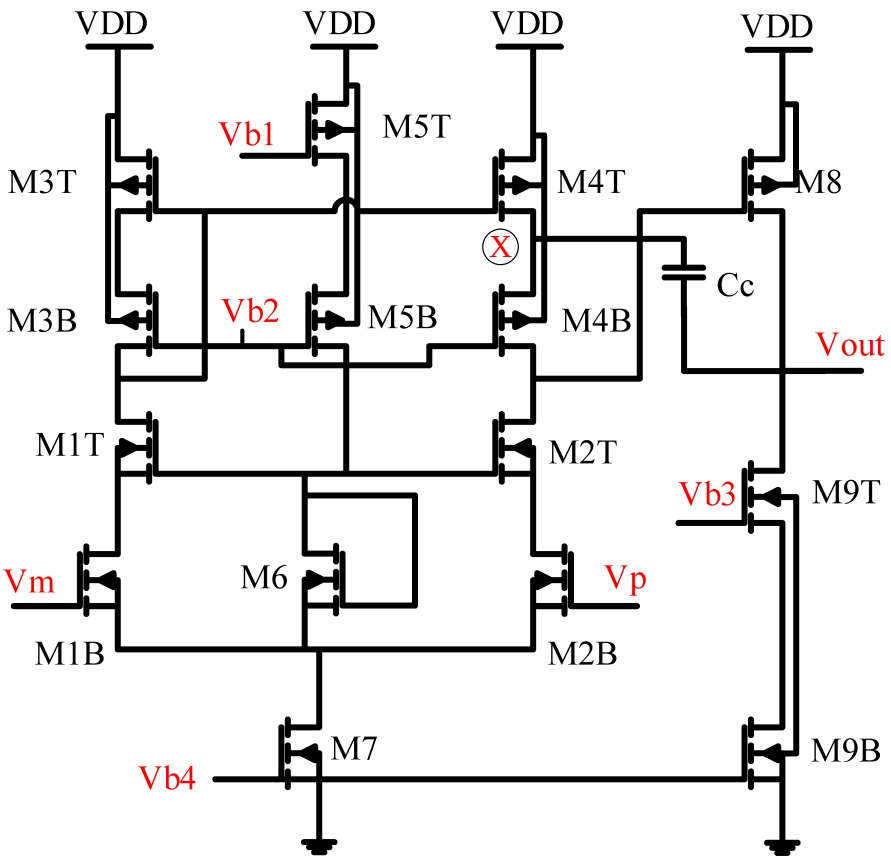

Figure 8.6: The schematic of the high-speed op-amp circuit.

op-amp. Fig. 8.7 shows the layout of the op-amp. Transistors M3, M4, M5, and M8 are laid out in a single common centroid stack (the top row). Deep-nwell NMOS transistors are used for M1, M2, and M6. Transistors M7 and M9 are also laid out in a single stack (the bottom row). The compensation capacitor is laid out as two 80 fF capacitors (connected in parallel) to make the layout symmetric.

The parasitic-extracted schematics are used to evaluate the performance of the op-amp. Fig. 8.8 (a) shows the gain and phase variations of the op-amp. The op-amp shows 89 dB of open-loop gain at DC. The gain-bandwidth product is about 700 MHz with a 65° phase margin. Fig. 8.8 (b) shows the output of the op-amp for the unity gain configuration while sweeping the input from 0 V to 1.8 V. The input

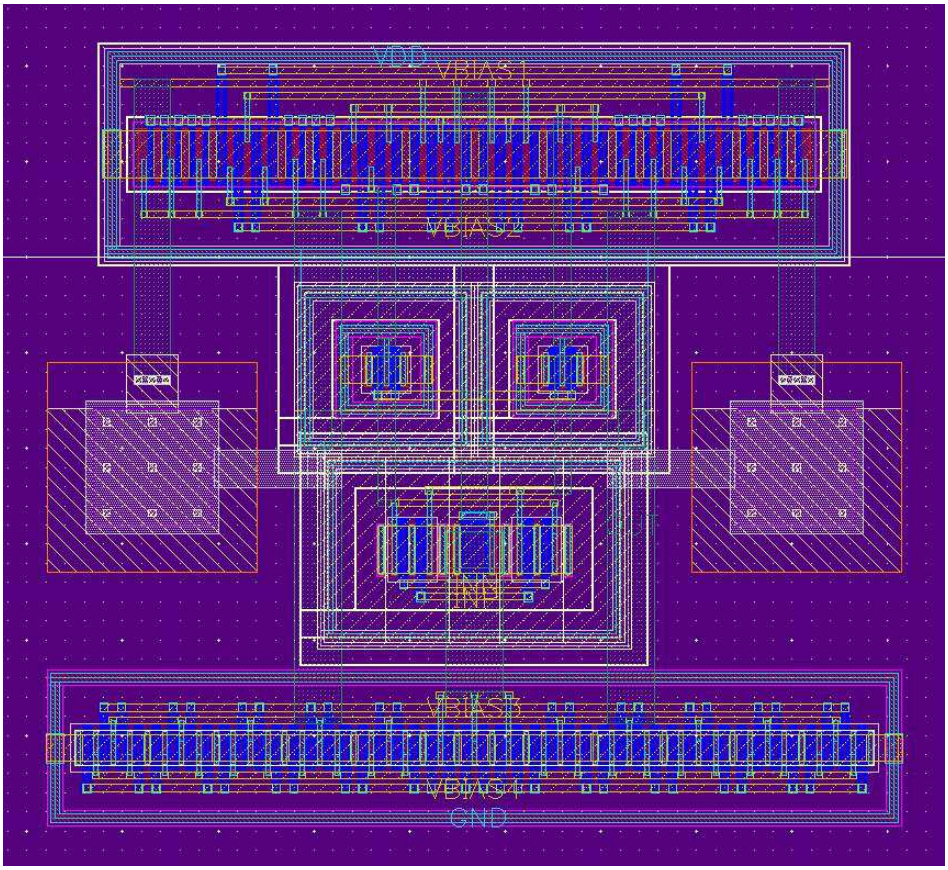

Figure 8.7: The layout of the op-amp.

common-mode range (ICMR) of the op-amp is reported as 1.6 V (the linear region of the curve). The slew rate of the op-amp is measured based on the response to a 1.4 V step input to the unity gain op-amp buffer (see Fig. 8.8 (c)). The resulting slew rate is 240 V/μs, which was calculated based on the falling edge of the response. Here, the falling edge slew rate is mainly restricted by the class A output stage of the op-amp. The input-referred noise of the designed op-amp is reported as 10 nV/$\sqrt{\text{Hz}}$ (see Fig. 8.8 (d)). Table 8.1 summarizes the simulated characteristics of the designed op-amp.

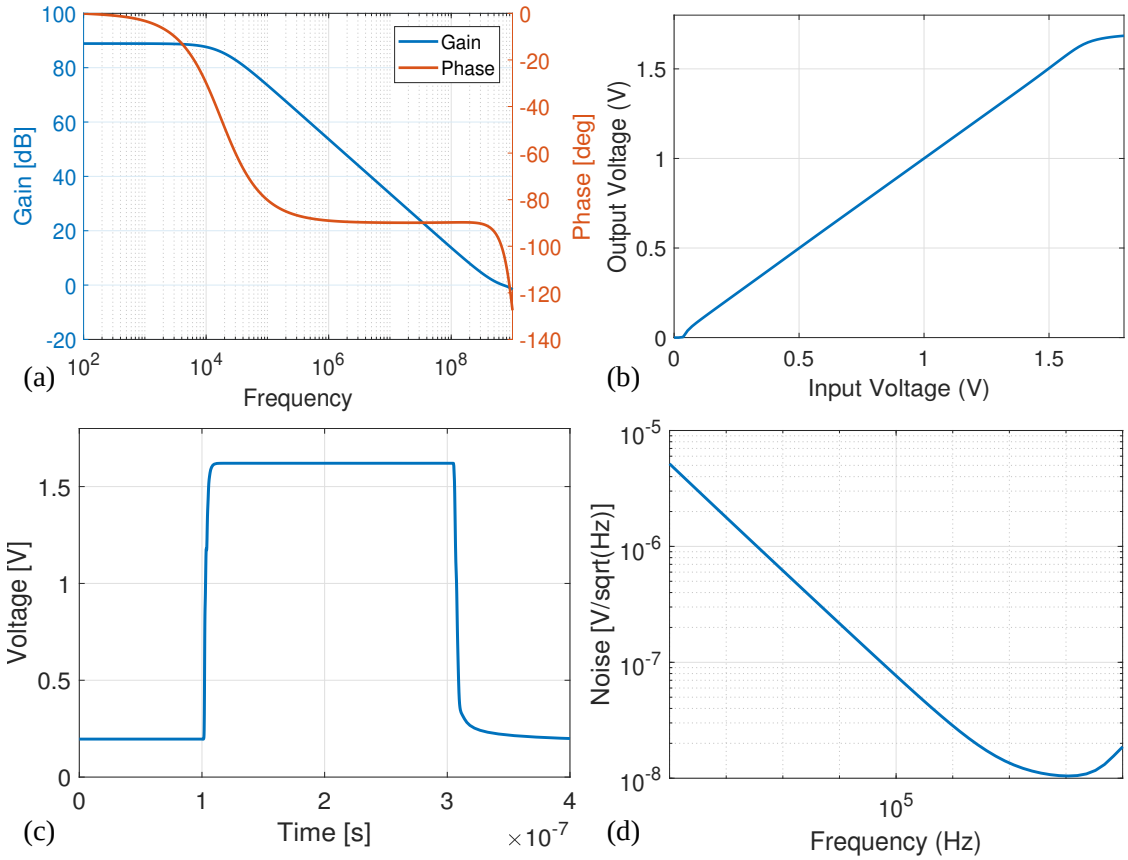

Figure 8.8: (a) Variation of the gain and the phase. (b) Output of the setup that measures the ICMR. (c) Simulation results to measure the slew rate. (d) Input referred noise of the designed op-amp.

8.2.4 CMOS All-pass Filter Design

An all-pass filter, consisting of cascaded resistor-capacitor networks, was designed. In the current system, the Laplace domain representation of an ideal all-pass filter $e^{-s\tau}$ is approximated as $e^{-s\tau} \approx \left[\frac{1-\frac{s\tau}{6}}{1+\frac{s\tau}{6}}\right]^3$, and is realized in an analog RC topology using a cascade of three first-order APFs. Fig. 8.9 shows the circuit diagram of the all-pass filter for three cascaded first-order stages with fixed capacitors (with value C) and variable resistors (with value R). The group delay τ of the all-pass filter is software programmable via variable resistors. In particular, a 4-bit binary- weighted

Table 8.1: Simulated characteristics of the designed op-amp.

Specification	**Result**
DC gain	89 dB
Phase margin	65
Power consumption	1.5 mW
Output swing	1.7 V
ICMR	1.6
Slew rate	345 V/us
Noise floor	10.1 nV/sqrt (Hz)

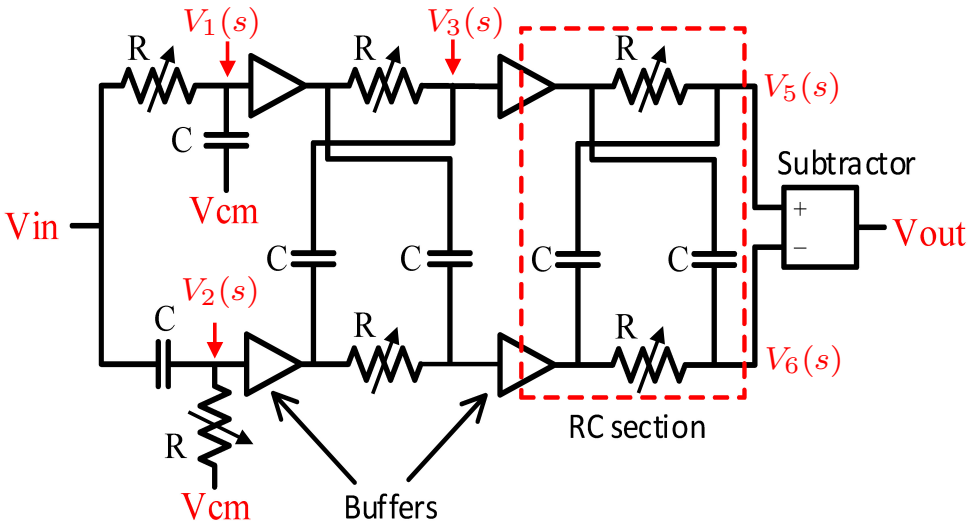

Figure 8.9: Schematic of the CMOS all-pass filter.

resistor DAC is used as the resistor. Buffers are employed in between each stage to minimize the loading.

Consider the first stage of the APF (first RC section). The s-domain voltage transfer functions $V_1(s)$ and $V_2(s)$ can be obtained as

$$V_1(s) = \frac{1}{1+RCs}V_{in}(s), \qquad V_2(s) = \frac{RCs}{1+RCs}V_{in}(s), \tag{8.5}$$

where $V_{in}(s)$ is the Laplace transform of the input alternating current signal $v_{in}(t)$. The voltage transfer functions $V_3(s)$ and $V_4(s)$ at output of the second stage are

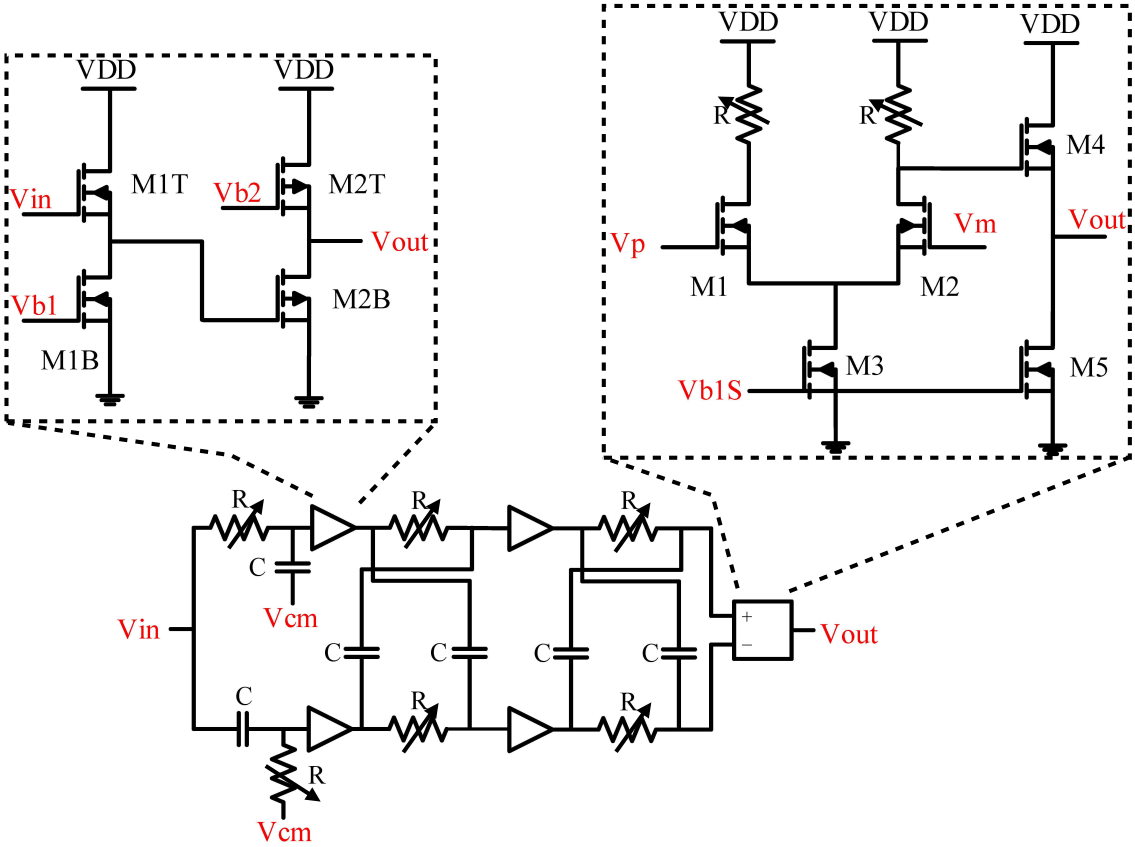

Figure 8.10: Schematics of the buffer and subtractor circuits.

obtained as

$$V_3(s) = \frac{1}{1+RCs}\left(\frac{1-RCs}{1+RCs}\right)V_{in}(s), \qquad V_4(s) = \frac{RCs}{1+RCs}\left(\frac{1-RCs}{1+RCs}\right)V_{in}(s). \tag{8.6}$$

If the two voltages at the output of the third stage are subtracted $(V_5(s) - V_6(s))$, the complete transfer function of the APF can be obtained as

$$\frac{V_{out}(s)}{V_{in}(s)} = \phi(s) = \left(\frac{1-RCs}{1+RCs}\right)^3 = \left(\frac{1-\frac{j\omega\tau}{6}}{1+\frac{j\omega\tau}{6}}\right)^3. \tag{8.7}$$

Here, the group delay of the APF is $\tau = 6RC$.

The intermediate buffer stages of the APF are realized using an NMOS-PMOS source follower pair. The bandwidth of the buffer circuits should be much higher than the system bandwidth, such that the group delay contribution from the buffers

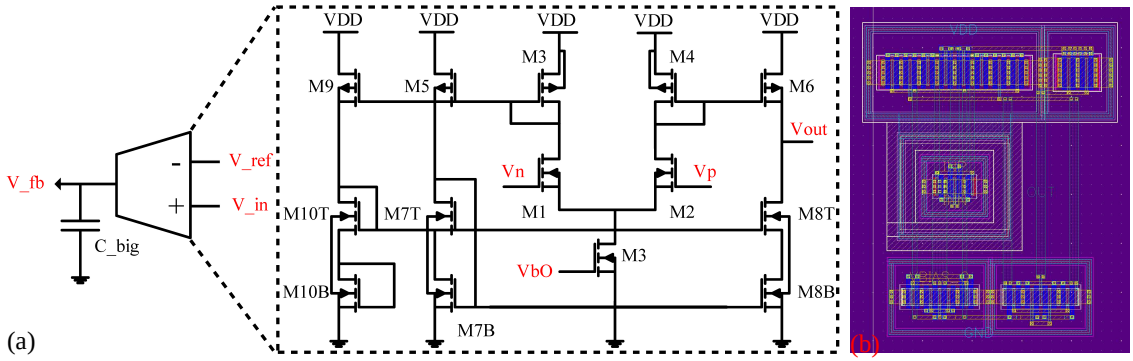

Figure 8.11: (a) Schematic of the OTA-based negative feedback circuit. (b) Layout of the OTA.

to the all-pass filter circuit is insignificant. Thus, a simple circuit with four transistors is designed. Fig. 8.10 shows the corresponding circuit. Transistors M1T and M2B operate as source followers (the input is fed into the gate of M1T and the output is taken from the source of M2B), while the other two transistors, M1B and M2T, are used to bias M1T and M2B, respectively. The group delay of the buffer remains constant at ∼110 ps throughout the frequency of interest. Note that the gain of the buffer is ∼0.8 dB less than unity, which requires compensation at the output of the APF (at the subtractor stage). The bias voltage Vb1 is supplied externally. The gate voltage of M2T is set by an internal self-feedback circuit. A negative feedback circuit was employed to maintain the output DC value of the buffer circuit at the desired value even when PVT variations are present. The feedback mechanism is essential, since the output DC voltage of the designed circuit is directly determined by the W/L ratios of the transistors, which can easily vary with PVT variations.

A simple analog circuit was used to subtract two voltages at the end of the APF. The proposed circuit only requires seven transistors and produces the output without using an op-amp. The group delay of op-amp-based subtractors is significant when compared to the all-pass filter delay, which makes it difficult to use with the proposed APF design. Fig. 8.10 shows the schematic of the subtractor. The first stage consists

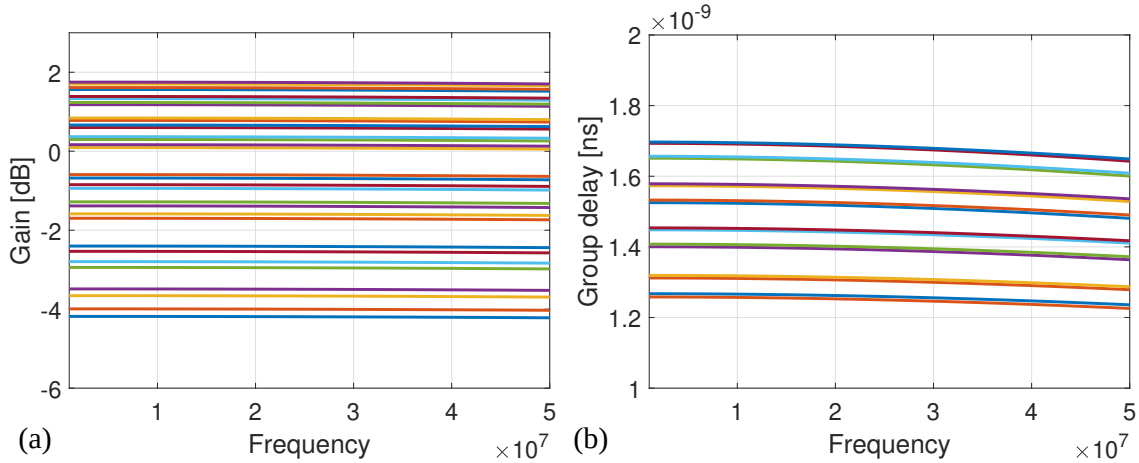

Figure 8.12: Tunability of the all-pass filter in terms of (a) the gain and (b) the group delay (with different resistor configurations in the DACs).

of a differential pair that produces the difference voltage. The output of the first stage is fed into a source follower before obtaining the output. This improves the output impedance of the all-pass filter. Ideally, the gain of the subtractor should be 1. However, the gain of the subtractor circuit was chosen such that the total gain of the APF circuit is unity (subtractor gain is higher than unity to compensate losses from the buffer stages and the second stage of the subtractor). Since the gain requirement is around 0 dB, resistors are employed as the load impedances of the differential stage (instead of transistors). The circuit is self-biased and the gate voltage of the M3 and M5 are controlled by a negative feedback loop such that the output DC voltage is fixed at the desired value.

An operational transconductance amplifier (OTA)-based negative feedback circuit was proposed to maintain the DC bias voltages of the buffer and the subtractor at desirable values. This makes the DC characteristics of the circuit insensitive to PVT variations and parasitics. A continuous-time transconductance (G_m) type integrator ($G_m - C$ integrator) was employed as the feedback system. Fig. 8.11 (a) shows the corresponding circuit. Here, C_{big} is a large capacitor, which was imple-

mented using MOS transistors. It was realized using a transistor (drain, source, and body of the transistor are connected to act as one plate of a parallel-plate capacitor, and the gate acts as the other plate of the capacitor). Together G_m and C_{big} form an integrator with a time constant $\frac{C_{big}}{G_m}$. The integrator compares the input signal voltage with a reference voltage and provides feedback such that it makes the two voltages equal. The time constant of the integrator should be much larger than $\frac{1}{\omega_{min}}$ to minimize the impact of the feedback circuit on the signal of interest, where ω_{min} is the lowest frequency of interest. Thus, all the transistors of the OTA were biased at sub-threshold. Cascoded current mirrors were utilized at the second stage to obtain an acceptable gain. The open-loop gain of the designed OTA was ∼65 dB.

Fig. 8.12 shows the parasitic extracted simulation results obtained from the software-programmable APF. The gain of all-pass filter is tunable in between -4 dB and 2 dB (see Fig. 8.12 (a)). The binary-weighted resistor DACs in the subtractor are utilized to tune the gain. Resistor DACs in the RC sections of the APF are used to tune the propagation delay from 1.2 ns to 1.7 ns (see Fig. 8.12 (b)). The tunability of the all-pass filter is important for compensating PVT variations following calibration.

