

Title	The role of FPGAs in Modern Option Pricing techniques: A survey
Authors	O'Mahony, Aidan;Hanzon, Bernard;Popovici, Emanuel
Publication date	2024-08-12
Original Citation	O'Mahony, A., Hanzon, B. and Popovici, E. (2024) 'The role of FPGAs in Modern Option Pricing Techniques: a survey', Electronics, 13(16), 3186. Available at: https://doi.org/10.3390/electronics13163186
Type of publication	Article (peer-reviewed);journal-article
Link to publisher's version	10.3390/electronics13163186
Rights	© 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/) - https://creativecommons.org/licenses/by/4.0/
Download date	2026-10-28 19:15:56
Item downloaded from	https://hdl.handle.net/10468/16309

Review

The Role of FPGAs in Modern Option Pricing Techniques: A Survey

Aidan O Mahony ^{1,*} , Bernard Hanzon ²  and Emanuel Popovici ¹

¹ Department of Electrical and Electronic Engineering, University College Cork, T12 CY82 Cork, Ireland; e.popovici@ucc.ie

² Department of Mathematics, University College Cork, T12 CY82 Cork, Ireland; b.hanzon@ucc.ie

* Correspondence: 103837793@umail.ucc.ie; Tel.: +353-214902210

Abstract: In financial computation, Field Programmable Gate Arrays (FPGAs) have emerged as a transformative technology, particularly in the domain of option pricing. This study presents the impact of Field Programmable Gate Arrays (FPGAs) on computational methods in finance, with an emphasis on option pricing. Our review examined 99 selected studies from an initial pool of 131, revealing how FPGAs substantially enhance both the speed and energy efficiency of various financial models, particularly Black–Scholes and Monte Carlo simulations. Notably, the performance gains—ranging from 270- to 5400-times faster than conventional CPU implementations—are highly dependent on the specific option pricing model employed. These findings illustrate FPGAs’ capability to efficiently process complex financial computations while consuming less energy. Despite these benefits, this paper highlights persistent challenges in FPGA design optimization and programming complexity. This study not only emphasises the potential of FPGAs to further innovate financial computing but also outlines the critical areas for future research to overcome existing barriers and fully leverage FPGA technology in future financial applications.

Keywords: field programmable gate arrays (FPGAs); option pricing; literature review; financial computation; high-frequency trading; Black–Scholes model; energy efficiency; FPGA optimization; hardware acceleration



Citation: O Mahony, A.; Hanzon, B.; Popovici, E. The Role of FPGAs in Modern Option Pricing Techniques: A Survey. *Electronics* **2024**, *13*, 3186. <https://doi.org/10.3390/electronics13163186>

Academic Editor: Raffaele Giordano

Received: 4 June 2024

Revised: 30 July 2024

Accepted: 2 August 2024

Published: 12 August 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The advent of Field Programmable Gate Arrays (FPGAs) has ushered in a new era in the domain of financial computation, notably in the computationally intensive field of option pricing [1,2]. The ability of FPGAs to perform parallel computations at high speeds, while maintaining energy efficiency, has positioned them as a relevant technology, capable of transforming traditional computational methodologies in financial applications. The primary goal of this survey is to review and analyze the existing literature on the application of FPGAs in modern option pricing techniques, highlighting their impact, benefits, and the challenges associated with their implementation. This paper provides a journey through the existing literature, exploring the various applications, challenges, and future prospects of FPGAs in financial computations, with a particular focus on the speedups of option pricing and also with consideration to power efficiency.

The financial industry has witnessed a surge in the complexity and volume of transactions, necessitating the adoption of robust, efficient, and rapid computational technologies. FPGAs, with their reprogrammable silicon, offer both parallelism and computational power, enabling them to execute complex algorithms at unparalleled speeds. This is especially important in areas like option pricing, where models such as the Black–Scholes model and its various derivatives demand significant computational resources, and in high-frequency trading, where speed is critical.

Despite the promising attributes of FPGAs, their application in financial computation is not devoid of challenges. Issues pertaining to design optimization, energy efficiency,

and the steep learning curve associated with FPGA programming have emerged as notable hurdles. This paper, through a review of the existing literature, seeks to provide a comprehensive overview of the current state of FPGA applications in financial computation, identifying their capabilities, highlighting persistent challenges, and paving the way for future research trajectories.

The remainder of the paper is organized as follows: Section 2 details the methodology employed in sourcing and analyzing the literature. Section 3 provide a brief introduction to various types of accelerators. Section 4 delivers a review of FPGA implementations for option pricing, while Section 5 looks into the challenges and future directions of FPGA applications in financial computation. Finally, Section 6 concludes the paper, summarizing the key findings and providing insights into potential future research avenues.

2. Methodology

2.1. Search Strategy

To conduct the study, we followed the guidelines on literature reviews provided in [3–5]. We used the search terms + FPGA + Option pricing + Hardware acceleration + Financial simulations + Monte Carlo simulations + Black–Scholes model + Heston model + High-frequency trading + Financial engineering + FPGA-based option priced algorithms + FPGA optimization + FPGA frameworks + FPGA synthesis + FPGA parallelism + FPGA energy efficiency + FPGA design + Derivatives pricing to search in the following digital libraries: IEEE Xplore; ACM Digital Library; SpringerLink; ScienceDirect; Google Scholar; and ProQuest. We gathered 131 papers.