8.2.5 Internal and Boundary Modules

Fig. 8.13 shows a schematic of the internal module. The output is the τ-delayed version of $E_z(i,t)$ (i.e. $E_z(i,t-\tau)$). Thus, the output can be directly fed into the neighboring modules. A unity gain buffer is employed prior to feeding the output. Next, consider the op-amp that performs the addition and subtraction. Capacitors are utilized instead of resistors to control the gains of the input and feedback signals, since capacitor ratios can be laid out more accurately in CMOS technology. Also,

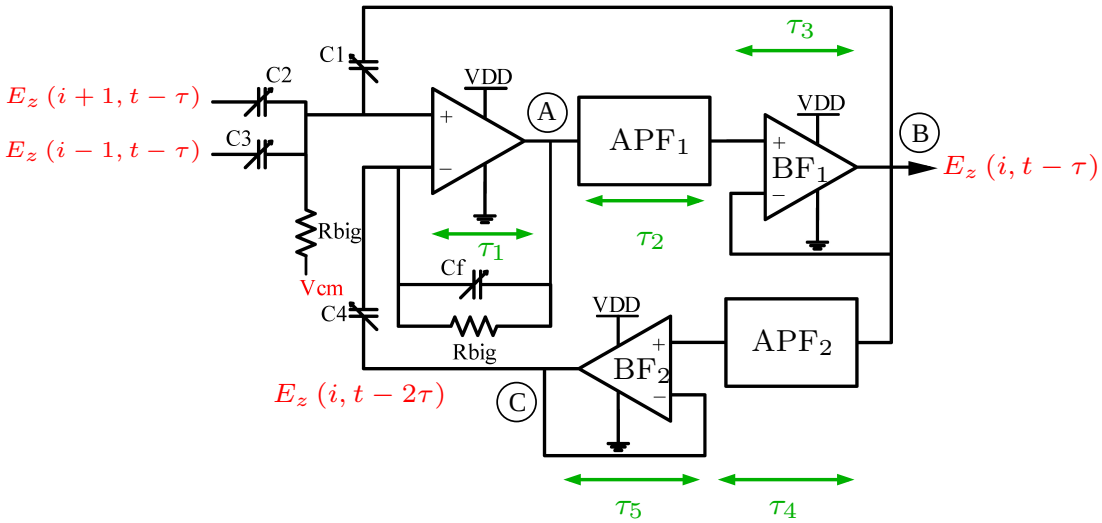

Figure 8.13: Schematic of the internal module of the 1-D wave equation solver.

capacitors ensure a DC gain of zero, which avoids building up any random DC offsets during the computation. In particular, 4-bit binary capacitor DACs (that can produce 16 different capacitor values) are used for compensating PVT variations, parasitics, and mismatches following calibration.

The implementation of CT delay τ is important for accurate computations in the AC (especially at high frequencies). As explained in Section 8.1.2, the PD compensation technique provides a convenient way of achieving this requirement. Also, the PD compensation is the key to achieve higher analog bandwidths from ACs (for CT computation), which is challenging in integrator-based methods employed in [8–11]. If analog bandwidth $F_{compute}$ and the spatial oversampling factor is fixed, the gain values of the op-amp mainly depend on τ (note that $K = \frac{c\tau}{\Delta x}$ defines gain values of the SDTC update equations). Consider the summing op-amp with a PD τ_1. The output of the op-amp is fed into all-pass filter APF$_1$ (with a group delay τ_2) followed by unity gain buffer BF$_1$ (with a group delay τ_3). Thus, the CT delay $\tau = \tau_1 + \tau_2 + \tau_3$. The signal at the output of the buffer (at B) can be denoted as $E_z(i, t-\tau)$. The resulting signal is then fed into the neighboring modules and

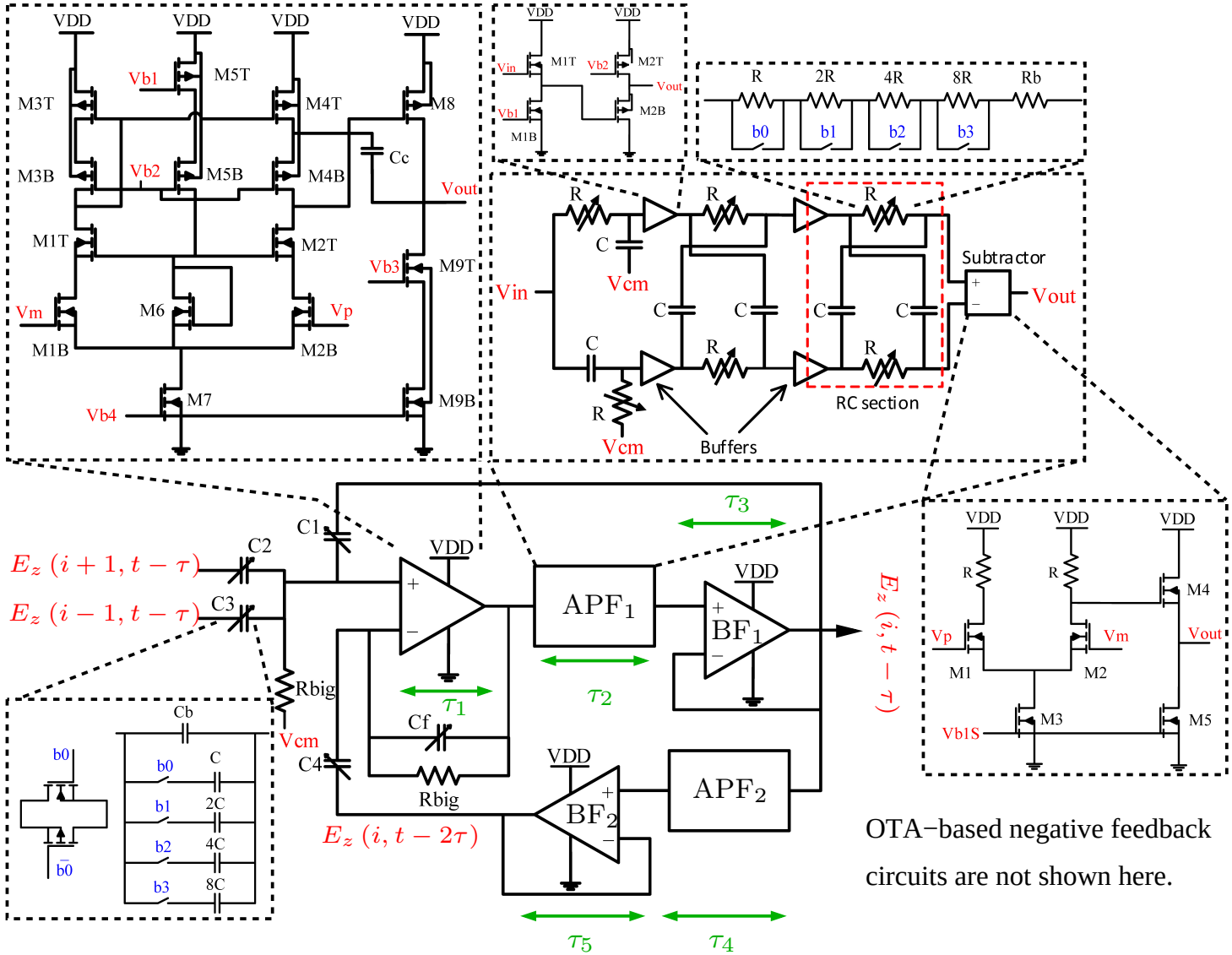

Figure 8.14: Complete schematic of the internal module including each CMOS circuits.

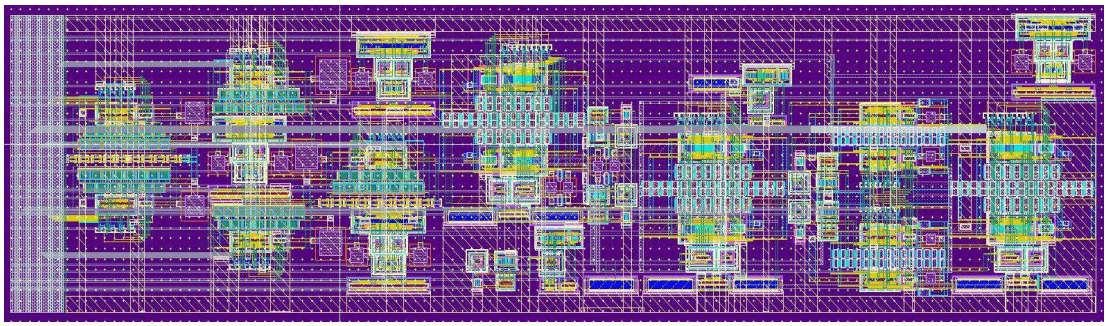

Figure 8.15: Layout of an internal module.

all-pass filter APF_2, which has a group delay τ_4 (in the feedback path). A unity gain buffer (with a group delay of τ_5) is cascaded with APF_2 to being fed to the summing op-amp. In the design, APF_1 is a fixed group delay all-pass filter (fixed resistors instead of resistor DACs), where as APF_2 is a tunable all-pass filter. The group delay τ_4 (of APF_2) is configured such that $\tau_4 = \tau - \tau_5$; and the signal at the output of BF_2 (at C) is $E_z(i, t - 2\tau)$. Neumann boundary, Dirichlet boundary, and radiation boundary modules are implemented following the same architecture.

Fig. 8.14 shows the complete schematic of the internal module including each CMOS circuit. The figure also shows the internal structure of the capacitor and resistor DACs. Transmission gates are used as the switches in the DACs. Programmable gains of the op-amp (capacitor DACs) can also be used to simulate multiple propagation media. Furthermore, they can be used to compensate for PVT variations and device parasitics. However, the introduction of capacitors blocks the DC path required to bias the op-amp. Thus, two large resistors are utilized to maintain the DC bias voltages. One resistor is tied between the non-inverting terminal of the op-amp and the common-mode voltage, while the other is tied between the output and the inverting terminal (see Fig. 8.14). Here, the large resistors are realized using a chain of PMOS pseudo resistors (also known as an adaptive element) to increase

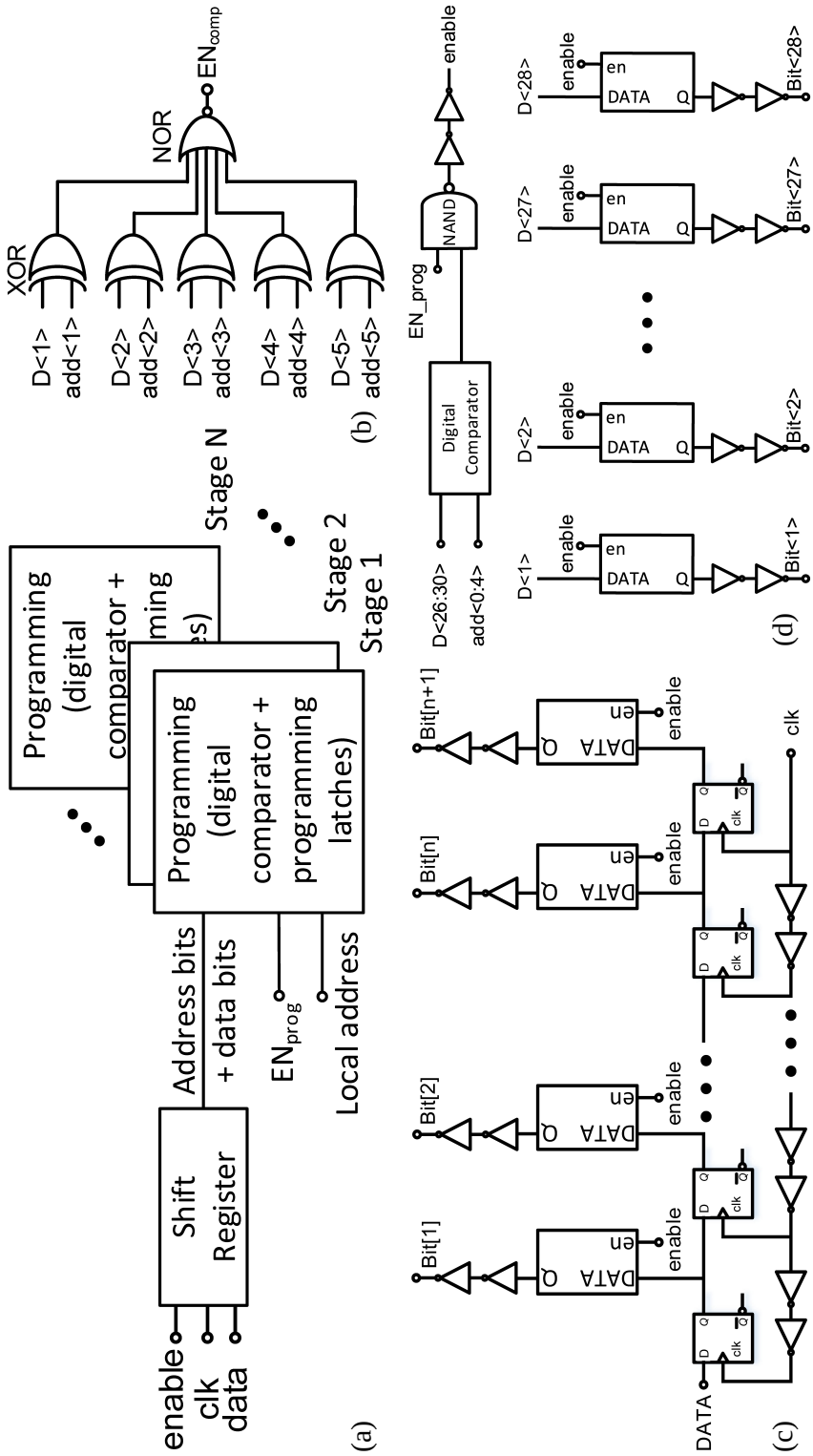

Figure 8.16: (a) Block diagram of the complete SPI block. (b) Schematic of the 5-bit digital comparator. (c) Schematic of the shift register. (d) Schematic of the programming block to program a specific stage.

the total resistance and linear range. Fig. 8.15 shows the complete layout of an internal module (without the digital programming module).

8.2.6 SPI Programming Module

Each module has 25 programming bits to configure (25×18 programming bits in total). Thus, a 30-bit (25 data bits + 5 address bits) serial peripheral interface (SPI) programming module was used to program the entire chip and save input/output pads. The complete block diagram of the SPI module is shown in Fig.8.16 (a). Four wires were utilized to implement the SPI interface (**clk**, **data** (MOSI), **program**, and **store**). Therefore, the complete SPI module only uses four I/O pads. A shift register block is first to receive data from the external circuits. Each module has its own programming block to compare the address bits (5 bits) coming from the shift register with its local address. If they match and the **program** signal is high, then the rest of the programming data are used to program the specific module.

In the schematic of the shift register block, shown in Fig. 8.16(c), the block has 30 D flip-flops (DFFs) in series acting as a shift register. A digital buffer is used between DFFs to delay the clock so that one of the DFFs has enough time to latch the data from the output of the previous DFF. Latches are used to store the programming bits when the enable signal is low. Each module in the chip has its own programming block, as shown in Fig. 8.16(d). A digital comparator (shown in Fig. 8.16(b)) is first used to compare the local address with the address bits coming from the shift register. Since the address contains 5 bits, 5 XOR gates are used to first compare the local address with the address bits. If they match, the extra NOR gate in the digital comparator outputs high. In this case, when the **program** signal is high, this specific module is selected, and all programming bits are stored in the

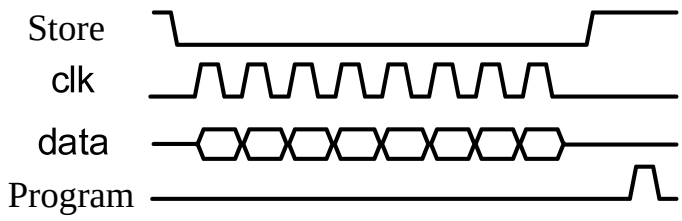

Figure 8.17: Timing diagram of the complete SPI block.

latches in this stage, as shown in Fig. 8.16(d), so that this stage is programmed successfully.

The timing diagram is shown in Fig. 8.17, where **store** is the enable signal for the shift register, **clk** (rising edge triggered) is the clock for the SPI bus, data is the data transferred through SPI bus, and **program** (high active) is the enable signal for the programming block.

8.2.7 Simulation Results

The proposed APDA-based AC that solves the 1-D wave equation was simulated using Cadence Spectre. A spatial grid of 16 spatial points is selected for the simulation. The analog bandwidth $F_{compute}$ is selected as 50 MHz. The spatial step size $\Delta x = \frac{\lambda_{min}}{6}$. The circuit satisfies the stability condition $\tau \leq \frac{\Delta x}{c}$. Fig. 8.18 (a) shows the spatio-temporal variation of the electric field for a Gaussian modulated cosine (GMC) electric field $E_m \cos(2\pi f(t-\tau_d))\, e^{-k(t-\tau_d)^2}$ excitation at the left boundary, where $E_m = 30$ mV, and $\tau_d = 0.5$ μs (left: 3-D view; right: top view). The center frequency of the GMC field is 40 MHz. The right boundary of the AC was simulated as a radiation boundary. The zero initial conditions were considered for all simulations. Fig. 8.18 (b) shows the simulation results for a sinusoidal signal at 40 MHz. Fig. 8.19 shows the simulation results obtained when the input excitation is at 30 MHz. Note that the group delay of the system varies with frequency and needs to be

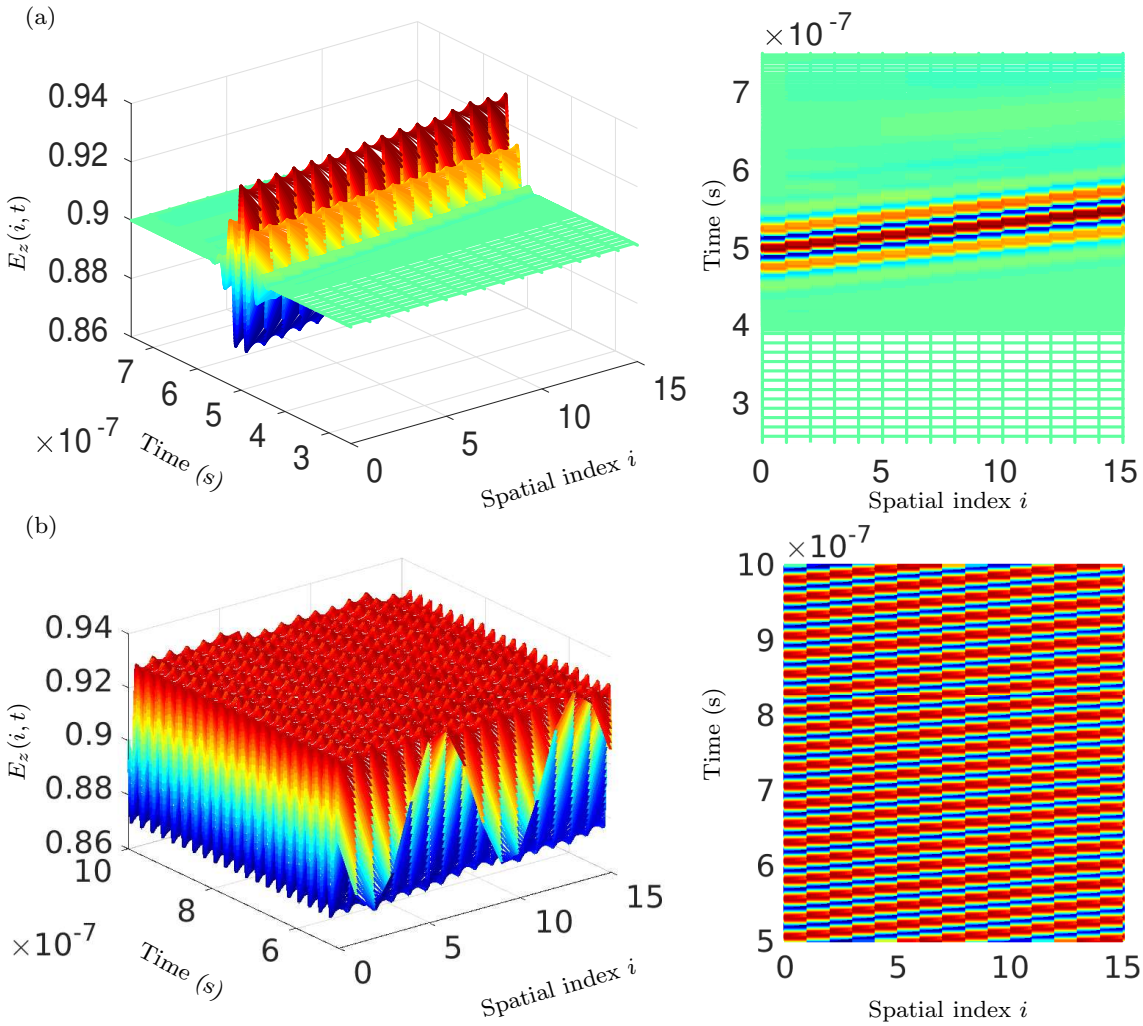

Figure 8.18: Obtained Cadence results from the CMOS AC when the system is excited using (a) a GMC electric field (b) a sinusoidal signal at the left boundary (left: 3-D view, right: top view). The center frequency of the signals are 40 MHz. The right boundary is simulated as a radiation boundary.

compensated prior to measurements. In the proposed system, the variable all-pass filter $\phi_2(s)$ in the internal module is used to tune the circuit. Other reconfigurable circuits (resistor DACs and capacitor DACs) can also be used to obtain the desired output (by compensating PVT variations and parasitics).

Fig. 8.20 shows the simulation results when the right boundary was simulated as a Dirichlet boundary (fixed at the common-mode voltage). Fig. 8.20(a) and

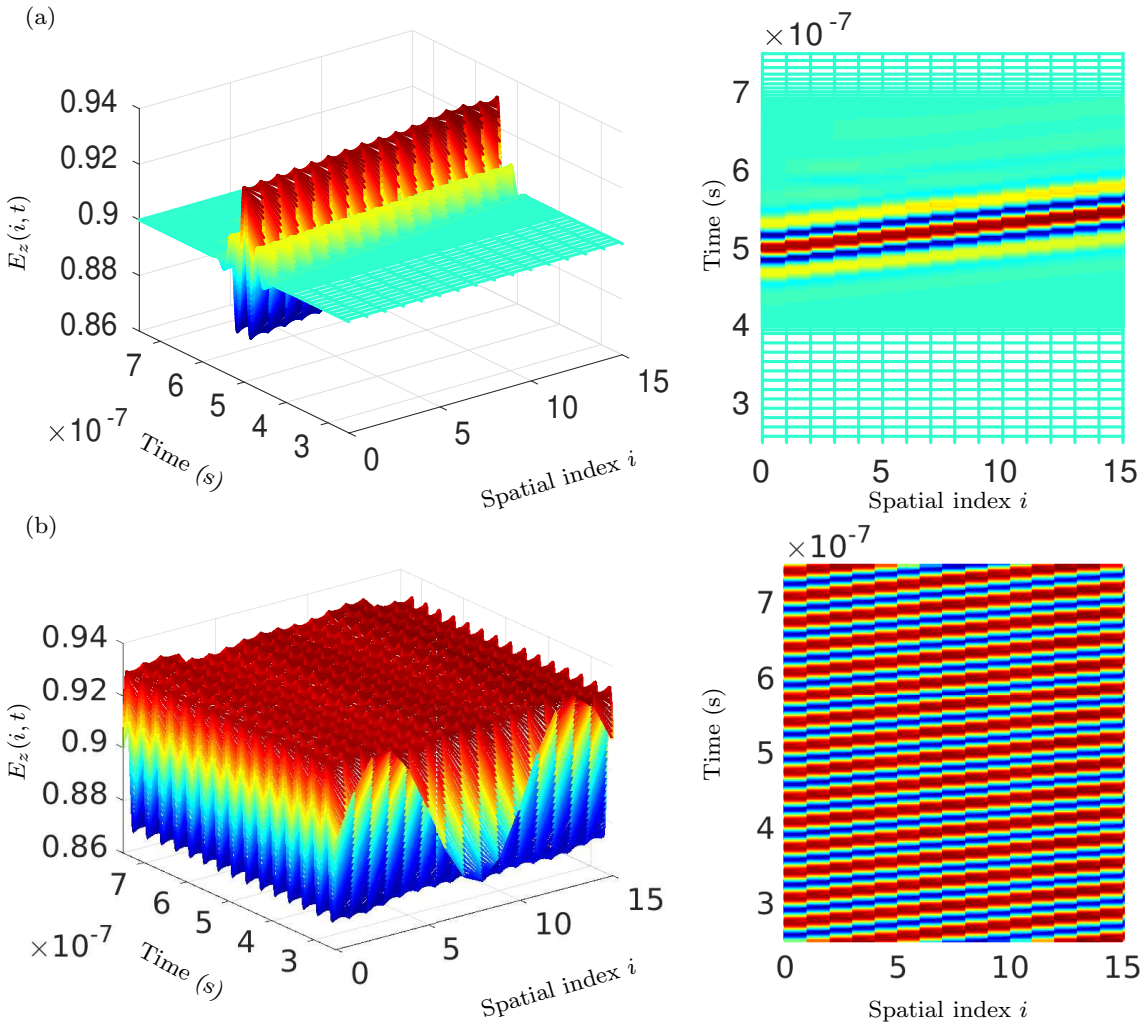

Figure 8.19: Obtained Cadence results from the CMOS AC when the system is excited using (a) a GMC electric field (b) a sinusoidal signal at the left boundary (left: 3-D view, right: top view). The center frequency of the signals are 30 MHz. The right boundary is simulated as a radiation boundary.

Fig. 8.20(b) correspond to GMC and sinusoidal excitation at 40 MHz, respectively.

The corresponding results for 30 MHz input signals are shown in Fig. 8.21.

8.2.8 Final Layout

Fig. 8.22 shows the final layout of the AC that computes the CT solution of the 1-D wave equation. The total chip size of the AC is 2 mm $\times$ 2 mm (including the pad

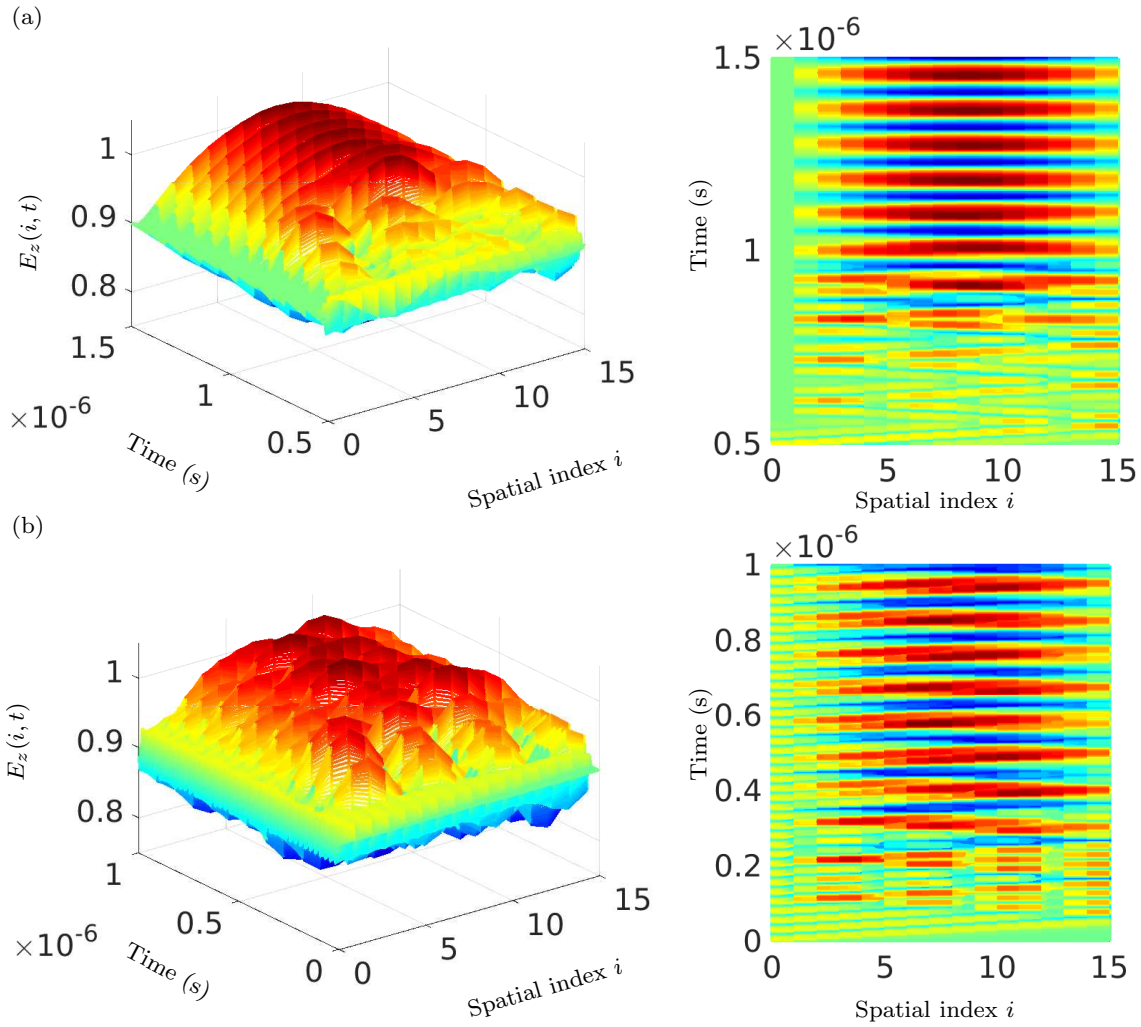

Figure 8.20: Obtained Cadence results from the CMOS AC when the system is excited using (a) a GMC electric field (b) a sinusoidal signal at the left boundary (left: 3-D view, right: top view). The center frequency of the signals are 40 MHz. The right boundary is simulated as a Dirichlet boundary (fixed at the common mode voltage).

frame). Each side of the layout (left and right) contains nine analog modules (right: M1-M9; left: M10-M18). The SPI bus and the programming modules are laid out between the two sets of modules. The AC also consists of separate circuit elements (op-amps and APFs) that can be tested individually for their characteristics and functionalities. The test circuits can also be used to check the operation of the SPI

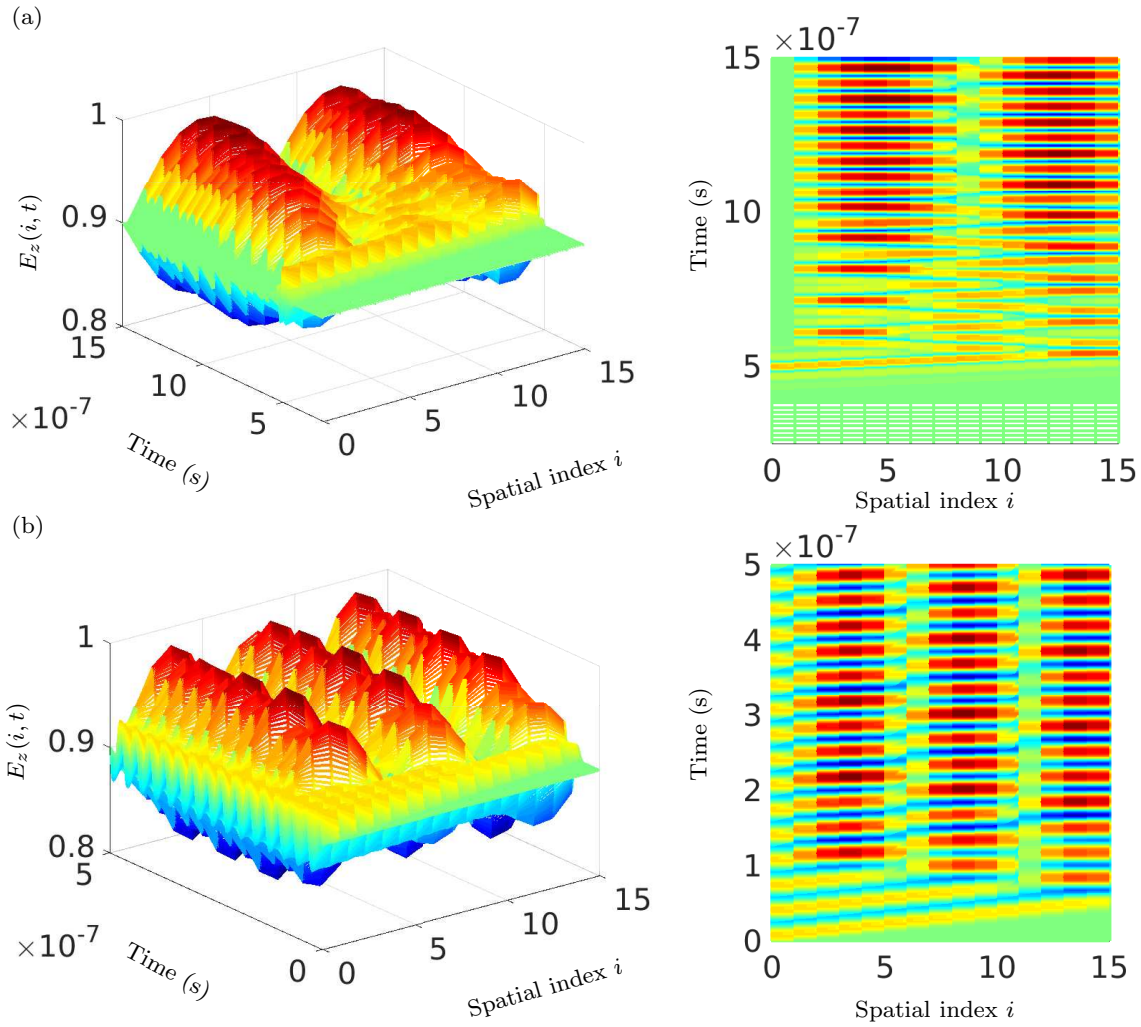

Figure 8.21: Obtained Cadence results from the CMOS AC when the system is excited using (a) a GMC electric field (b) a sinusoidal signal at the left boundary (left: 3-D view; right: top view). The center frequency of the signals are 30 MHz. The right boundary is simulated as a Dirichlet boundary (fixed at the common mode voltage).

interface, since these circuits are directly controlled by the SPI inputs. Furthermore, we have included several test points inside the SPI interface for individual testing of the SPI module. The selection of the boundary modules and the option of cascading chips are controlled by four separate control signals (they are not controlled using SPI signals). As a summary, the AC consists of 72 op-amps, 36 APFs, and 36 OTAs

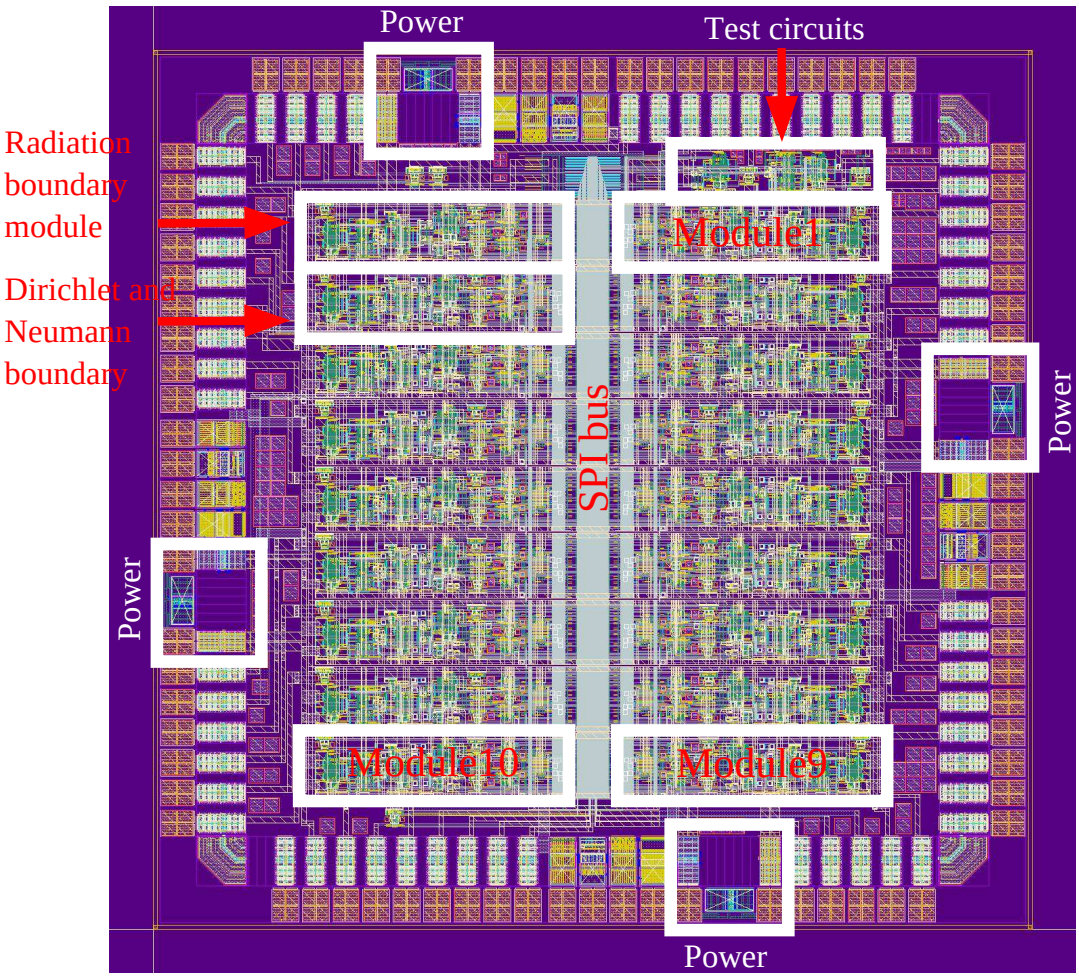

Figure 8.22: Final layout of the wave equation solving chip.

for the CT computation of the 1-D wave equation (4 op-amps, 2 APFs, and 2 OTAs per module). The simulated power consumption is ∼200 mW.

CHAPTER 9

MEASUREMENT SETUP AND CALIBRATION

We designed and implemented a software-programmable measurement setup to program, calibrate, and evaluate the fabricated AC that computes the approximate solution of the wave equation problem. It consists of the analog chip, FPGAs, ADCs, DACs, microcontrollers, and off-the-shelf microwave components. All computations were performed in the CT domain by using the special-purpose AC that inputs and outputs signals via matched 50 Ω transmission lines. The chip micrograph of the wave equation–solving AC is shown in Fig. 9.1. The 4 mm^2 chip (2 mm^2 of active area) consumes 200 mW of power. Input boundary boundary conditions were generated inside an FPGA and supplied to the analog chip through a DAC board. Computed analog solutions (computational outputs of the analog chip) are routed back into the FPGA through a 16-input ADC board. Reconfiguration and calibration commands were sent by an Atmel microcontroller. Bias voltages are also supplied using a microcontroller through a 16-bit DAC. Variable gains, delays, and bias voltages were first optimized through a series of calibration stages. For the calibration, the simultaneous perturbation stochastic approximation (SPSA) algorithm is used as the optimization technique [135,136]. Fig. 9.2 shows an overview of the measurement setup. Each supporting instrument was connected to a computer using Ethernet and universal serial bus (USB) connections. The computer was used to configure control inputs, run calibration algorithms, and collect/analyze results.

9.1 Measurement Setup

This section provides a detailed description of the measurement setup that is used to obtain measurements of the AC. A brief description of the FPGA designs that are used to generate input data and capture solution outputs are also provided. This

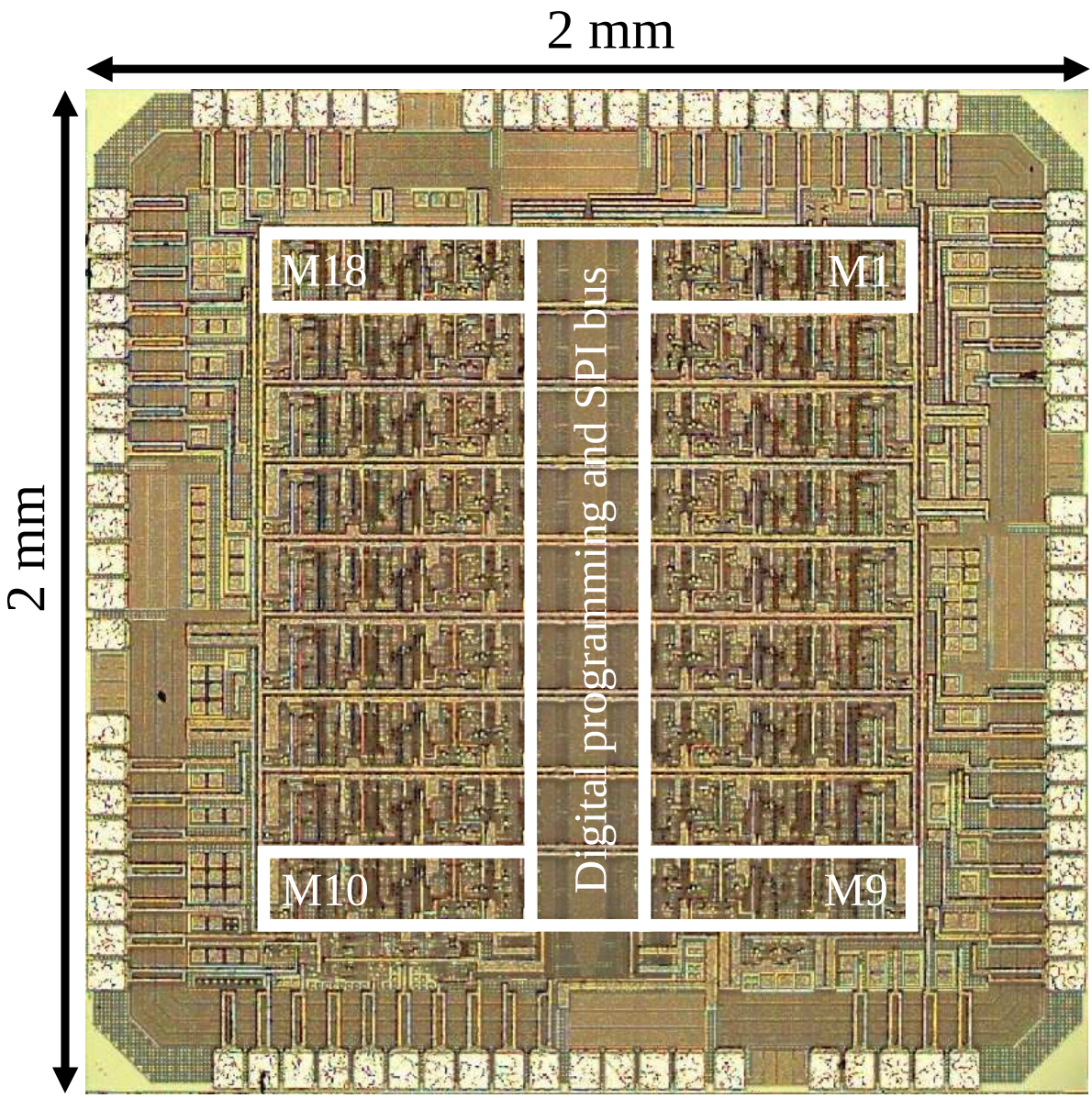

Figure 9.1: Die micrograph (die size: 4 mm^2, active area: 2 mm^2). Each side of the analog chip (left and right) consists of nine analog modules (right: M1-M9; left: M10-M18). The SPI bus and the programming modules are laid out in between the two sets of modules. The AC consumes 200 mW of power and consists of 72 op-amps, 36 APFs, and 36 OTAs for the CT computation of the 1-D wave equation (4 op-amps, 2 APFs, and 2 OTAs per module).

section also discusses the initial test measurements that were used to characterize the chip.

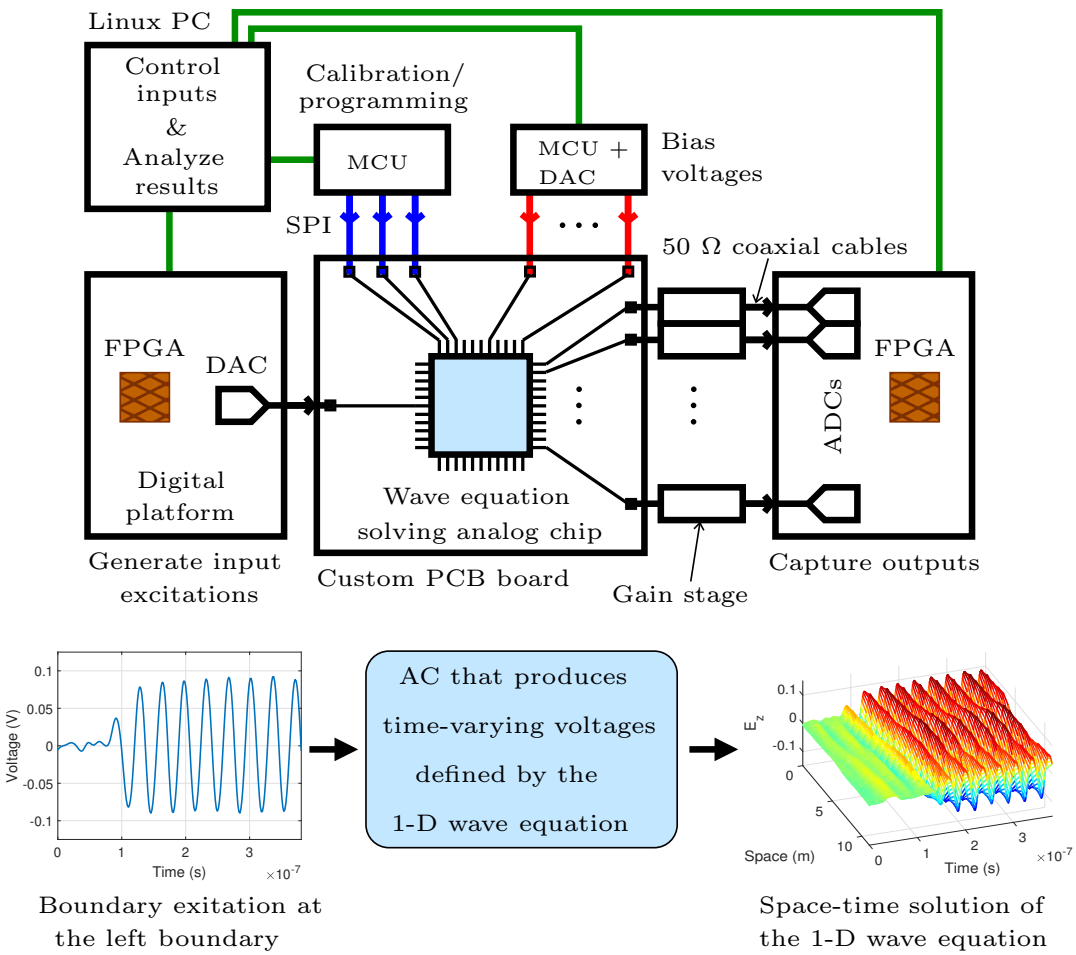

Figure 9.2: Overview of the designed analog-digital hybrid computing architecture.

9.1.1 Evaluation Board

A custom PCB was designed to test and evaluate the analog computing chip. The evaluation board is shown in Fig. 9.3, which is fully populated with circuit components. EAGLE PCB design software is used to design the board layout. The board consists of four layers. The top layer routes most of the data signals, whereas the second, third, and fourth layers serve as ground, power and ground layers, respectively (Appendix 11.2 shows the layout and the schematic of the evaluation board). The analog chip at the center of the board connects all inputs and outputs via matched 50 Ω transmission lines. Here, we used U.FL connectors to connect the

evaluation board to the external circuits (ADCs and DACs). Hirose U.FL connectors are convenient miniaturized replacements for SMAs and they reduce the size of the board and the lengths of the input/output transmission lines. Off-chip op-amp buffers were used in between the chip outputs and the U.FL connectors in order to isolate the chip outputs from the external circuits (ADCs and the oscilloscope). Off-the-chip buffers ensure that the chip would not be loaded by large capacitive loads. Bias voltages were supplied through voltage regulators followed by potentiometers. Each bias voltage was connected to the chip through a pin header, such that the bias voltages could be easily disconnected from the chip if needed. The pin headers are also important to drive the bias voltages directly using external power supplies such as DACs. Reconfiguration and calibration SPI commands were sent by an Atmel microcontroller. Three dip switches were used to configure different boundary conditions (radiation boundary, Dirichlet boundary, and Neumann boundary) as well as the selection of the two cascading chips.

An infrared reflow oven was used with a soldering stencil to populate the evaluation board. The use of a soldering stencil made it easier to solder tiny surface mount devices (SMDs). When soldering, it is important to make sure that all the pads of the SMDs are lined up with the stencil before pasting any solder as well as to ensure that the stencil does not move when pasting the solder. Surface mount components were carefully placed using a tweezer, and a BestEquip T962 reflow oven was used to melt the solder. Fig. 9.3 shows an image of the soldered board (including through-hole components).

After soldering, the components were visually inspected using a digital microscope. Bad connections were resoldered to improve the soldering condition. A multimeter was then used to confirm the connectivity throughout the PCB. Finally,

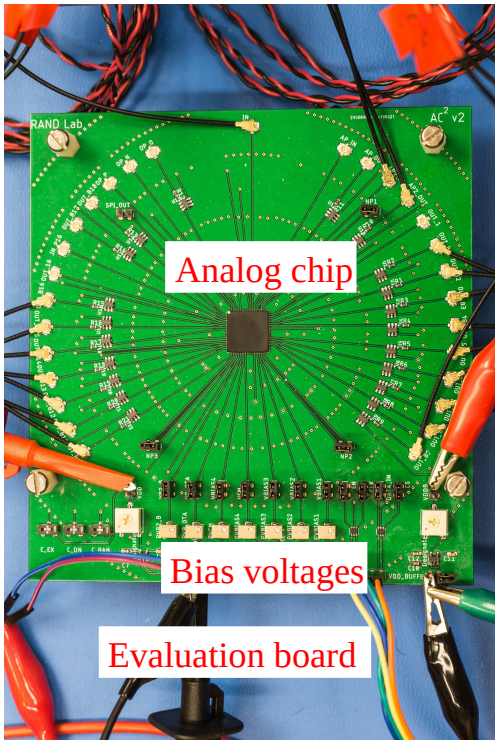

Figure 9.3: The custom-designed PCB used to test and evaluate the chip.

all power regulators and potentiometers had been checked for expected outputs by supplying a direct current voltage and probing each output using a multimeter.

9.1.2 Programming/calibration Data

An SPI interface with an additional control wire is used to program and calibrate the analog chip. The proposed programming/calibration interface consists of four wires (**clk**, **data**, **store**, **program**), which transfers data serially. The **data** bit-stream consists of 18 control words. Each control word is 30 bits wide. One word is used to configure the capacitor and the resistor DACs in a single module (in the analog chip). The first 5 bits of the control word represents the address of the module that needs to be configured. Each module has a unique 5-bit address starting from 00001 and ending at 10010. If the address of the module and the first 5 bits of the

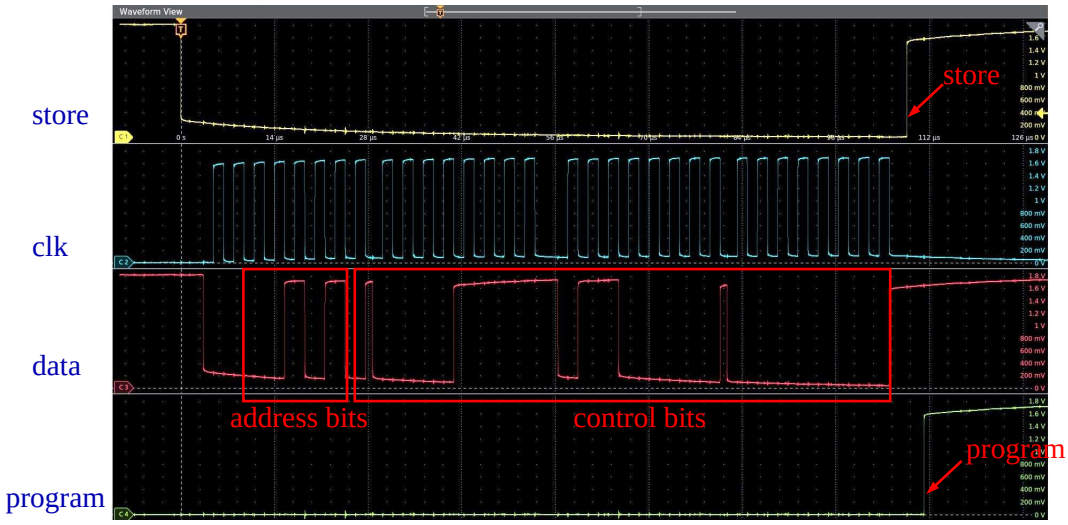

Figure 9.4: Captured oscilloscope data from the SPI interface (shows when the control word is for module 5).

control word match, then the rest of the programming data (the remaining 25 bits) are used to program the specific module. The structure of the 25 programming bits is as follows:

- Bits 0–11: For capacitor DACs to control the gains of the summing op-amp.
- Bits 12–16: For resistor DAC of the subtractor to control the gain of the all-pass filter.
- Bits 17–20: For resistor DAC of the all-pass filter to control the group delay of the all-pass filter.
- Bits 21–24: For capacitor DACs to control one of the gains of the summing op-amp.

The programming/calibration bitstream was generated using an Atmel microcontroller (an Arduino board). They were transferred to the evaluation board through the SPI interface of the Arduino board. The built-in SPI library of the Arduino programming language was used to generate **data** and **clk** bitstreams. Two control

signals **store** and **program**, that are required to store and program the control word, were generated separately by configuring two general-purpose input/output pins. Four wires were first observed using an oscilloscope to confirm the generation of the correct bitstreams. Fig. 9.4 shows an oscilloscope image that captures the generated SPI control word for Module 5 (module address 00101). Note that the 3 V outputs from the Arduino board were level shifted to 1.8 V prior to being fed to the chip. These data were captured at the outputs of the on-board level shifters.

9.1.3 Current-voltage Curves of the On-chip Current Sources/sinks

Three pairs of nmos/pmos current sinks/sources (diode-connected transistors) have been placed in three different corners inside the chip. During the initial testing of the analog chip, its current-voltage (I-V) characteristics can be used to characterize the fabrication variations inside the chip (to estimate transistor process corners). The bias voltages (gate voltage of current sinking or sourcing transistors) that correspond to the given bias currents have been measured using a source meter. Figs. 9.5 (a) and (b) show the measured I-V curves of the NMOS current sinks and PMOS current sources, respectively. Here, the drain current varies between 5 uA and 120 uA. Based on the measurements, all three pairs of transistors have similar I-V characteristics (PMOS-1 is slightly different). This data is compared with the simulation results to adjust the bias voltages such that the required current is applied through the transistors. The corresponding simulated I-V curves are shown in purple in this figure. It is noticed that the measured gate voltages of the PMOS current sources vary by ∼50 - 75 mV as compared to the voltages in the schematic-level simulations.

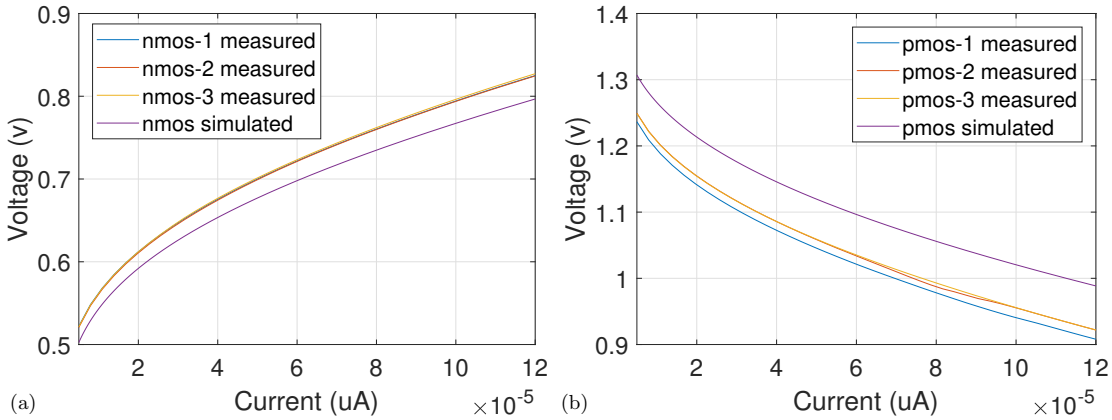

Figure 9.5: Measured I-V curves of the (a) NMOS current sinks and (b) PMOS current sources. Corresponding simulated I-V curves are shown in purple.

This difference was taken into consideration when biasing PMOS transistors in the circuit.

9.1.4 Initial Testing

In the initial tests, the chip was powered up using all bias voltages (based on the measured and simulated IV results). The chip draws about 115 mA of current, which is the expected value. Next, the direct current voltages at each output were checked and were found to be near the expected voltage of 900 mV. Two test points inside the chip were selected to evaluate how well the SPI interface functions for different control/calibration inputs: i) at the shift register and ii) at the input of the programmer. The correct functionality of the SPI interface was confirmed by probing the test point outputs and capturing them using an oscilloscope. Furthermore, two separate all-pass filers and one op-amp were selected to test and evaluate the functionality of the basic building blocks of the wave equation solver. APFs were tested by exciting them using sinusoidal signals. A signal generator was used to input the signals to the chip, and the outputs are captured using an oscilloscope.

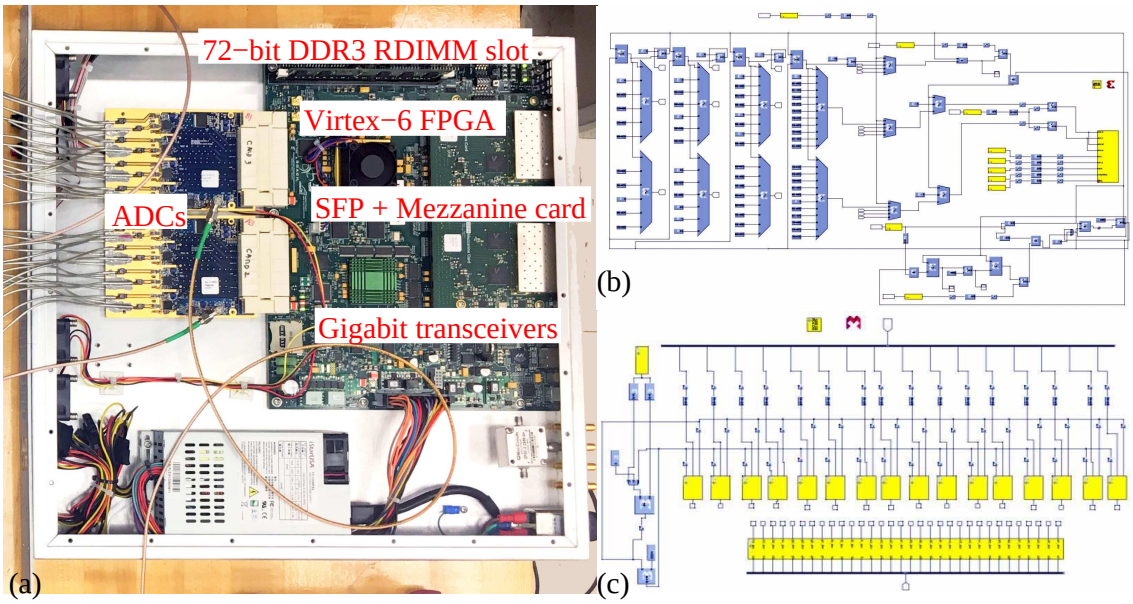

Figure 9.6: (a) ROACH-2 hardware platform with ADC connected. MATLAB Simulink-based digital designs that were used to (b) to generate boundary conditions and input excitations and (c) capture computed analog solutions.

The tunability of the APF in terms of the gain and group delay of the all-pass filter was verified by changing the resistor values of the resistor DACs. This was another test that was used to verify the functionality of the SPI programming interface.

9.1.5 Digital FPGA Designs

Digital architectures were designed to generate required boundary excitations and to capture computed analog waveforms. They were designed in the MATLAB environment using the Xilinx system generator (XSG). The designs were targeted to realize on Reconfigurable Open Architecture Computing Hardware-2 (ROACH-2) FPGA platform.

ROACH-2 is an open-source FPGA platform designed in Cape Town, South Africa, together with collaborators from the Collaboration for Astronomy Signal Processing and Electronics Research (CASPER) as part of the Square Kilometer

Array (SKA) radio telescope project. The core processing module of ROACH-2 is a Xilinx Virtex-6 sx475t FPGA (XC6VSX475T-1FFG1759C) chip, equipped with on-board inter-FPGA links interfaced with 10Gb Ethernet interfaces using SFP+ mezzanine cards for all cross-FPGA communications. Key peripherals of a ROACH-2 consists of two multi-gigabit transceiver card slots supporting 4x10Ge links (SFP+), four 72Mb QDR II+ SRAMs and a 72-bit DDR3 RDIMM slot connected to the FPGA along with two ZDok+ interfaces supporting a high-speed DAC/ADC. A wide range of ADC boards are available that can be interfaced with ROACH-2 that have 1, 4, 8 and 16 ADCs with sampling rates up to 240, 480 and 960 MHz per output. Fig. 9.6 (a) shows the ROACH-2 board that we used to realize digital architectures.

Figs. 9.6(b) and 9.6(c) show the digital architectures that we designed to generate the input excitations and capture computed solutions, respectively. The former creates a predefined digital signal inside the FPGA and converts it to its analog counterpart using the CASPER - DAC2x1000-16 DAC, which supports 16-bit accuracy at a maximum clock frequency of 1 GHz. The proposed FPGA architecture was designed such that the amplitude and the frequency of the generated signal are able to vary on the fly. The design is also able to configure the duration of the signal. The Simulink design shown in Fig. 9.6(c) utilizes the CASPER - ADC16x250-8 ADC to capture computed analog solutions. This ADC card supports 16 channels with 8-bit accuracy. The ROACH-2 FPGA supports two such ADC cards. Thus, the ROACH-2 supports a total of 32 channels, which are capable of operating at a maximum frequency of 230 MHz. The captured digital data was stored using an array of block random access memories (RAMs), which was then transferred to the computer through an Ethernet connection (for post-processing).

9.1.6 Software-programmable Test Bench

A software programmable test bench was designed and implemented using FPGAs, ADCs, DACs, and microcontrollers such that the analog chip can be easily programmed, calibrated, tested and evaluated for its functionality. Each of the supporting instruments was accessible through a network connection and could be programmed from a single computer. A set of MATLAB, C++ and Python scripts was used to communicate with each instrument (and program). Fig. 9.7 shows the corresponding analog-digital hybrid computational platform that has been designed using the analog chip and the ROACH-2 FPGA platforms.

All computations were performed in the CT domain by using the analog chip that inputs and outputs signals via 50 Ω transmission lines. U.FL connectors were used to connect the board with ADCs and DACs. Input excitations and boundary conditions were generated inside an FPGA and supplied to the analog chip through the DAC board. Computed analog solutions (computational outputs of the analog chip) were routed back into the FPGA through the ADC board. Both ADC and DAC boards were programmed and controlled using Python scripts. Bias voltages were supplied through voltage regulators followed by potentiometers. During the calibration, they were supplied using an eight-channel DAC board, which was controlled by a microcontroller (via an I2C interface). Reconfiguration and calibration SPI commands were also sent by an Atmel microcontroller. MATLAB and C++ scripts were used to communicate with the microcontrollers.

9.2 Calibration

The calibration of the AC is an important factor to improve the accuracy of the computation. Thus, several calibration steps have been employed. The calibration

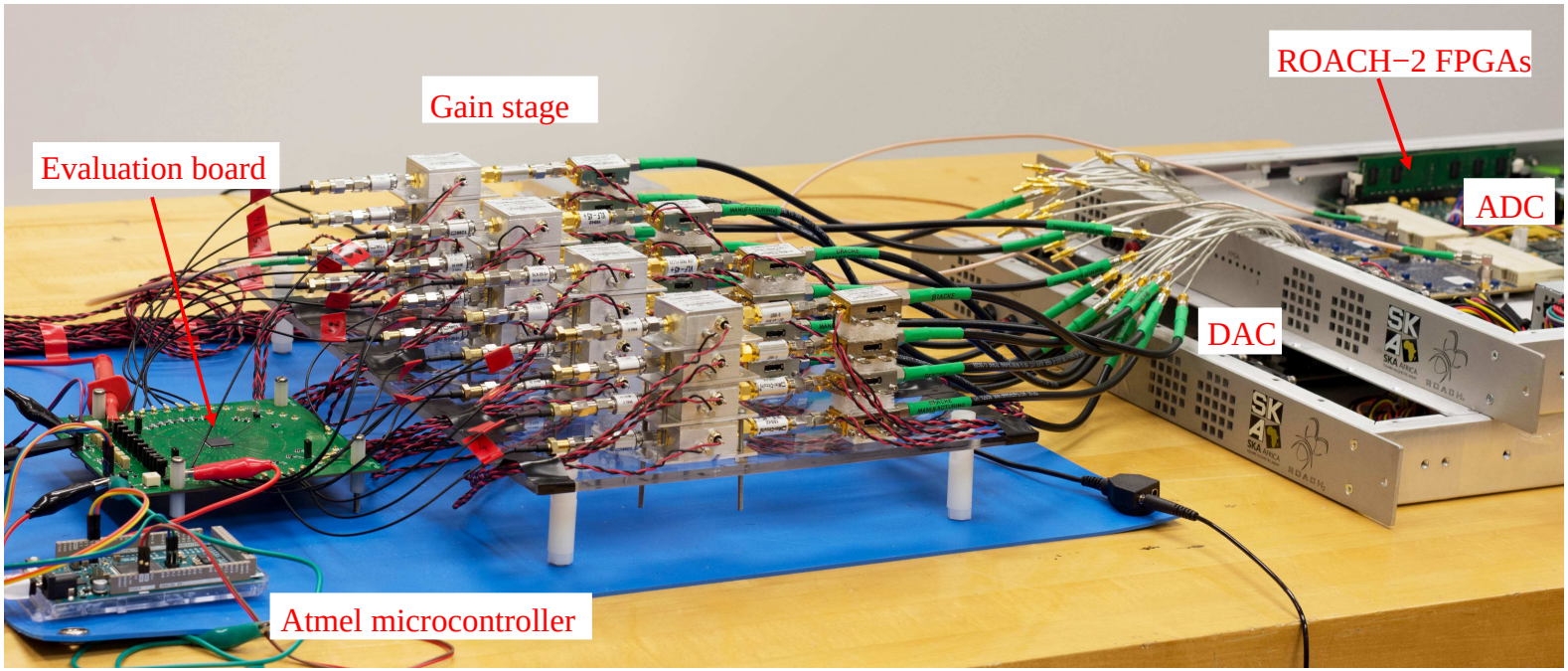

Figure 9.7: The analog-digital hybrid computational platform that used to test the analog wave equation solving chip.

was performed when the right boundary of the spatial grid was realized with a radiation boundary condition. The SPSA algorithm was utilized as the optimization algorithm for most of the stages, where the objective function (or loss function) of the optimization algorithm was selected as the expectation of noise-to-signal energy ratio $E(\gamma)$ [135, 136]. This method does not require information about the gradient of the objective function. Instead, it approximates the gradient using objective function measurements. The SPSA method only requires two measurements per iteration (regardless of the dimension of the problem) to estimate the gradient of the objective function [135, 136]. This was one reason to select the SPSA method over other stochastic methods, as those methods require $2p$ number of measurements for a p-dimensional problem.

9.2.1 SPSA Optimization Algorithm

The SPSA is a recursive optimization algorithm that relies only on measurements of the objective function to be optimized, and not on direct measurements of the gradient (derivative) of the objective function [135, 136]. Thus, the SPSA algorithm does not require information about the relationship between the objective function and the parameters to be optimized. This is important for complex optimization problems that contain a large number of parameters to optimized. This method approximates the gradient of the objective function using two measurements around the current parameter values. More importantly, the number of measurements does not increase with the dimension of the optimization problem as in other stochastic optimization algorithms [135, 136]. The finite difference stochastic approximation (FDSA) algorithm (most common stochastic method) requires $2p$ number of measurements to estimate the gradient of a p-dimensional system [137]. In the SPSA

method, the two measurements are obtained by simultaneously perturbing all of the parameters. Note that this simultaneous variation is performed in a "proper" random fashion. The SPSA algorithm is capable of working with measurements of the objective function that are corrupted by noise. In addition, it should be noted that the loss function of the SPSA algorithm does not uniformly decrease as the iteration process proceeds, which is a common feature of all stochastic algorithms. However, the algorithm iteratively minimizes the objective function (on average). All these important and practical features of the SPSA algorithm were taken into consideration when selecting an optimization method for the calibration.

Consider an optimization problem that minimizes the (scalar) differentiable loss function $L(\theta)$, where θ is a p-dimensional vector that consists of all parameters to be optimized. The following is a step-by-step procedure of the SPSA algorithm that iteratively produces a sequence of estimates [135, 136].

- Step 1: **Initialization and coefficient selection:** Initialize the parameter vector $\hat{\theta}_0$ (where $\hat{\theta}_0$ is the intial estimate of θ). Pick non-negative coefficients a, c, A, α, and β for the SPSA gain sequences $a_k = \frac{a}{(A+k+1)^\alpha}$ and $c_k = \frac{c}{(k+1)^\beta}$. The accurate selection of a_k and c_k are important for the performance of the SPSA algorithm. The reference [135] provides some guidance on picking these coefficients.

- Step 2: **Generation of the simultaneous perturbation vector:** Generate a $p-$dimensional random perturbation vector Δ_k using a Bernoulli± 1 distribution with a probability of 0.5.

- Step 3: **Loss function evaluations:** Obtain two measurements of $L(\theta)$ around the current parameter set $\hat{\theta}_k$ (at $\hat{\theta}_k + c_k\Delta_k$ and $\hat{\theta}_k - c_k\Delta_k$).

- Step 4: **Gradient approximation:** Approximate the gradient of the objective function by using the following equation:

$$\hat{g}_k\left(\hat{\theta}_k\right) = \frac{L\left(\hat{\theta}_k + c_k\Delta_k\right) - L\left(\hat{\theta}_k - c_k\Delta_k\right)}{2c_k}.\Delta_k. \tag{9.1}$$

- Step 5: **Updating θ estimate:** Estimate the parameter vector $\hat{\theta}_{k+1}$ as

$$\hat{\theta}_{k+1} = \hat{\theta}_k - a_k\hat{g}_k\left(\hat{\theta}_k\right). \tag{9.2}$$

- Step 6: **Iteration or termination:** Return to Step 2 if the iteration progresses. If not, terminate the process.

9.2.2 Calibration of the Analog Computer

Each module of the AC consists of four 4-bit resistor DACs (to control the gains of the summing op-amp), one 5-bit resistor DAC that controls the gain of the APFs, and one 4-bit DAC that tunes the group delay of the APFs. Additionally, there are seven bias voltages common to each module. Thus, the AC has 115 parameters (6×18+7) in total for calibration. As the initial step of the calibration process, optimum values for high sensitivity variables were determined, while all other variables were fixed. The gain and the group delay of the APFs were identified as the most sensitive variables of the objective function $E(\gamma)$. Here, the optimum values were found out using an exhaustive search. The objective function was recorded for all possible DAC configurations (a 2-D plot). There are 512 different configurations based on the two resistor DACs (4-bit and 5-bit). Here, the same DAC values were used for each module (in all 18 modules) when recording the measurements for a particular configuration. Optimum resistor DAC values were selected based on the minimum value of $E(\gamma)$ in the 2-D space.

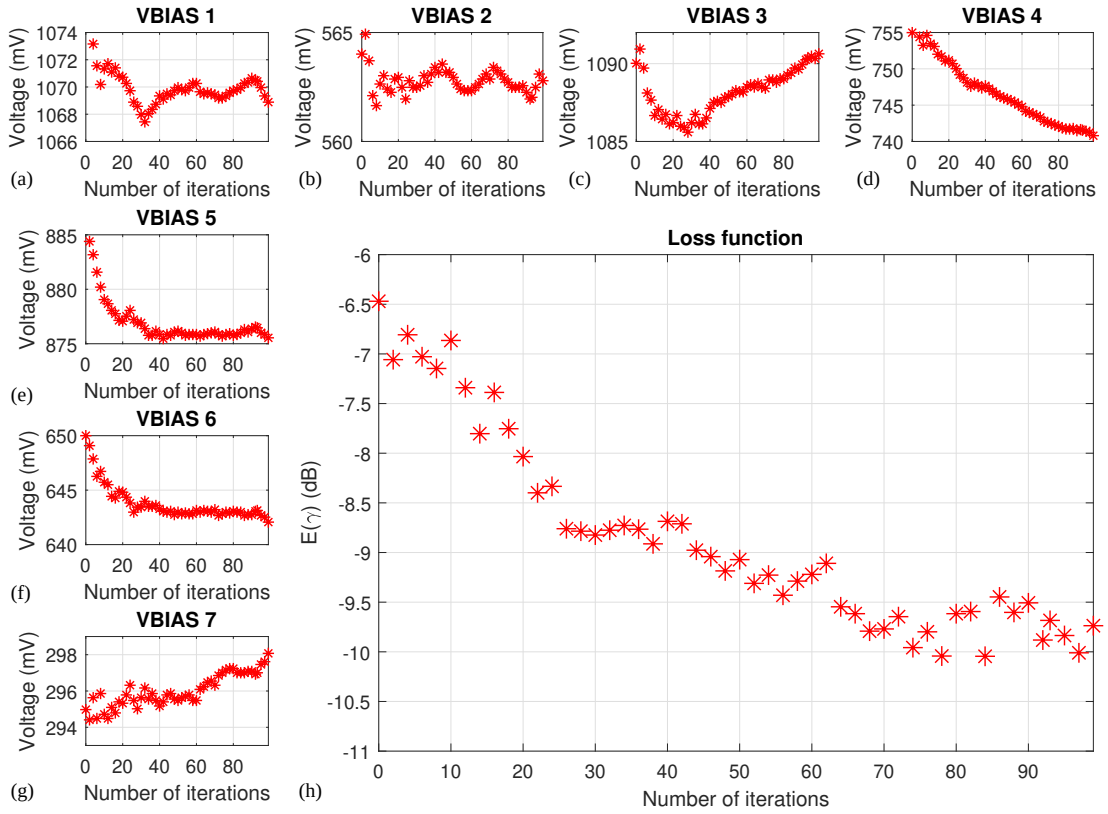

Figure 9.8: A series of calibration steps have been employed to improve the accuracy of the computation (expected value of noise-to-signal energy ratio). Here, the SPSA algorithm is used to find the optimized bias voltage values, while keeping the programmable gains constant. (a)-(g) The variation of the bias voltage with each iteration. (h) The loss function (expectation of the noise-to-signal energy ratio) reduces from -6dB to 10dB in 100 iterations.

As the second step, the bias voltages (there are a total of seven bias voltages) were optimized while keeping the gains and the group delays constant. Here, highly sensitive variables were fixed at the values that were determined in the previous stage. The bias voltages were supplied to the chip using an 8-channel 16-bit DAC (AD5669), which is controlled using a microcontroller (through an I2C interface). Since this is a 7-dimensional problem, the SPSA algorithm is utilized for the calibration. Here, the bias voltages were considered as continuously varying variables, even though they were programmed using a DAC. This assumption is valid, since the step size of the DAC is 0.0000763 V. The flow diagram of the calibration pro-

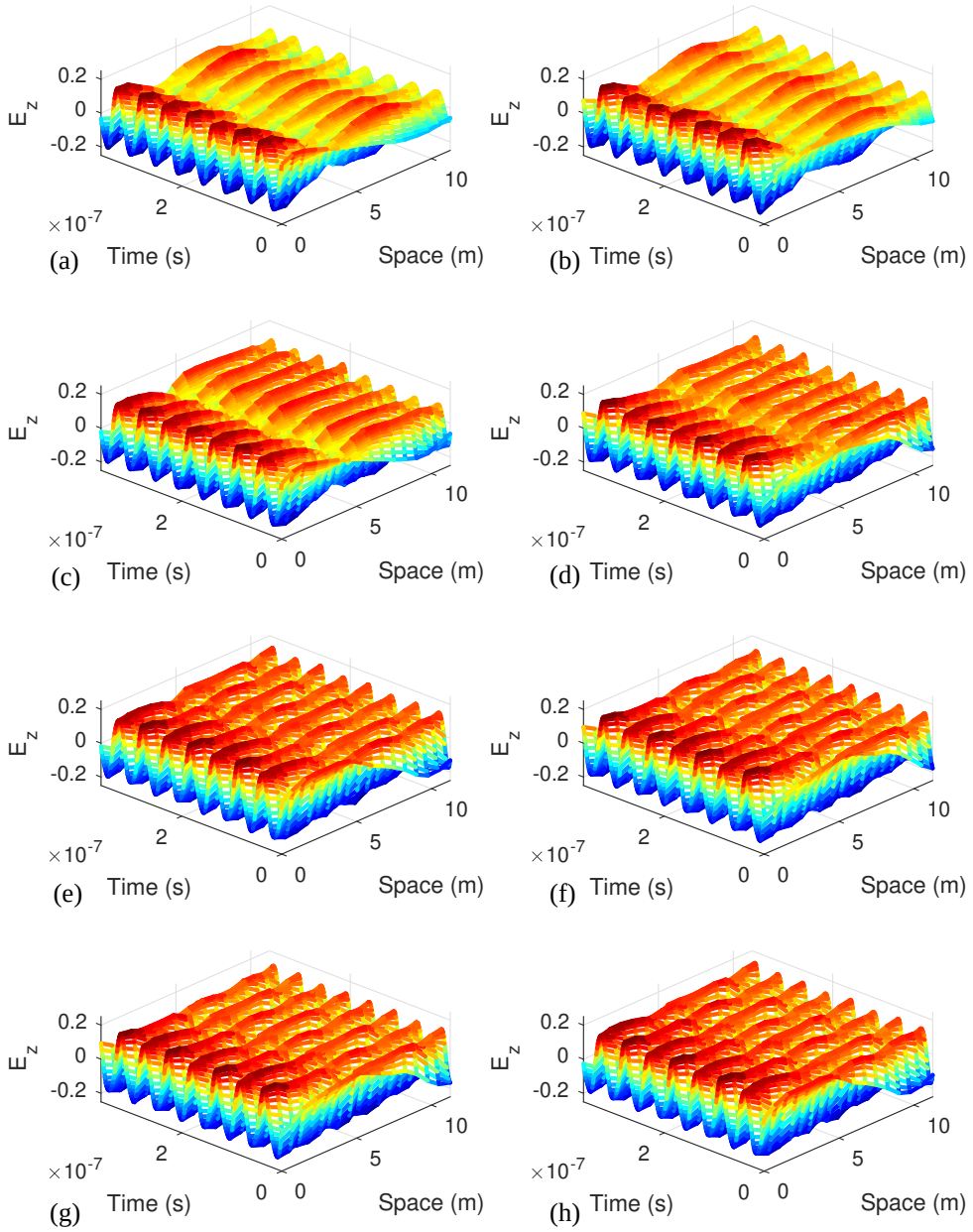

Figure 9.9: Measurement results obtained at several stages of the SPSA-based calibration process ((a)-(i) correspond to the iterations 1, 10, 20, 30, 40, 50, 60, and 70, respectively). In an absorbing boundary, the waves propagating in the medium should not have any reflection from the boundary. Also, the signal amplitude at each spatial point should be constant (over the spatial dimension).

cess that optimizes the bias voltages is shown in Fig. 9.8 (a). The process starts by setting the bias voltages at initial values $\hat{\theta}_0$ (note that $\hat{\theta}_0$ is a vector with seven voltage values). Here, the simulated bias voltages were used as the initial conditions. In each iteration k, the gradient of the loss function was approximated by taking two measurements of the objective function. Two measurements were taken by perturbing the bias voltages around the current values ($\hat{\theta}_k + c_k\Delta_k$ and $\hat{\theta}_k - c_k\Delta_k$), where $\hat{\theta}_k$ is the current set of bias voltages. Here, c_k is a small positive constant that becomes smaller as the iteration number becomes larger, and Δ_k is a 7-dimensional random vector with ± 1 (Bernoulli distribution with probability of 0.5). The two measurements were obtained by exciting the left boundary using a sinusoidal signal, capturing the computed solutions using ADCs, and comparing them with the FDTD solutions in MATLAB. The estimate of $\hat{\theta}_{k+1}$ was obtained using the standard stochastic approximation form $\hat{\theta}_{k+1} = \hat{\theta}_k - a_k\hat{g}_k\left(\hat{\theta}_k\right)$, where a_k and $\hat{g}_k\left(\hat{\theta}_k\right)$ are a positive constant and the approximated gradient of the loss function, respectively. Fig. 9.8 shows the results obtained as the bias voltages are being calibrated using the SPSA algorithm. Figs. 9.8 (a)-(g) show the variation of the bias voltages with each iteration. The process ran for 100 iterations. The loss function (expectation of the noise-to-signal energy ratio) reduces from -6dB to -10dB in 100 iterations (see Fig. 9.8 (h)). Fig. 9.9 shows how the measurements improve as the process progresses. In an absorbing (radiation) boundary, the waves propagating in the medium should not have any reflection from the boundary. Also, the signal amplitude at each spatial point should be constant (over the spatial dimension). Figs. 9.9 (a) - (h) correspond to the measurements captured at iterations 1, 10, 20, 30, 40, 50, 60, and 70, respectively.

As the next step, the variable gains of the IMs were calibrated. Here, the same gain values were assigned for the corresponding DACs at each module. Finally, the

gain values of the radiation boundary module were calibrated. The calibrated AC was then used to obtain the measurements for different boundary conditions.

CHAPTER 10

MEASUREMENTS, COMPARISON, AND SPEED-UP RESULTS

Following the calibration, the measurement setup has been utilized to capture the results from the AC for different wave propagation scenarios (different boundary conditions). The captured results were then compared with the MATLAB-based FDTD solver (the corresponding FDTD results are also shown). The metrics for the mean squared difference (MSD) and noise-to-signal energy ratio γ are utilized for the comparison. Two analog chips were tested to verify the consistency of the measurements. The power consumption, chip area, and the analog bandwidth of the AC were then compared with ACs that were reported in the literature. The ACs proposed in [8–11] were capable of solving PDEs using integrator-based signal flow graphs. However, their $F_{compute}$ for the CT operations were limited to 20 kHz–25 kHz. The reported $F_{compute}$ of our special-purpose AC is 30 MHz (with an equivalent update rate of 625 MHz).

The comparison metrics $F_{compute}$/Area and $F_{compute}$/power were then estimated for each AC (assuming they were solving the 1-D wave equation). These two metrics are good estimates on the achievable analog bandwidths of the ACs for a given area and a power constraint. The combination of the power and area requirements for a particular analog bandwidth was compared using $F_{compute}$/(area*power) metrics. The speed-up of our AC was estimated by comparing the computation times of FDTD solvers to run on a CPU (two Intel Xeon Silver 4110 CPUs @ 2.10 GHz with 8 cores and 256 GB of RAM). Two FDTD solvers were implemented on the CPU (based on MATLAB and C++). Furthermore, a digital FDTD solver was implemented on an FPGA (Xilinx RF system on chip xczu29dr-ffvf1760) to compare its computation speed with our AC.

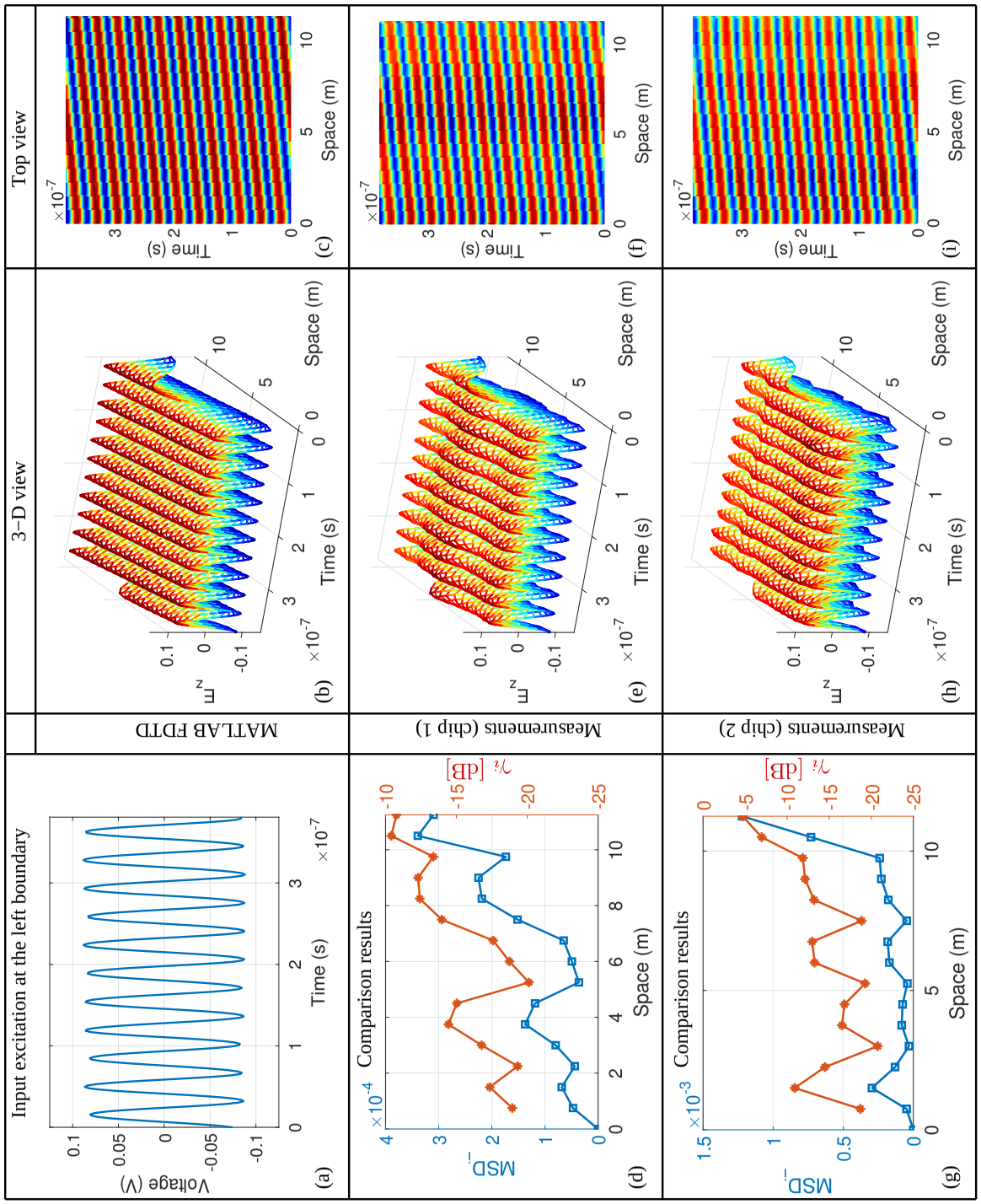

Figure 10.1: (a) The first module of the AC (left boundary) is excited using a sinusoidal pulse at 30 MHz. The right boundary is realized as a radiation boundary. (b-c) Simulation results obtained from the MATLAB FDTD solver for the same input. (e-f) Measurement results obtained from chip1. (h-i) Measurement results obtained from chip1. (d) and (g) the corresponding comparison of results with the MATLAB FDTD solver.

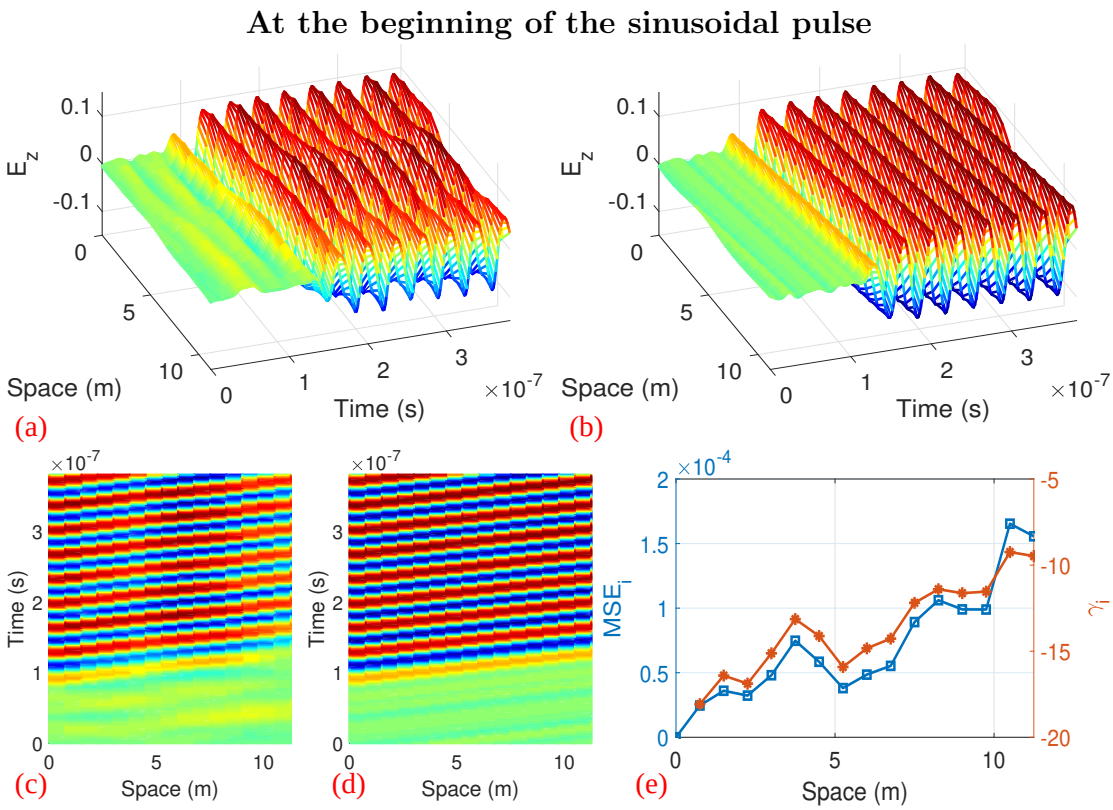

Figure 10.2: The left boundary (first module of the AC) is excited using a sinusoidal pulse at 30 MHz. The right boundary is realized as a radiation boundary. (a)-(c) Measurements obtained from the AC (which begins capturing data just before exiting the left boundary). (b)-(d) Simulation results obtained from the MATLAB FDTD solver. (e) Comparison of results with the MATLAB FDTD solver.

10.1 Measurements for the Analog Computer

Fig. 10.1 shows the measurement results obtained from the calibrated analog chips. Two chips were tested. The first module of the AC, which corresponds to the left boundary of the spatial grid, was excited using a 30 MHz sinusoidal signal (as shown in Fig. 10.1 (a)) representing an electric field of 30 MHz. The right boundary ($i = 17$) was programmed as a radiation boundary. The output voltages for all modules were captured, which correspond to the electric fields at each spatial point. Figs. 10.1(e-f) and (h-i) show the corresponding space-time variation (3-D and top views) for chip1 and chip2, respectively. The results from the MATLAB FDTD

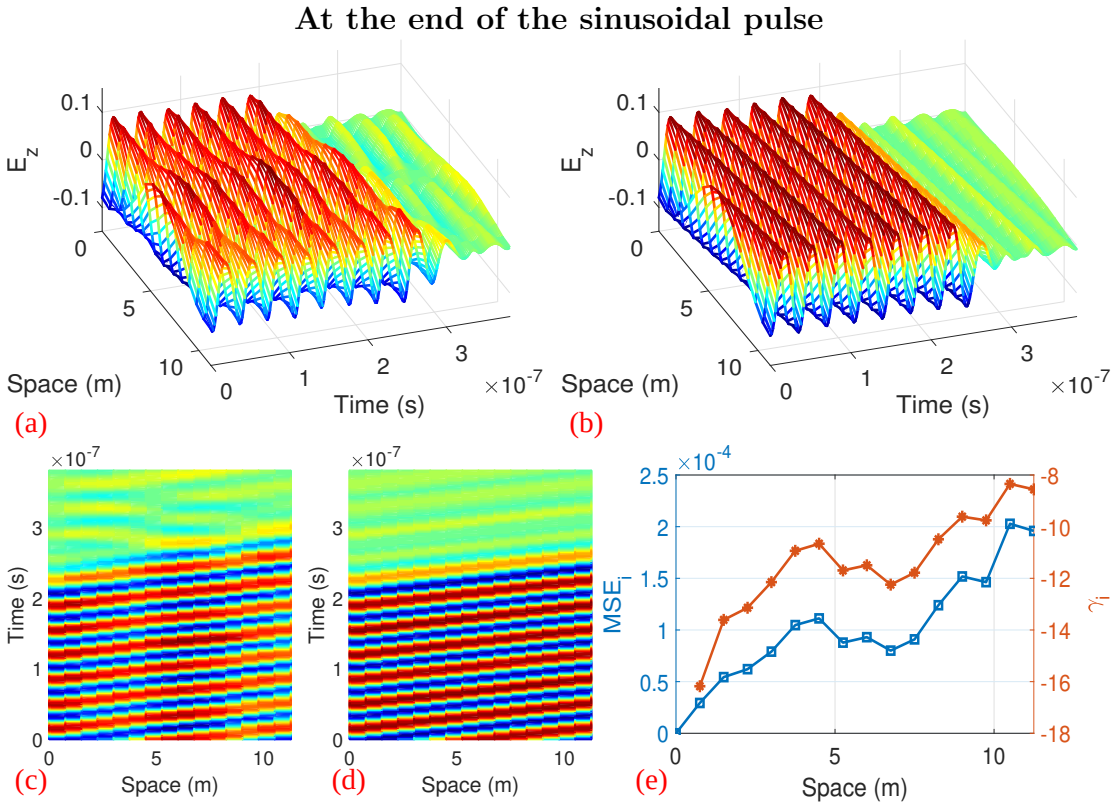

Figure 10.3: The left boundary is excited using a sinusoidal pulse at 30 MHz. The right boundary is realized as a radiation boundary. (a)-(c) Measurements obtained from the AC (which stopped capturing results just after removing the exitation signal). (b)-(d) Simulation results obtained from the MATLAB FDTD solver. (e) Comparison of results with those of the MATLAB FDTD solver.

simulations are shown in Figs. 10.1(b) and 10.1(c) for the same excitation input at the left boundary (and an absorbing boundary at the right edge of the spatial grid). In order to compare the captured solutions with the MATLAB-based FDTD solvers, the measurements at each spatial point were first interpolated with a time-step of 1.6 ns (the same as the FDTD time-step) using the MATLAB interp1 function and the spline interpolation method. The interpolated data were then compared with the FDTD solutions. Figs. 10.1(d) and 10.1(g) show the comparison between the two results (MSD and signal-to-noise ratio γ) for chip1 and chip2, respectively. Based on the measured results, the AC is capable of computing the solution to the wave

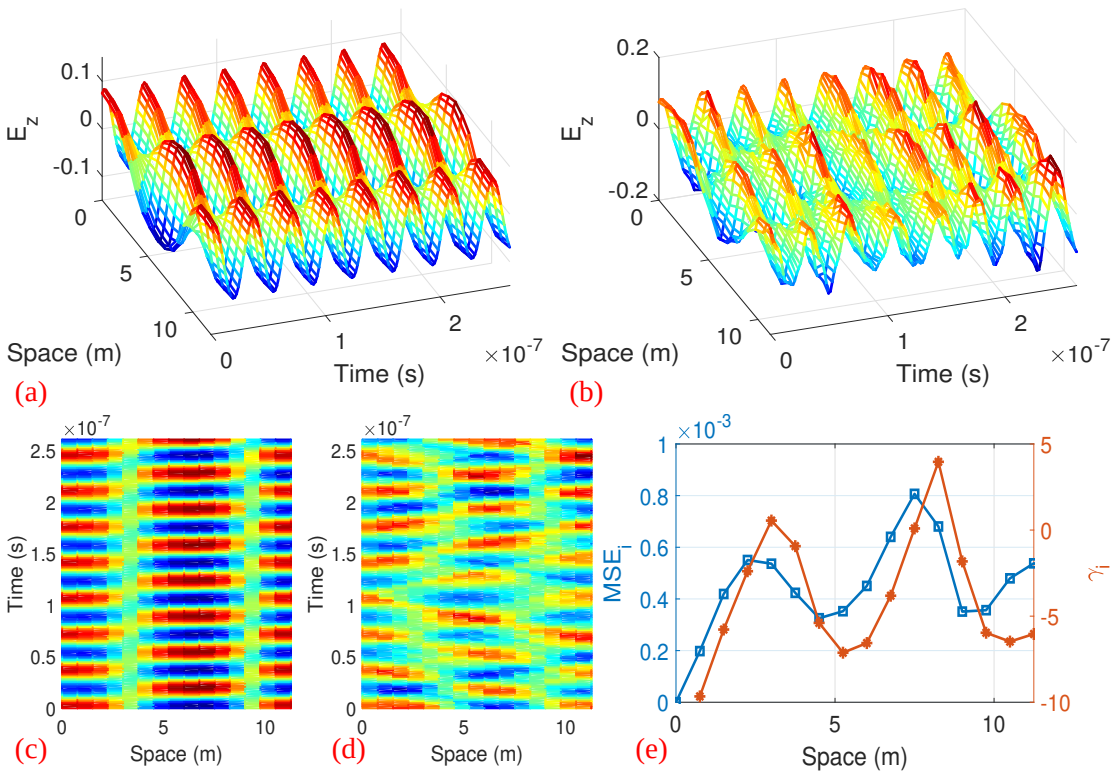

Figure 10.4: The left boundary is excited using a sinusoidal pulse at 30 MHz. The right boundary is realized as a Dirichlet boundary. (a, c) Measurements obtained from the analog solver. (b, d) Simulation results obtained from the MATLAB FDTD solver. (e) Comparison of results with those of the MATLAB FDTD solver.

equation with a difference less than -10 dB in terms of γ. The mean square error percentage of the AC is 1%–10% (depending on the spatial location).

Fig. 10.2 and Fig. 10.3 show the measurement results obtained at the beginning and the end of the excitation sinusoidal pulse, respectively. Here, only chip1 measurements are provided. In Fig. 10.2, we started the capturing process just before the boundary excitation was applied to the left boundary. Figs. 10.2(a) and (c) show the 3-D view and the top view of the measured results, respectively. Fig. 10.2(b) and Fig. 10.2(d) show the corresponding results for the MATLAB FDTD solver for the same excitation input. Fig. 10.2(e) compares the measured results to the simulation results in terms of MSD and γ, which also shows a 1%-10% mean squared

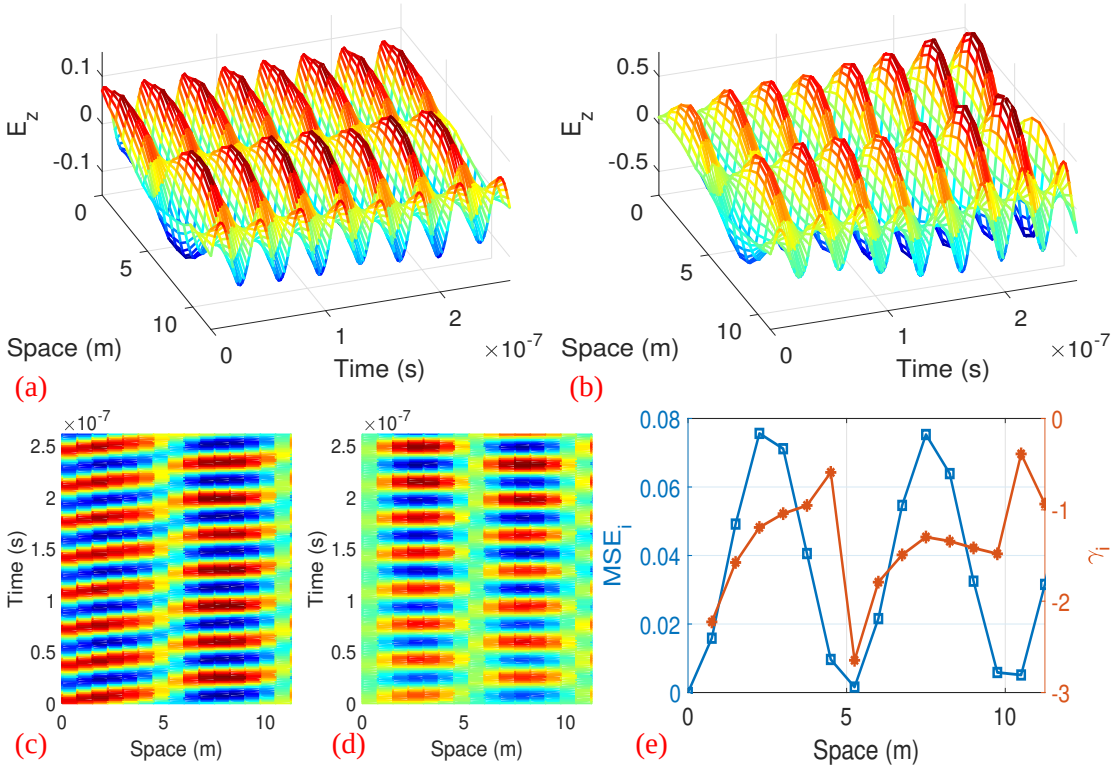

Figure 10.5: The left boundary is excited using a sinusoidal pulse at 30 MHz. The right boundary is realized as a Neumann boundary. (a, c) Measurements obtained from the analog solver. (b, d) Simulation results obtained from the MATLAB FDTD solver. (e) Comparison of results with those of the MATLAB FDTD solver.

error percentage as similar to the previous measurements. Fig. 10.3 shows the corresponding measurement results that were captured at the end of the boundary excitation signal. The accuracy of the CT computation is similar to the previous measurements.

Dirichlet Boundary: Fig. 10.4 shows the measurement and MATLAB simulation results obtained when the right boundary ($i = 17$) was programmed as a Dirichlet boundary (fixed boundary). Here, the left boundary was excited using a sinusoidal voltage signal at 30 MHz. Fig. 10.4(a) and Fig. 10.4(c) show the 3-D view and the top view of the measured results, respectively. Figs. 10.4(b) and (d) show the corresponding results from the MATLAB FDTD solver. Fig. 10.4(e) compares the

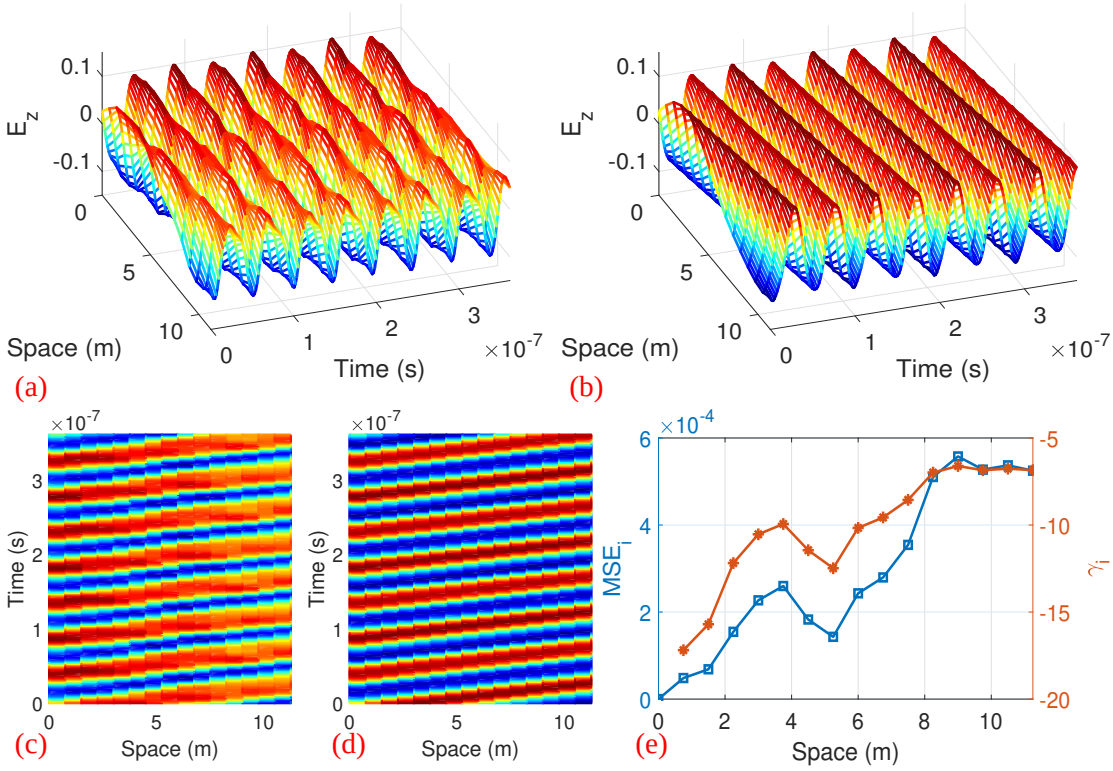

Figure 10.6: The left boundary is excited using a sinusoidal pulse at 20 MHz. The right boundary is realized as a radiation boundary. (a, c) Measurements obtained from the analog solver. (b, d) Simulation results obtained from the MATLAB FDTD solver. (e) Comparison of results with those of the MATLAB FDTD solver.

measured results to the simulation results in terms of MSE and γ. The accuracy of the computation can be improved by further calibrating the AC with the Dirichlet boundary.

Neumann Boundary: Fig. 10.5 shows the measurement and MATLAB simulation results obtained when the right boundary was programmed as a Neumann boundary (soft boundary). The left boundary was excited using the same sinusoidal signal at 30 MHz (representing an electric field at 30 MHz). Fig. 10.5(a) and Fig. 10.5(c) show the 3-D view and the top view of the measured results, respectively. Figs. 10.5(b) and (d) show the corresponding results for MATLAB FDTD solver. Fig. 10.5(e) compares the measured results versus the simulation results in terms of MSE and

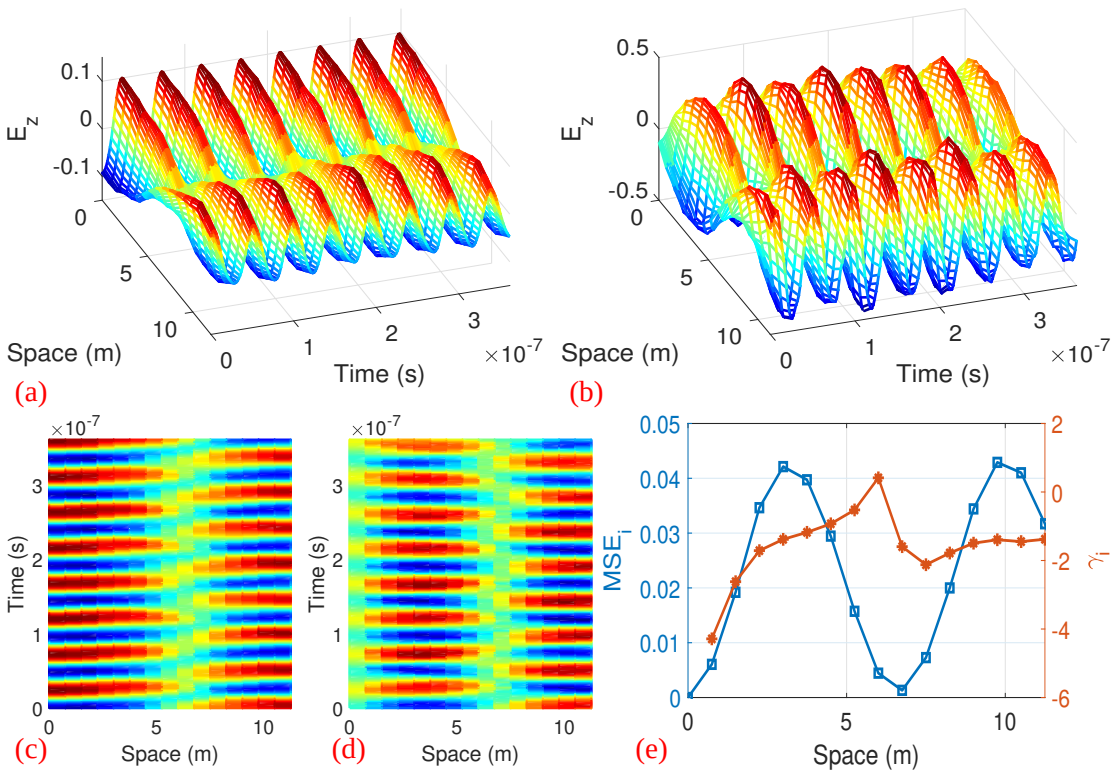

Figure 10.7: The left boundary is excited using a sinusoidal pulse at 20 MHz. The right boundary is realized as a Dirichlet boundary. (a) and (c), Measurements obtained from the analog solver. (b) and (d), Simulation results obtained from the MATLAB FDTD solver. (e) Comparison results with the MATLAB FDTD solver.

γ. The accuracy of the computation can be improved by further calibrating the AC with the Neumann boundary.

Radiation Boundary, 20 MHz Input: Fig. 10.6 shows the measurement and MATLAB simulation results obtained when the left boundary was excited using a sinusoidal voltage signal at 20 MHz (representing an electric field at 20 MHz). Here, the right boundary was realized as a radiation boundary. Figs. 10.6(a) and (c) show the 3-D view and the top view of the measured results, respectively. Figs. 10.6(b) and (d) show the corresponding results for MATLAB FDTD solver. Fig. 10.6(e) provides the comparison results (in terms of MSE and γ), which shows a 1%–20% mean squared error percentage.

Dirichlet Boundary, 20 MHz Input: Fig. 10.7 shows the measurement and MATLAB simulation results obtained when the right boundary was programmed as a Dirichlet boundary. The left boundary was excited using a sinusoidal voltage signal at 20 MHz. Figs. 10.7(a) and Fig. 10.7(c) show the 3-D view and the top view of the measured results, respectively. Figs. 10.7(b) and (d) show the corresponding results for MATLAB FDTD solver. Fig. 10.7(e) shows the comparison results.

10.2 Comparison with Results of Previous Work

With Moore's law slowing down, analog and analog-digital hybrid computing platforms realized using scaled CMOS are becoming attractive for complex computations and simulations [7, 12–18]. Columbia University's prototype analog accelerators are recent examples of energy-efficient hybrid computers that accelerate ODE- and PDE-based computations [7–11]. The reported ACs represent the variables using differential currents, whereas the proposed AC represents electric fields using voltages. The two ACs reported in [8–11] are capable of solving PDEs using integrator-based signal flow graphs (using integrators, multipliers/VGAs, and fanouts). The AC in [10, 11] contains 80 integrators, 80 multipliers/VGAs, and 160 fanouts such that it can be configured to solve different problems. The chip reported in [8, 9] is an improvement over the one reported in [10, 11], which consists of four integrators, eight multipliers/VGAs, eight fanouts with additional programmable circuits. Both these ACs are general-purpose AC that can be configured to solve different problems as required. In addition, they are capable of solving nonlinear systems of equations.

Table 10.1 shows the performance summary of our AC and the corresponding metrics of the previous works reported in [8–11]. Here, the ACs in [8, 9] and [10, 11] are denoted as AC_1 and AC_2, respectively. Our AC consists of 72 op-amps, 36 APFs,

Table 10.1: Performance comparison with the previous work report in [8–11] . Here, the ACs in [8, 9] and [10, 11] are denoted as AC_1 and AC_2, respectively.

	Our chip	AC_1 [8, 9]	AC_2 [10, 11]
Supply voltage	**1.8 V**	1.2 V	2.5 V
Technology	**TSMC 0.18 μm**	TSMC 65 nm	TSMC 0.25 μm
$F_{compute}$ (analog bandwidth)	**30 MHz**	20 kHz	25 kHz
$F_{max} = 1/\tau$ (equivalent update rate)	**625 MHz**	Not available	Not available
Die size	**4 mm^2**	3.8 mm^2	Not available
Active area (estimate: including circuits used for testing and programming)	**2 mm^2**	2 mm^2	100 mm^2
Power consumption	**200 mW**	1.2 mW	300 mW
Programming interface	**SPI**	SPI	Non-standard
Number of analog inputs	**2**	4	64
Number of analog outputs	**17**	4	64

and 36 OTAs (four op-amps, two APFs, and two OTAs per module). The 4mm^2 chip (including the pad frame) consumes 200 mW of power. The analog bandwidth $F_{compute}$ of our AC is 30 MHz, whereas the analog bandwidths of the AC_1 and AC_2 are 20 kHz and 25 kHz, respectively. Note that the PD compensation technique used in our AC is the key to achieve a higher bandwidth, which is challenging in integrator-based methods employed in [8–11]. The equivalent update rate of our AC is 625 MHz. Such a metric is not defined for AC_1 and AC_2. Even though the active area of our AC and AC_1 are the same (2 mm^2), the power consumption of AC_1 is only 1.2 mW (compared to the 200 mW of our AC). Low power consumption was achieved in AC_1 by biasing most of the transistors in the sub-threshold region, whereas most of the transistors in our AC was biased in the saturation region (except the OTAs). A standard SPI interface was utilized in AC_1 for programming (which is similar to our AC), whereas AC_2 was programmed using a non-standard interface.

Based on the results reported in [8–11], the power and area consumption of a single module in an analog wave equation solver has been estimated and compared with a single module of our AC. The corresponding module (in AC_1 and AC_2) requires two integrators, two fanouts, and one VGA. Table 10.2 shows the corresponding power and area results. The area of a single module of our AC is 0.09 mm^2 and it consumes 10.10 mW of power. The metrics $F_{compute}$/area, $F_{compute}$/power, and $F_{compute}$/(area*power) are utilized to compare the area and power results of the ACs. These metrics are good estimates on the achievable analog bandwidths of the ACs for a given area and a power constraint. The results show that our AC is 2560×, 24×, and 44× better in $F_{compute}$/area, $F_{compute}$/power, and $F_{compute}$/(area*power) metrics, respectively, when compared with the results in [8, 9]. The comparison results for AC_2 show 6660×, 290×, 1650× improvement in $F_{compute}$/area, $F_{compute}$/power, and $F_{compute}$/(area*power) metrics, respectively.

Table 10.2: The power and area consumption of a single module of the wave equation solver (when realized using our AC, AC_1, and AC_2). The performance metrics $F_{compute}$/Area, $F_{compute}$/Power, and $F_{compute}$/(Area*Power) are presented for comparison.

	Our chip	AC_1 [8, 9]	AC_2 [10, 11]
Active area for 1 module of the 1-D wave equation solver (estimate)	**0.09 mm^2**	0.16 mm^2 (1)	0.58 mm^2 (2)
$F_{compute}$/Area	**333 MHz/mm^2**	0.13 Hz/mm^2	0.05 MHz/mm^2
Power consumption for 1 module of the 1-D wave equation solver (estimate)	**10.10 mW**	0.17 mW (3)	2 mW (4)
$F_{compute}$/Power	**2.9 MHz/mW**	0.12 Hz/mW	0.01 MHz/mW
$F_{compute}$/(Area*Power)	**33 MHz/(mW.mm^2)**	0.74 MHz/(mW.mm^2)	0.02 MHz/(mW.mm^2)

(1) The results presented in Table II in [9] are used for the estimation.
(2) Fig. 3 in [10] and results presented in Table IV in [8] are used for the estimation.
(3) Results presented in Table III in [8] and Table II in [9] are used for the estimation.
(4) The results presented in Table IV in [8] are used for the estimation. The power consumption of the non-linear blocks in one macro in [10] is assumed as 40% (60% for all other blocks). The power consumption ratios of the integrator, VGA, and the fanout are assumed as same as the results presented in Table II in [9].

Table 10.3: Speedup comparison with MATLAB- and C-based FDTD solvers running or a CPU.

Input signal is 30 MHz sinusoidal signal with a sample period of 1.6 ns	**Our chip**	Computer with two Intel Xeon Silver 4110 CPU @ 2.10GHz, 8 Core(s), and 256 GB of RAM	
		FDTD MATLAB code	FDTD C code[1]
Time to simulate 1 ms of physical time	**1 ms**	99.1 ms	26.3 ms
Time to simulate 10 ms of physical time	**10 ms**	984 ms	261 ms
Time to simulate 100 ms of physical time	**100 ms**	10075 ms	2506 ms
Time to simulate 1000 ms of physical time	**1000 ms**	101589 ms	27009 ms
Average speedup		**100×**	**26×**

10.3 Estimated Speed-up of the Analog Computer

To compare speed-up against modern DCs, the execution times that the FDTD solvers take to simulate 1 ms, 10 ms, 100 ms, and 1000 ms of input waves were recorded when each AC/DC solver was excited using a 30 MHz sinusoidal signal with a time step of 1.6 ns. The radiation condition was configured at the right boundary. The FDTD solvers were implemented using MATLAB and C were executed on a computer that has two Intel Xeon Silver 4110 CPU @ 2.10GHz, 8 Core(s), resulting in 16 logical processors and is equipped with 256 GB of RAM. All the parameters were selected based on the analog solver (for a fair comparison). Corresponding results are shown in Table 10.3. The AC computes the solutions of the wave equation in real-time, whereas CPUs take many clock cycles to compute a single temporal frame of the update equation. The estimated speedup of the analog solver is about 100× and 26× compared to the MATLAB- and C-based FDTD solvers, respectively.

Furthermore, an FPGA-based FDTD solver was implemented on a Xilinx RF system on chip (SoC) (xczu29dr-ffvf1760) using the ZCU1275 board. The digital

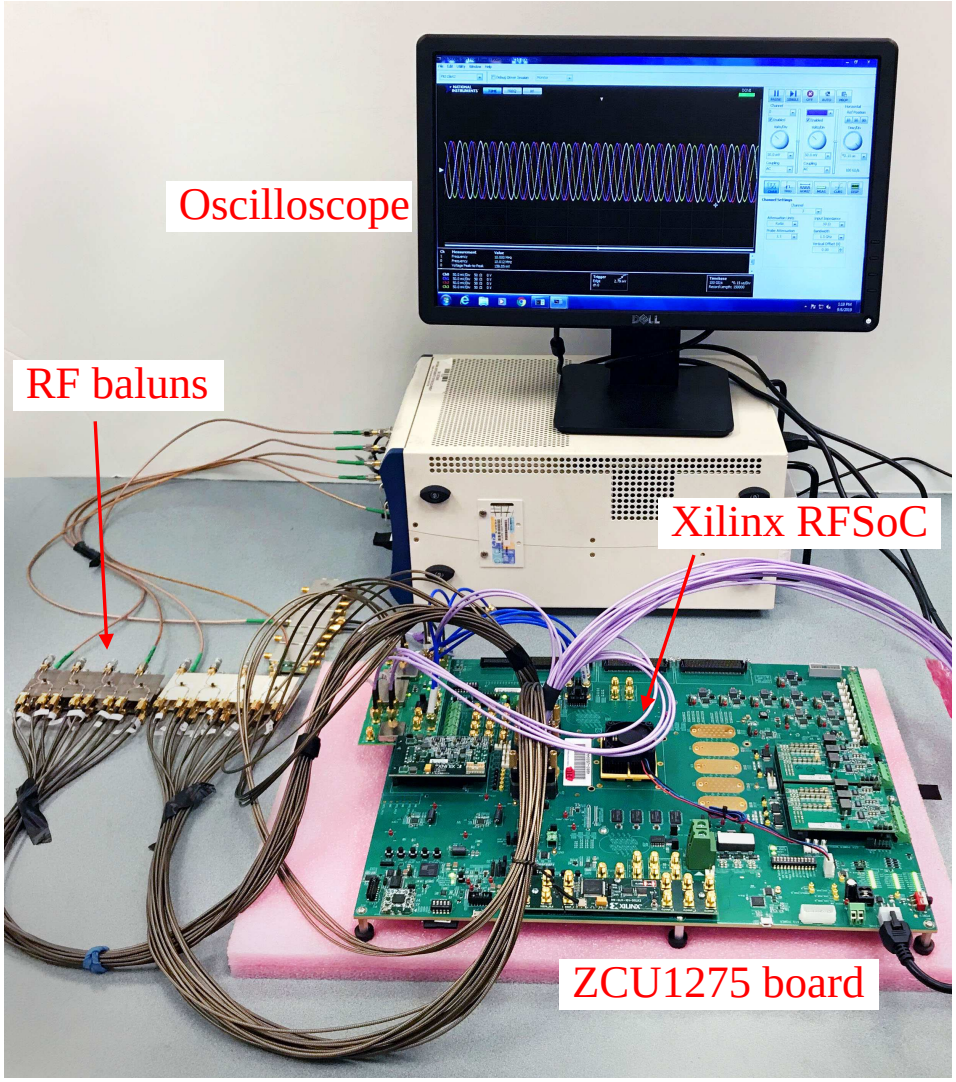

Figure 10.8: A digital hardware FDTD solver is implemented on Xilinx RFSoC (xczu29dr-ffvf1760) using the ZCU1275 board (clocked at a maximum frequency of 222 MHz). The left boundary is excited using a signal generator which is then digitized using an ADC. All the outputs are converted to the analog domain using the 16-channel DAC. Four analog outputs are captured using the oscilloscope.

design was implemented in MATLAB Simulink using the Xilinx system generator. The FDTD solver was then integrated with ADCs to excite the system using analog inputs and DACs to capture the outputs on an oscilloscope. The Vivado IP integrator was used to design the complete system and generate the bitstream to program the RFSoC. The maximum frequency of the design is 222 MHz. The dynamic power

Table 10.4: Speedup comparison with FPGA-based FDTD solvers running on RF-SoC (ZCU1275 evaluation board)

Input signal is 30 MHz sinusoidal signal with a sample period of 1.6 ns	Our chip	FDTD FPGA design running on xczu29dr-ffvf1760 RFSoC @222 MHz
Time to simulate 1 ms of physical time	1 ms	2.8 ms
Time to simulate 10 ms of physical time	10 ms	28 ms
Time to simulate 100 ms of physical time	100 ms	280 ms
Time to simulate 1000 ms of physical time	1000 ms	2800 ms
Average speedup		2.8×
$F_{compute}$/Power	**0.15 MHz/mW**	0.01 MHz/mW

consumption is about 900 mW (excluding ADC and DACs). Fig. 10.8 shows the experiment setup of the FPGA-based FDTD solver. The left boundary of the spatial grid was excited using an analog signal generated from a signal generator via ADC. The computed 16 outputs were taken back into the analog domain using a 16-channel DAC running at 800 MHz. Four outputs are shown in the oscilloscope. Similar to the CPU-based FDTD solvers, the times taken to simulate 1 ms, 10 ms, 100 ms, and 1000 ms of the physical time of a 30 MHz signal were recorded. The corresponding results are shown in Table 10.4. The speedup of the 180 nm CMOS AC is 2.8× at 200 mW. Based on the power results, our AC is 15 times better in terms of the $F_{compute}$/Power metric, when compared to the FPGA-based FDTD solver running on the RFSoC.

CHAPTER 11

CONCLUSION AND FUTURE WORK

In this dissertation, we proposed analog computing techniques to compute the approximate solutions of linear and nonlinear PDEs. The main objective was to perform the computations at a speed that is 1 to 2 orders of magnitude greater than the available ACs and DCs. The proposed computing techniques were first verified by designing ideal ACs for solving Maxwell's and wave equations. They were then simulated in Cadence, and the accuracy of the computation was compared with standard FDTD solvers. CMOS-based ACs were then designed and simulated by addressing the non-ideal effects of the CMOS-based circuits. Finally, an energy-efficient AC that approximately solves the wave equation was designed, simulated, and fabricated using 180 nm CMOS technology. Recent examples of such energy-efficient approximate computing reported in [8–11] are accurate but have less than 25 kHz of analog bandwidth for CT operations. Our special-purpose AC has an analog bandwidth of 30 MHz (with an equivalent update rate of 625 MHz) at a power consumption of 200 mW. The power and area consumption of our AC was compared with previous work. The acceleration of the AC is 1 to 3 orders of magnitude greater than the existing ACs for a given area and power budget if they compute the CT solution of the wave equation. Furthermore, our AC is 100×, 26× faster when compared to the MATLAB- and C-based FDTD solvers running on a CPU (at 2.10 GHz) with 8 cores and 256 GB of RAM, respectively. An FPGA-based FDTD solver running on an RFSoC consumes 900 mW of power. The corresponding speed-up of the AC is 2.8× at 200 mW.

This dissertation explored several paths towards building analog accelerators for solving linear and nonlinear PDEs. The following section summarizes the major technical accomplishments of this dissertation.

11.1 Technical Accomplishments

Following is a list of major achievements from the dissertation.

Accomplishment 1: Two methods have been introduced to map a given PDE into analog circuits, which can eventually compute the CT solution of the PDE. Both methods are discrete in space but continuous in time such that they can be implemented using analog circuits.

i) Continuous-time in Laplace domain (CTLD) method: In this method, the partial derivatives in the spatial dimension are approximated using discrete finite differences, while the LT is applied to partial derivatives in the time dimension [29–31, 106, 107]. The resulting mixed-domain update equation is used to design an analog module that can compute the solution for a given spatial point. The resulting analog computing modules are then interconnected in a systolic array architecture to compute the solution over the entire spatial grid.

ii) All-pass delay approximation (APDA) method: This method replaces the discrete-time difference operators in the standard FDTD cell (Yee cell), using a CT delay operator, which can be realized as an analog APF [110–115]. The summing and scaling operations in the Yee cell are realized using op-amp-based analog circuits. Individual cells can then be interconnected in a systolic array to compute the complete solution.

Accomplishment 2: Maxwell's equations and the wave equation, which are extremely important in electromagnetics and have numerous applications in modern communication systems, were used to verify the proposed methods. ACs for solving 1-D Maxwell's equations and the 1-D and 2-D wave equations were designed and simulated using ideal models in Cadence integrated circuit design software. Both CTLD and APDA methods were considered. Different boundary conditions were

simulated, which include Dirichlet boundary, Neumann boundary, and radiation boundary conditions. Cadence-based results were provided for different simulation scenarios based on ideal models. Furthermore, the behavior of the wave equation in two different mediums was analyzed. In addition, a mathematical model for implementing the damped wave equation in a CT system was introduced by approximating the losses by a first-order partial derivative term.

Accomplishment 3: Methods and techniques have been investigated towards expanding the proposed analog computing systems into multiple dimensions and larger computational domains. The need for scalability was addressed by taking a highly modular and regular approach to circuit realization. By drawing analogies to digital systolic arrays, the proposed analog accelerators formed a highly modular, regular and locally interconnected fabric of analog computational accelerator cores, which can be scaled up in size to take into account larger computational domains as required by the PDE system being simulated.

Accomplishment 4: In order to quantify how much the CT solution deviates from the standard FDTD solution, two metrics have been calculated: the mean squared difference (MSD) between the two solutions and the noise energy to signal energy ratio γ (in dB). Here, noise refers to the deviation between the two solutions. These metrics describe how much the computed CT solution deviates with respect to the FDTD solution, when the proposed solvers are realized using analog circuits. For this purpose, an FDTD-based Maxwell's equation solver and a wave equation solver were implemented using MATLAB.

Accomplishment 5: By exploiting the APDA method, a CT mathematical model for solving non-linear coupled PDEs was proposed. Here, conservative systems (an important class of systems with various real-world applications), which are governed by non-linear coupled PDEs, were considered for designing the AC. The investigation was started with an existing FDTD numerical method, which can solve coupled non-linear PDEs [32, 117]. The proposed method replaces the discrete-time delays in the FDTD cell using a CT delay (an analog all-pass filter), thereby obtaining the corresponding spatially discrete but time-continuous (SDTC) version of the PDE solver [110–115]. The remaining operators such as addition, subtraction, multiplication, and amplification were designed using standard analog circuits as separate building blocks.

Accomplishment 6: A low-frequency prototype of an AC that solves the 1-D wave equation has been designed and simulated using discrete op-amp ICs in the OrCAD analog circuit design software. The Bruton transformation was applied to avoid the use of inductors [31, 106, 107, 129], which introduces a circuit element called frequency-dependent negative resistance (FDNR) to the circuit. The FDNR element was realized using active circuits (op-amps) [31, 106, 107, 129]. The OrCAD models were then extended into a PCB level implementation. Eight- and sixteen-spatial point ACs were implemented and tested. The measurement results were captured to demonstrate the wave propagation in the space-time domain.

Accomplishment 7: The key challenges toward CMOS implementations of the proposed ACs were identified. The main challenge of realizing such large-scale analog networks is to minimize the effects of circuit non-idealities such as noise, nonlinearity, component mismatch, and parasitic elements (mostly device and

interconnect capacitances) on the computation. As an example, practical op-amps have non-ideal characteristics including finite gain-bandwidth, voltage offsets, input bias currents noise and drift, finite input and output impedances, finite common-mode and power-supply rejection, and limited linear range. Each of these non-idealities affects the analog solutions with a certain sensitivity that needs to be analyzed prior to a CMOS implementation. These analysis results can then be used to determine the required performance metrics of the op-amps and other circuits. Furthermore, a PD compensation technique was proposed for the APDA-based ACs to compensate for the effect of finite bandwidth circuits (which is inevitable for all practical elements).

Accomplishment 8: Both CTLD- and APDA-based ACs were extended to analog CMOS implementations. They were designed and simulated employing the TSMC 180 nm CMOS technology for solving Maxwell's and wave equations. Low-voltage (operating voltage of 1.8 V) MOS transistor models were used. Cadence integrated circuit design software is used to design the system. As the main building block (in both methods), a wide swing cascode op-amp with a 90 dB of gain and a 500 MHz of gain-bandwidth product was designed. An analog APF was designed using a cascade of a resistor-capacitor network for the APDA method. The proposed CMOS ACs were simulated in Cadence and simulated for different wave/field propagation scenarios. Full-scale BSMI3 models that take into account full process parameters were used in the design and simulations.

Of all the ACs considered, the ADPA-based AC that solves the 1-D wave equation was extended into a chip-level implementation with an expected analog bandwidth of 50 MHz. The analog solver consists of 18 analog modules. The

outputs of each module produce a set of time-varying voltages that correspond to the electric fields at each spatial point (defined by the wave equation). Precision circuit design techniques had been utilized (e.g., capacitor ratios to control the gains of op-amps) to maximize the linear range and also to minimize the systematic mismatches of the analog circuits, which eventually reduce the effects from element parasitics and PVT variations. Furthermore, precision layout techniques were employed to minimize systematic and random mismatches (e.g., common centroid layouts). The effects of parasitic elements were estimated using post-layout extraction and back-annotation to adjust the design as necessary. An SPI module was designed to program and calibrate the entire chip. The total chip size of the analog solver is 2 mm $\times$ 2 mm (including the pad-frame).

Accomplishment 9: A prototype of an analog-digital hybrid computational platform was designed and implemented using the analog chip, FPGAs, ADCs, DACs, and microcontrollers so that the analog chip can be easily programmed, calibrated, tested and and evaluated to verify their functionalities. Input excitations and boundary conditions were generated inside an FPGA and supplied to the analog chip through a DAC. Digital architectures were designed to generate required boundary excitations and to capture computed analog waveforms. These architectures were designed within the MATLAB environment using the Xilinx system generator. The designs were targeted to be realized on the ROACH-2 FPGA platform. All computations were performed in the analog chip that inputs and outputs signals via 50 Ω transmission lines. Computed analog solutions were routed back into the FPGA through a 16 channel ADC. Reconfiguration and calibration SPI commands were sent by an Atmel microcontroller.

Accomplishment 10: The designed hybrid test bench was utilized to collect the measurements of the AC that solves the 1-D wave equation. Before taking the measurements, a series of calibration steps were carried out to improve the accuracy of the computation (to calibrate the programmable gains and bias voltages). The SPSA algorithm was used as the optimization technique [135, 136]. Measurements of the AC were then captured for different boundary conditions and frequencies. The reported analog bandwidth of the AC is 30 MHz with an equivalent update rate of 625 MHz. The measurement results were then compared with MATLAB FDTD solutions. The mean square error percentage of the AC is 1%-10%.

Accomplishment 11: The speed, area, and power results of the AC were compared with modern ACs reported in [8–11]. The speed-up of our AC is 1 to 3 orders of magnitude greater than the available ACs. For a given power and area budget, the analog bandwidth of the ACs is also 1 to 3 orders of magnitude higher than the existing ACs. To compare speed-up against modern DCs, a reference FDTD solver is implemented using a C-code and is executed on a computer that has two Intel Xeon Silver 4110 CPUs (at 2.10 GHz) with 8 cores and 256 GB of RAM. The estimated speedup of the AC is 26× compared to the C-code. Furthermore, an FPGA-based FDTD solver was implemented on a Xilinx RF system on chip (SoC) (xczu29dr-ffvf1760) using the ZCU1275 board and clocked at a maximum frequency of 222 MHz at about 900 mW of dynamic power. The corresponding speedup of the 180 nm CMOS AC is 2.8× at 200 mW.

The outcomes from this dissertation provide a promising pathway to explore future analog computing accelerators.

11.2 Future Work

The followings are some suggestions to explore based on the outcomes and findings from this dissertation.

Analog Computer for Solving Nonlinear PDEs: Mathematical models that were proposed in Chapter 6 to solve nonlinear PDEs can be extended towards an analog CMOS implementation. The proposed AC only requires adder/subtractors, buffers, programmable gain amplifiers, APFs, and multipliers, in analog-domain, as building blocks towards implementing a fully-integrated analog chip. An example system such as the acoustic shock tube problem discussed in Subsection 6.4.1 can be employed to design, simulate and fabricate the AC. Similar to the wave equation–solving AC, the PDs of each element can be compensated to achieve larger bandwidths for the CT computations. However, the bandwidth of the multiplier would mainly decide the analog bandwidth of the AC.

A Current-mode Analog Computer: The proposed ACs in this dissertation are voltage-mode circuits (in our ACs, the dependent variables in PDEs are represented using voltages). As an example, the electric field values in the wave equation solving AC were represented using voltages. Each module in the AC produces a time-varying voltage that corresponds to the electric field at the particular spatial point (each module corresponds to a particular spatial point in the spatial grid). In a current-mode AC, we can represent variables using currents in the analog circuit instead of the voltages. The same mathematical models that we developed in this dissertation can be used to design the current-mode AC. Here, the summation operation can be realized by simply connecting multiple wires together. Also, the summation operation performs instantly (no PD). In contrast, the voltage-mode

circuit requires a summing circuit that has a finite PD. This is one advantage of current-mode circuits. However, the current-mode operation requires fanout blocks (a current mirror) to buffer the outputs prior to feeding the next operation. The scaling and multiplication operations can be implemented as proposed in [8–11]. The power consumption of the current-mode AC can be reduced by biasing the transistors in the sub-threshold region instead of saturation. In a voltage-mode circuit, most of the transistors are biased in saturation, which leads to high power consumption. However, it should be noted that the accuracy of sub-threshold current mirrors is greatly affected by PVT variations.

A General-purpose AC for Solving FDTD Computations: The ACs implemented in this dissertation are special-purpose computers that can only solve one computational problem. However, the proposed analog computing methods (CTLD and APDA methods) are capable of solving any given linear PDE. Also, the APDA method is capable of solving nonlinear PDEs as well, as long as there is a numerical method available for solving them. Thus, a general update equation can be obtained by applying the proposed methods on a general form of a PDE as similar to Chapter 3. The resulting SDTC update equation can be used to design the corresponding AC. The AC can be configured (using software inputs) before solving any PDE such that it models the given PDE. The APDA method is a better candidate for a general-purpose AC as compared to the CTLD method, since the APDA method is based on standard FDTD algorithms. An APDA-based AC only requires adder/subtractors, buffers, programmable gain amplifiers, and APFs as building blocks (multipliers are required for solving nonlinear PDEs). Both voltage- and current-mode circuits are capable of implementing a general-purpose AC with appropriate programming modules.

BIBLIOGRAPHY

[1] DataGrapha. (2019) Moore's law graphed vs real CPUs & GPUs 1965 - 2019. [Online] Available: $https://www.youtube.com/watch?v=7uvUiq\_jTLM$

[2] V. Bush, "The differential analyzer. A new machine for solving differential equations," *Journal of the Franklin Institute*, vol. 212, no. 4, pp. 447 – 488, 1931.

[3] J. S. Small, *The analogue alternative: The electronic analogue computer in Britain and the USA, 1930-1975.* Routledge, 2013.

[4] K. H. Lundberg, "The history of analog computing: introduction to the special section," *IEEE Control Systems*, vol. 25, no. 3, pp. 22–25, June 2005.

[5] C.-H. Jan and et al, "RF CMOS technology scaling in High-k/metal gate era for RF SoC (system-on-chip) applications," in *IEEE International Electron Devices Meeting (IEDM)*, Dec 2010, pp. 27.2.1–27.2.4.

[6] H. Shimomura, "A study on high-frequency performance in MOSFETs scaling," *Tokyo Inst. Technol., Tokyo, Japan, Tech. Rep*, pp. 1–167, 2011.

[7] Y. Tsividis, "Not your father's analog computer," *IEEE Spectrum*, vol. 55, no. 2, pp. 38–43, February 2018.

[8] N. Guo, Y. Huang, T. Mai, S. Patil, C. Cao, M. Seok, S. Sethumadhavan, and Y. Tsividis, "Energy-efficient hybrid analog/digital approximate computation in continuous time," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 7, pp. 1514–1524, 2016.

[9] Y. Huang, N. Guo, M. Seok, Y. Tsividis, and S. Sethumadhavan, "Evaluation of an analog accelerator for linear algebra," in *2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA)*, June 2016, pp. 570–582.

[10] G. E. R. Cowan, R. C. Melville, and Y. P. Tsividis, "A VLSI analog computer/digital computer accelerator," *IEEE Journal of Solid-State Circuits*, vol. 41, no. 1, pp. 42–53, Jan 2006.

[11] G. E. R. Cowan, R. C. Melville, and Y. P. Tsividis, "A VLSI analog computer/math co-processor for a digital computer," in *IEEE International Solid-*

State Circuits Conference, Digest of Technical Papers. IEEE, 2005, pp. 82–586.

[12] Y. Huang, N. Guo, M. Seok, Y. Tsividis, and S. Sethumadhavan, "Analog computing in a modern context: A linear algebra accelerator case study," *IEEE Micro*, vol. 37, no. 3, pp. 30–38, 2017.

[13] S. Fifer, *Analogue computation: theory, techniques, and applications.* McGraw-Hill, 1961, vol. 4.

[14] N. Guo, Y. Huang, T. Mai, S. Patil, C. Cao, M. Seok, S. Sethumadhavan, and Y. Tsividis, "Continuous-time hybrid computation with programmable nonlinearities," in *41st European Solid-State Circuits Conference (ESSCIRC)*, Sept 2015, pp. 279–282.

[15] K. accelerating intelligence, "Could analog computing accelerate complex computer simulations?" 2015. [Online] Available: http://www.kurzweilai.net/could-analog-computing-accelerate-complex-computer-simulations

[16] Y. Huang, N. Guo, M. Seok, Y. Tsividis, K. Mandli, and S. Sethumadhavan, "Hybrid analog-digital solution of nonlinear partial differential equations," in *Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture.* ACM, 2017, pp. 665–678.

[17] S. Kim, J. Hasler, and S. George, "Integrated floating-gate programming environment for system-level ICs," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 6, pp. 2244–2252, June 2016.

[18] S. George, S. Kim, S. Shah, J. Hasler, M. Collins, F. Adil, R. Wunderlich, S. Nease, and S. Ramakrishnan, "A programmable and configurable mixed-mode FPAA SoC," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 24, no. 6, pp. 2253–2261, June 2016.

[19] S. Siddiqui, "Why are differential equations used for expressing the laws of physics?" *arXiv preprint arXiv:1406.1112*, 2014.

[20] M. Sadiku, *Elements of electromagnetics.* Oxford university press, 2014. [Online] Available: http://www.ebook.de/de/product/21893771/matthew_sadiku_elements_of_electromagnetics.html

[21] J. D. Anderson and J. Wendt, *Computational fluid dynamics*. Springer, 1995, vol. 206.

[22] G. K. Batchelor, *An introduction to fluid dynamics*. Cambridge university press, 2000.

[23] W. Chew, "150 years of maxwell's equations: A reflection," in *USNC-URSI Radio Science Meeting (Joint with AP-S Symposium)*. IEEE, 2014, pp. 288–288.

[24] M. D. Petkovic and P. S. Stanimirovic, "Generalized matrix inversion is not harder than matrix multiplication," *Journal of Computational and Applied Mathematics*, vol. 230, no. 1, pp. 270 – 282, 2009. [Online] Available: http://www.sciencedirect.com/science/article/pii/S0377042708006237

[25] V. Pan, "New fast algorithms for matrix operations," *SIAM Journal on Computing*, vol. 9, no. 2, pp. 321–342, 1980. [Online] Available: https://doi.org/10.1137/0209027

[26] Hybrid computer (wikipedia the free encyclopedia). [Online] Available: https://en.wikipedia.org/wiki/Hybrid_computer

[27] A. E. Rohers. (1963) Hybrid computation: What and why. [Online] Available: https://archive.org/details/bitsavers_computersA_6121266/page/n9

[28] N. Udayanga, S. I. Hariharan, S. Mandal, L. Belostotski, L. T. Bruton, and A. Madanayake, "Continuous-time algorithms for solving maxwells equations using analog circuits," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 10, pp. 3941–3954, Oct 2019.

[29] M. Maini, "Analog 2D spatio-temporal recursive filters for plane wave array processing," 2006. [Online] Available: http://hdl.handle.net/1880/102598

[30] L. T. Bruton, "2D/3D analog mixed-domain beam/fan filters," Internal Rep, University of Calgary, AB, Canada, 2003.

[31] L. T. Bruton, A. Madanayake, C. Wijenayake, and M. Maini, "Continuous-time analog two-dimensional IIR beam filters," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 59, pp. 419 –423, July 2012.

[32] U. S. Inan and R. A. Marshall, *Numerical Electromagnetics: The FDTD Method*. Cambridge University Press, 2011.

[33] Acceleware, "Finite-difference time-domain solver and SDK." [Online] Available: http://www.acceleware.com/fdtd-solvers.

[34] K. S. Kunz and R. J. Luebbers, *The finite difference time domain method for electromagnetics*. CRC press, 1993.

[35] M. Andersson, R. Sigurdsson, and M. Passare, "Analytic solutions to partial differential equations," in *Complex Convexity and Analytic Functionals*. Springer, 2004, pp. 129–150.

[36] J. J. Sakurai and J. Napolitano, *Modern quantum mechanics*. Cambridge University Press, 2017.

[37] J. D. Logan, *The Physical Origins of Partial Differential Equations*. New York, NY: Springer US, 1998, pp. 1–49. [Online] Available: https://doi.org/10.1007/978-1-4684-0533-0_1

[38] M. F. Iskander, *Electromagnetic fields and waves*. Waveland Press, 2013.

[39] J. D. Callen, C. C. Hegna, and A. J. Cole, "Transport equations in tokamak plasmas," *Physics of Plasmas*, vol. 17, no. 5, pp. –, 2010.

[40] A. Taflove and S. C. Hagness, *Computational electrodynamics: the finite-difference time-domain method*. Artech house, 2005.

[41] K. Yee, "Numerical solution of initial boundary value problems involving maxwell's equations in isotropic media," *IEEE Transactions on Antennas and Propagation*, vol. 14, no. 3, pp. 302–307, May 1966.

[42] G. Mur, "Absorbing boundary conditions for the finite-difference approximation of the time-domain electromagnetic-field equations," *IEEE transactions on Electromagnetic Compatibility*, no. 4, pp. 377–382, 1981.

[43] W. F. Ames, *Nonlinear partial differential equations in engineering*. Academic press, 1965, vol. 18.

[44] A. Sawitzki, "Electromagnetic modelling of surfaces using method of moments with calculated phase mesh," *IET Microwaves, Antennas Propagation*, vol. 9, no. 12, pp. 1354–1362, 2015.

[45] H. Lee, J. Chai, and S. Patnakar, "Finite volume method for radiation heat transfer," *Journal of thermophysics and heat transfer*, vol. 8, pp. 419–425, 1994.

[46] R. J. LeVeque, *Finite volume methods for hyperbolic problems*. Cambridge university press, 2002, vol. 31.

[47] S. C. Brenner and C. Carstensen, "Finite element methods," *Encyclopedia of computational mechanics*, 2004.

[48] G. Strang and G. J. Fix, *An analysis of the finite element method*. Prentice-hall Englewood Cliffs, NJ, 1973, vol. 212.

[49] C. Canuto, M. Y. Hussaini, A. Quarteroni, A. Thomas Jr *et al.*, *Spectral methods in fluid dynamics*. Springer Science & Business Media, 2012.

[50] J. S. Hesthaven, S. Gottlieb, and D. Gottlieb, *Spectral methods for time-dependent problems*. Cambridge University Press, 2007, vol. 21.

[51] E. Hairer, C. Lubich, and G. Wanner, *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*. Springer Science & Business Media, 2006, vol. 31.

[52] J. Sanz-Serna, "Runge-kutta schemes for hamiltonian systems," *BIT Numerical Mathematics*, vol. 28, no. 4, pp. 877–883, 1988.

[53] S. Blanes, F. Casas, and A. Murua, "Splitting and composition methods in the numerical integration of differential equations," *arXiv preprint arXiv:0812.0377*, 2008.

[54] G. Tanaka, T. Yamane, J. B. Héroux, R. Nakane, N. Kanazawa, S. Takeda, H. Numata, D. Nakano, and A. Hirose, "Recent advances in physical reservoir computing: a review," *Neural Networks*, 2019.

[55] N. M. Estakhri, B. Edwards, and N. Engheta, "Inverse-designed metastructures that solve equations," *Science*, vol. 363, no. 6433, pp. 1333–1338, 2019.

[56] B. D. L. Costello and A. Adamatzky, "Calculating voronoi diagrams using chemical reactions," in *Advances in Unconventional Computing*. Springer, 2017, pp. 167–198.

[57] R. Sarpeshkar, "Analog versus digital: Extrapolating from electronics to neurobiology," *Neural Computation*, vol. 10, no. 7, pp. 1601–1638, Oct 1998.

[58] B. J. MacLennan, "A review of analog computing," *Department of Electrical Engineering & Computer Science, University of Tennessee, Technical Report UT-CS-07-601 (September)*, 2007.

[59] W. Little and A. Soudack, "On the analog computer solution of first-order partial differential equations," *Mathematics and Computers in Simulation*, vol. 7, no. 4, pp. 190 – 194, 1965. [Online] Available: http://www.sciencedirect.com/science/article/pii/S0378475465800350

[60] L. D. K. Ach and W. Comley, "The analog computer as a teaching aid in diffential equations," *Mathematics and Computers in Simulation*, vol. 3, no. 2, pp. 60 – 63, 1961. [Online] Available: http://www.sciencedirect.com/science/article/pii/S0378475461800244

[61] P.-A. Absil, "Continuous-time systems that solve computational problems." *IJUC*, vol. 2, no. 4, pp. 291–304, 2006.

[62] A. T. Doodson, "Tide-predicting machines," 1926.

[63] H. C. Lin and H. C. Chien, "Analog computing method of solving a second order differential equation," Mar. 29 1988, uS Patent 4,734,879.

[64] D. Welbourne, *Analogue Computing Methods: The Commonwealth and International Library: Applied Electricity and Electronics Division*. Elsevier, 2013.

[65] P. A. Holst, "George a. philbrick and polyphemus: The first electronic training simulator," *Annals of the History of Computing*, vol. 4, no. 2, pp. 143–156, April 1982.

[66] W. Keen, "Analog simulation, solution of field problems," 1961.

[67] T. D. Truitt and A. E. Rogers, *Basics of analog computers*. JF Rider, 1960, vol. 256.

[68] O. Bournez and M. L. Campagnolo, "A survey on continuous time computations," in *New Computational Paradigms*. Springer, 2008, pp. 383–423.

[69] J. R. Ragazzini, R. H. Randall, and F. A. Russell, "Analysis of problems in dynamics by electronic circuits," *Proceedings of the IRE*, vol. 35, no. 5, pp. 444–452, May 1947.

[70] J. S. Small, *The analogue alternative: The electronic analogue computer in Britain and the USA, 1930-1975*. Routledge, 2013, vol. 15.

[71] J. S. Small, "General-purpose electronic analog computing: 1945-1965," *IEEE Annals of the History of Computing*, vol. 15, no. 2, pp. 8–18, 1993.

[72] T. DeMichele. (2015) The antikythera mechanism is the oldest analog computer. [Online] Available: http://factmyth.com/factoids/the-antikythera-mechanism-is-the-oldest-analog-computer/

[73] T. Lindell. (2003) The torpedo data computer (tdc). [Online] Available: http://www.usscod.org/tdc.html

[74] A. Nordsieck, "The nordsieck computer," in *Managing Requirements Knowledge, International Workshop on*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, feb 1953, p. 227. [Online] Available: https://doi.ieeecomputersociety.org/10.1109/AFIPS.1953.21

[75] G. aircraft corporation. The GEDA commutator stabilization system. [Online] Available: http://www.cowardstereoview.com/analog/good.htm

[76] G. Bekey and W. Karplus, *Hybrid Computation*. John Wiley. New York, 1968.

[77] H. K. Skramstad, "Combined analog-digital techniques in simulation," in *Advances in Computers*. Elsevier, 1962, vol. 3, pp. 275–298.

[78] G. Kron, "Equivalent circuits for oscillating systems and the Riemann-Christoffel curvature tensor," *Transactions of the American Institute of Electrical Engineers*, vol. 62, no. 1, pp. 25–32, Jan 1943.

[79] G. Kron, "Numerical solution of ordinary and partial differential equations by means of equivalent circuits," *Journal of Applied Physics*, vol. 16, no. 3, pp. 172–186, 1945.

[80] W. R. Zimmerman, "Time domain solutions to partial differential equations using SPICE," *IEEE Transactions on Education*, vol. 39, no. 4, pp. 563–573, Nov 1996.

[81] G. Kron, "Equivalent circuit of the field equations of Maxwell-I," *Proceedings of the IRE*, vol. 32, no. 5, pp. 289–299, May 1944.

[82] W. R. Zimmerman, "Network analogs of Maxwell's field equations in one and two dimensions," *IEEE Transactions on Education*, vol. 25, no. 1, pp. 4–9, Feb 1982.

[83] G. Liebmann, "Solution of partial differential equations with a resistance network analogue," *British Journal of Applied Physics*, vol. 1, no. 4, p. 92, 1950.

[84] W. Karplus, "The use of electronic analogue computers with resistance network analogues," *British Journal of Applied Physics*, vol. 6, no. 10, p. 356, 1955.

[85] R. M. Howe and V. S. Haneman, "The solution of partial differential equations by difference methods using the electronic differential analyzer," *Proceedings of the IRE*, vol. 41, no. 10, pp. 1497–1508, Oct 1953.

[86] A. Carlson, "Handbook of analog computation," 1967.

[87] W. J. Karplus, *Analog simulation: solution of field problems.* McGraw-Hill, 1958.

[88] R. Vichnevetsky, "Hybrid methods for partial differential equations," *Simulation*, vol. 16, no. 4, pp. 169–180, 1971.

[89] R. Vichnevetsky, "Analog/hybrid solution of partial differential equations in the nuclear industry," *Simulation*, vol. 11, no. 6, pp. 269–281, 1968.

[90] W. D. Little, "Hybrid computer solutions of partial differential equations by monte carlo methods," in *Proceedings of the November 7-10, 1966, fall joint computer conference.* ACM, 1966, pp. 181–190.

[91] G. A. Korn, "Hybrid computer monte carlo techniques," *Simulation*, vol. 5, no. 4, pp. 234–245, 1965.

[92] T. Silvey and J. R. Barker, "Hybrid computing techniques for solving parabolic and hyperbolic partial differential equations," *The Computer Journal*, vol. 13, no. 2, pp. 164–170, 1970.

[93] W. J. Karplus, "A hybrid computer technique for treating nonlinear partial differential equations," *IEEE Transactions on Electronic Computers*, no. 5, pp. 597–605, 1964.

[94] V. Vemuri and J. A. Dracup, "Analysis of nonlinearities in ground water hydrology: A hybrid computer approach," *Water Resources Research*, vol. 3, no. 4, pp. 1047–1058, 1967.

[95] Y. Huang, N. Guo, S. Sethumadhavan, M. Seok, and Y. Tsividis, "A case study in analog co-processing for solving stochastic differential equations," in *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, Nov 2018, pp. 1–5.

[96] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[97] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, p. 80, 2008.

[98] J. J. Yang, D. B. Strukov, and D. R. Stewart, "Memristive devices for computing," *Nature nanotechnology*, vol. 8, no. 1, p. 13, 2013.

[99] C. Li, Y. Li, H. Jiang, W. Song, P. Lin, Z. Wang, J. J. Yang, Q. Xia, M. Hu, E. Montgomery, J. Zhang, N. Dvila, C. E. Graves, Z. Li, J. P. Strachan, R. S. Williams, N. Ge, M. Barnell, and Q. Wu, "Large memristor crossbars for analog computing," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1–4.

[100] M. Hu, C. E. Graves, C. Li, Y. Li, N. Ge, E. Montgomery, N. Davila, H. Jiang, R. S. Williams, J. J. Yang *et al.*, "Memristor-based analog computation and neural network classification with a dot product engine," *Advanced Materials*, vol. 30, no. 9, p. 1705914, 2018.

[101] M. A. Zidan, Y. Jeong, J. Lee, B. Chen, S. Huang, M. J. Kushner, and W. D. Lu, "A general memristor-based partial differential equation solver," *Nature Electronics*, vol. 1, no. 7, p. 411, 2018.

[102] I. Richter, K. Pas, X. Guo, R. Patel, J. Liu, E. Ipek, and E. G. Friedman, "Memristive accelerator for extreme scale linear solvers," in *Government Microcircuit Applications & Critical Technology Conference (GOMACTech)*, 2015.

[103] W. C. Chew, E. Michielssen, J. Song, and J.-M. Jin, *Fast and efficient algorithms in computational electromagnetics*. Artech House, Inc., 2001.

[104] J.-M. Jin, *The finite element method in electromagnetics*. John Wiley & Sons, 2015. [Online] Available: http://www.ebook.de/de/product/21080358/jian_ming_jin_the_finite_element_method_in_electromagnetics.html

[105] A. F. Peterson, S. L. Ray, R. Mittra, I. of Electrical, and E. Engineers, *Computational methods for electromagnetics*. IEEE press New York, 1998.

[106] L. Bruton, "Network transfer functions using the concept of frequency-dependent negative resistance," *IEEE Transactions on Circuit Theory*, vol. 16, no. 3, pp. 406–408, Aug 1969.

[107] L. Bruton, *RC active circuits: theory and design*. Prentice hall, 1980.

[108] N. Udayanga, A. Madanayake, and S. I. Hariharan, "Continuous-time algorithms for solving the electromagnetic wave equation in analog ICs," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 29–32.

[109] N. Udayanga, A. Madanayake, S. I. Hariharan, and N. Hawk, "Continuous-time analog computing circuits for solving the electromagnetic wave equation," in *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1–5.

[110] P. Ahmadi, B. Maundy, A. S. Elwakil, L. Belostotski, and A. Madanayake, "A new second-order all-pass filter in 130-nm CMOS," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 3, pp. 249–253, March 2016.

[111] A. Madanayake, L. Belostotski, C. Wijenayake, L. Bruton, and V. Devabhaktuni, "Analog 2-d iir beam filters for ears in uas ecosystems," in *2014 IEEE Antennas and Propagation Society International Symposium (APSURSI)*, July 2014, pp. 991–992.

[112] P. Ahmadi, B. Maundy, A. S. Elwakil, L. Belostotski, and A. Madanayake, "A new second-order all-pass filter in 130-nm CMOS," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 63, no. 3, pp. 249–253, Mar. 2016.

[113] C. Wijenayake, A. Madanayake, Y. Xu, L. Belostotski, and L. T. Bruton, "A steerable DC-1 GHz all-pass filter-sum RF space-time 2-D beam filter in 65 nm CMOS," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, May 2013, pp. 1276–1279.

[114] C. Wijenayake, Y. Xu, A. Madanayake, L. Belostotski, and L. T. Bruton, "RF analog beamforming fan filters using CMOS all-pass time delay approximations," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 5, pp. 1061–1073, May 2012.

[115] C. Wijenayake, A. Madanayake, L. Belostotski, Y. Xu, and L. T. Bruton, "All-pass filter-based 2-D IIR filter-enhanced beamformers for AESA receivers," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 61, no. 5, pp. 1331–1342, May 2014.

[116] N. Udayanga, A. Madanayake, C. Wijenayake, P. Ahmadi, L. Belostotski, B. Maundy, L. T. Bruton, and A. Elwakil, "All-pass filter based synthesis of multifunctional microwave active circuits," in *2017 IEEE 85th Vehicular Technology Conference (VTC Spring)*, June 2017, pp. 1–4.

[117] K. A. Hoffmann, *Computational fluid dynamics for engineers.* Engineering Education System, 1989.

[118] R. P. Sallen and E. L. Key, "A practical method of designing RC active filters," *IRE Transactions on Circuit Theory*, vol. 2, no. 1, pp. 74–85, March 1955.

[119] C. H. Papas, *Theory of electromagnetic wave propagation.* Courier Corporation, 2014.

[120] W.-C. Wang, *Electromagnetic wave theory.* Wiley, New York, 1986.

[121] T. Deliyannis, "High-q factor circuit with reduced sensitivity," *Electronics Letters*, vol. 4, no. 26, pp. 577–579, December 1968.

[122] P. Lax and B. Wendroff, "Systems of conservation laws," *Communications on Pure and Applied mathematics*, vol. 13, no. 2, pp. 217–237, 1960.

[123] A. Selle, R. Fedkiw, B. Kim, Y. Liu, and J. Rossignac, "An unconditionally stable maccormack method," *Journal of Scientific Computing*, vol. 35, no. 2-3, pp. 350–371, 2008.

[124] R. W. MacCormack, "Numerical solution of the interaction of a shock wave with a laminar boundary layer," in *Proceedings of the second international conference on numerical methods in fluid dynamics*. Springer, 1971, pp. 151–163.

[125] R. J. LeVeque, "Conservative methods for nonlinear problems," in *Numerical Methods for Conservation Laws*. Springer, 1990, pp. 122–135.

[126] A. Bressan, G.-Q. G. Chen, M. Lewicka, and D. Wang, *Nonlinear conservation laws and applications*. Springer, 2011.

[127] D. Gottlieb, E. Turkel, and S. Abarbanel, "Analysis of the error for approximations to systems of hyperbolic equations," *J. Comput. Phys.*, vol. 151, no. 2, pp. 997–1007, May 1999.

[128] S. I. Hariharan and H. C. Lester, "Acoustic shocks in a variable area duct containing near sonic flows," *Journal of Computational Physics*, vol. 58, no. 1, pp. 134–145, 1985.

[129] J. W. Haslett, M. K. N. Rao, and L. T. Bruton, "High-frequency active filter design using monolithic nullors," *IEEE Journal of Solid-State Circuits*, vol. 15, no. 6, pp. 955–962, Dec 1980.

[130] L. Bruton, "Low-sensitivity digital ladder filters," *IEEE Transactions on Circuits and Systems*, vol. 22, no. 3, pp. 168–176, Mar 1975.

[131] W. M. Sansen, *Analog design essentials*. Springer Science & Business Media, 2007, vol. 859.

[132] R. J. Baker, *CMOS: circuit design, layout, and simulation*, 3rd ed. Wiley-IEEE press, 2010.

[133] V. Saxena and R. J. Baker, "Indirect feedback compensation of cmos op-amps," in *2006 IEEE Workshop on Microelectronics and Electron Devices, 2006. WMED '06.*, 2006, pp. 2 pp.–4.

[134] V. Saxena and R. J. Baker, "Indirect compensation techniques for three-stage cmos op-amps," in *2009 52nd IEEE International Midwest Symposium on Circuits and Systems*, Aug 2009, pp. 9–12.

[135] J. C. Spall, "Implementation of the simultaneous perturbation algorithm for stochastic optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 3, pp. 817–823, July 1998.

[136] J. C. Spall, "An overview of the simultaneous perturbation method for efficient optimization," *Johns Hopkins apl technical digest*, vol. 19, no. 4, pp. 482–492, 1998.

[137] J. Kiefer, J. Wolfowitz *et al.*, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

APPENDIX A

LAYOUT AND SCHEMATIC OF THE EVALUATION BOARD

Figs. 1 and 2 show the layout and the schematic of the evaluation board that was designed to test the AC, respectively. The evaluation board consists of 4 layers.

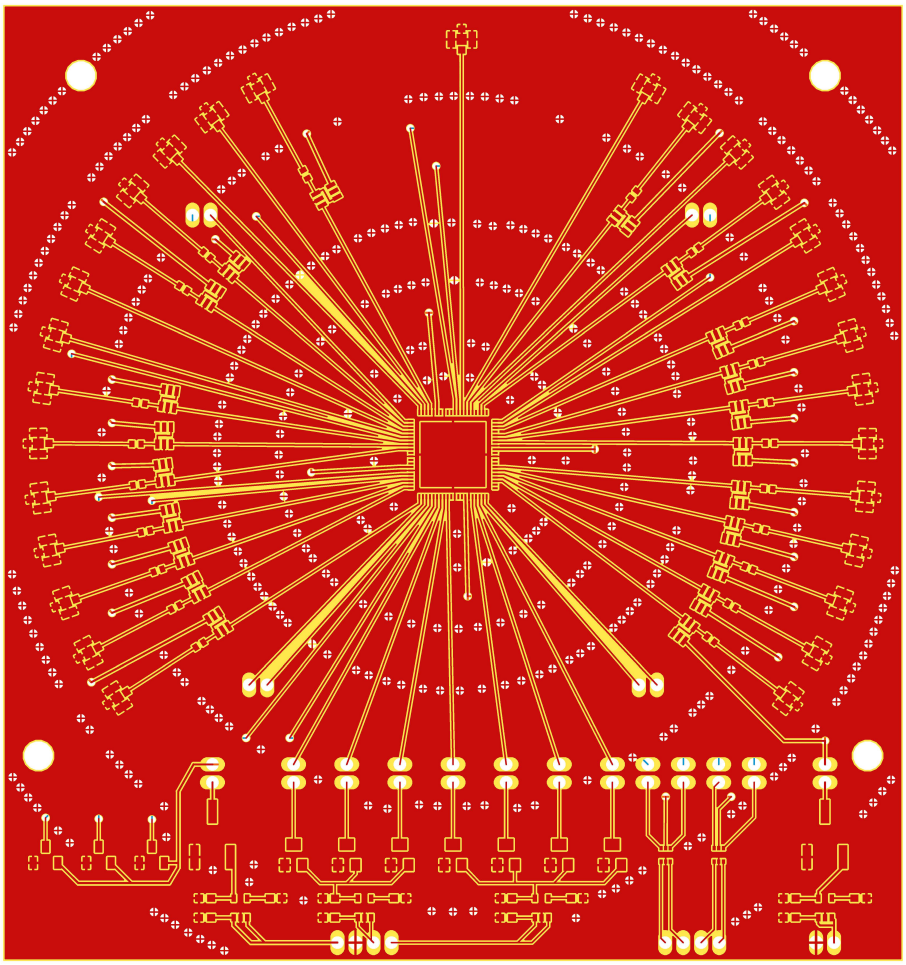

Figure 1: The layout of the evaluation board.

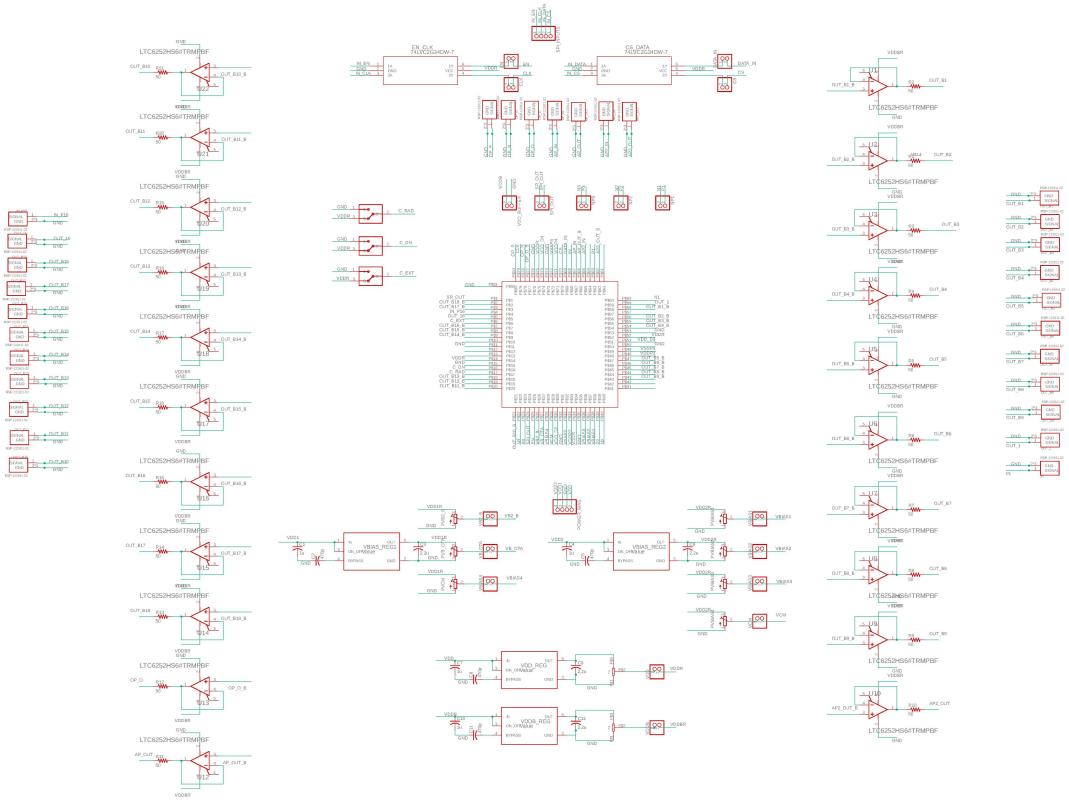

Figure 2: The schematic of the evaluation board.

VITA

NILAN UDAYANGA GALABADA KANKANAMGE

April 17, 1987	Born, Mahamodara, Sri Lanka
2011	B.Sc., Electronics and Telecommunication Engineering University of Moratuwa Katubedda, Sri Lanka
2015	M.Sc., Electrical and Computer Engineering The University of Akron Akron, Ohio

PUBLICATIONS AND PRESENTATIONS

1. Continuous-time Algorithms for Solving Maxwell's Equations using Analog Circuits, N. Udayanga, S.I. Hariharan, S. Mandal, L. Belostotski, Len T. Bruton, and A. Madanayake, IEEE Transactions on Circuits and Systems I, vol. 66, no. 10, pp. 3941–3954, Oct 2019.
2. Design and Digital Implementation of Fast and Recursive DCT II-IV Algorithms, S.M. Perera, A. Madanayake, N. Dornback, and N. Udayanga, Circuits System Signal Process, vol. 38, no 2, pp. 529–555, Feb 2019.
3. Low-Complexity Wideband Transmit Beamforming Using Network-Resonant Digital Plane-Wave Filters, C. Wijenayake, S. Pulipati, A. Madanayake, N. Udayanga and L. Bruton, IEEE Antennas and Wireless Propagation Letters, vol. 17, no. 7, pp. 1300-1304, July 2018.
4. Wideband N-Beam Arrays Using Low-Complexity Algorithms and Mixed-Signal Integrated Circuits, S. M. Perera, V. Ariyarathna, N. Udayanga, A. Madanayake, G. Wu, L. Belostotski, Y. Wang, S. Mandal, R. J. Cintra, and T. S. Rappaport, IEEE Journal of Selected Topics in Signal Processing, vol. 12, no. 2, pp. 368-382, May 2018.
5. A High-Speed CMOS Analog Computer for Solving the 1-D Wave Equation, N. Udayanga, A. Madanayake, S. I. Hariharan, J. Liang, S. Mandal, and L. Belostotski, Custom Integrated Circuits Conference (CICC), 2020 (under review).
6. Analog Coprocessors for Solving Partial Differential Equations (Abstract), N. Udayanga, A. Madanayake, S. I. Hariharan, J. Liang, S. Mandal, and L. Belostotski, USNC-URSI National Radio Science Meeting, 2019.

7. Continuous-time Analog Computing Circuits for Solving The Electromagnetic Wave Equation, N. Udayanga and A. Madanayake and S. I. Hariharan, in IEEE International Symposium on Circuits and Systems (ISCAS), May 2018.
8. Analog 65/130 nm CMOS 5 GHz Sub-Arrays with ROACH-2 FPGA Beamformers for Hybrid Aperture-Array Receivers, V. Ariyarathna, N. Udayanga, A. Madanayake, L. Belostotski, P. Ahmadi, S. Mandal, and A. Nikoofard, GOMACTech, March 2017.
9. Continuous-time algorithms for solving the electromagnetic wave equation in analog ICs, N. Udayanga and A. Madanayake and S. I. Hariharan, in IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), Aug 2017, pp. 29-32.
10. All-Pass Filter Based Synthesis of Multifunctional Microwave Active Circuits, N. Udayanga and A. Madanayake and C. Wijenayake and P. Ahmadi and L. Belostotski and B. Maundy and L. T. Bruton and A. Elwakil, in IEEE 85th Vehicular Technology Conference (VTC Spring), June 2017.
11. Design of a low-complexity wideband analog true-time-delay 5-beam array in 65nm CMOS, A. Madanayake and V. Ariyarathna and N. Udayanga and L. Belostotski and S. K. Perera and R. J. Cintra, in IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS), Aug 2017, pp. 1204-1207.
12. Design methodology of an analog 9-beam squint-free wideband IF multi-beamformer for mmW applications, V. Ariyarathna and N. Udayanga and A. Madanayake and S. M. Perera and L. Belostotski and R. J. Cintra, in Moratuwa Engineering Research Conference (MERCon), May 2017, pp. 236-240.
13. Wideband delay-sum digital aperture using Thiran all-pass fractional delay filters, A. Madanayake and N. Udayanga and V. Ariyarathna, in IEEE Radar Conference (RadarConf), May 2016.
14. An Analog-digital Hybrid Wave Equation Solver using a 180 nm Analog CMOS Chip and ROACH-2 Digital FPGA, N. Udayanga, J. Liang, A. Madanayake, S. I. Hariharan, S. Mandal, and L. Belostotski, CASPER workshop, Center for Astrophysics, Harvard University, Cambridge, 2019.
15. Analog Coprocessors for Solving Partial Differential Equations, N. Udayanga, A. Madanayake, S. I. Hariharan, J. Liang, S. Mandal, and L. Belostotski, USNC-URSI National Radio Science Meeting, Boulder, Colorado, 2019.
16. Towards High-Speed Analog PDE Solvers in Standard CMOS, N. Udayanga, A. Madanayake, S. I. Hariharan, J. Liang, S. Mandal, and L. Belostotski, IEEE AP/MTT Graduate Student Poster Competition, The ElectroScience Laboratory, The Ohio State University, 2018.