2.2. Inclusion and Exclusion Criteria

In order to decide which of them further analyze, we performed three exclusion stages. Firstly we filtered the papers based on titles, then on abstracts. In reading the abstracts we excluded papers regarding non-engineering aspects (e.g., papers addressing purely economic aspects of option pricing). Finally, we were left with 99 papers distributed across the years illustrated in Figure 1. If we consider the papers published in recent years, i.e., 2021 onwards, we see modern systems such as the Intel Stratix V FPGA (Intel Terasic Stratix-V GX FPGA, Intel Corporation, Santa Clara, CA, USA) [6], Intel Stratix 10 GX (Intel Corporation, Santa Clara, CA, USA) [7], NVIDIA RTX A5000 (Nvidia Corporation, Santa Clara, CA, USA) [8], Intel Xeon Platinum 8260L (Intel Corporation, Santa Clara, CA, USA) (Cascade Lake) [9] are used as offloading and acceleration hardware. Furthermore, modern frameworks such as SYCL (Khronos Group SYCL 2020 Specification, OR, USA) [9], OpenCL (Khronos Group OpenCL 1.0 Specification, OR, USA) [10], Intel OneAPI (version 21.2, Intel Corporation, Santa Clara, CA, USA) [7] and CUDA (Nvidia CUDA capability 1.0, Nvidia Corporation, Santa Clara, CA, USA) [11] are used in many papers to enable rapid development of option pricing solutions.

2.3. Data Extraction

From reading the 99 papers, we were able to create 7 categories.

- Black–Scholes Model
- Binomial and Trinomial Tree Methods
- Monte Carlo Simulations
- Finite Difference
- Heston Model
- Quadrature Methods
- Miscellaneous

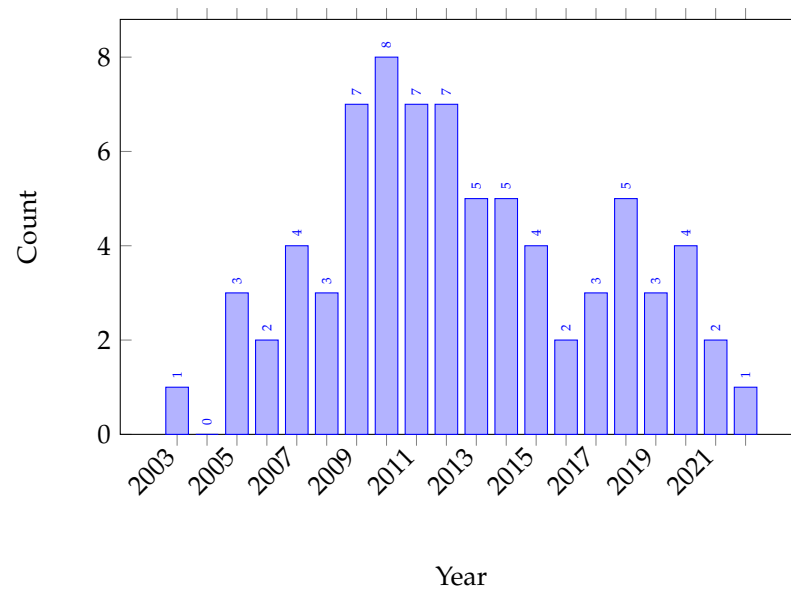


Figure 1. Bar chart of yearly counts of publications.

3. Comparison of FPGA, CPU, and GPU

It is useful to provide a discussion highlighting the differences between the main hardware accelerator types. The three main types of hardware for implementing algorithms are CPUs, GPUs, and FPGAs, each with distinct advantages and disadvantages [12,13]. CPUs, the most common processors, are designed for general-purpose tasks and excel at executing sequential instructions quickly. However, they are less suited for parallel processing tasks like those in machine learning and image processing.

GPUs, with their numerous parallel computing cores, are ideal for tasks such as gaming, 3D rendering, and machine learning, thanks to their high memory bandwidth. Despite their performance, they lack support for various software libraries and have limited instruction sets, making them less suitable for general-purpose computing.

FPGAs can be programmed with custom logic circuits to perform specific functions efficiently and typically consume less power than CPUs and GPUs, making them perfect for embedded applications. However, they require specialized skills and tools for programming, which makes them challenging for general-purpose computing. Table 1 provides a detailed comparison of FPGA, CPU, and GPU across various noteworthy aspects such as power consumption, computing power, flexibility, latency, programming complexity, development time, cost, and parallelism [6,14]. In summary, CPUs are great for sequential tasks, GPUs excel in parallel processing, and FPGAs offer customizable, power-efficient solutions for specific applications.

Table 1. Comparison between FPGA, CPU, and GPU across various aspects.

Aspect	FPGA	CPU	GPU
Consumption	Low to Moderate	Moderate to High	High
Computing Power	High (specific tasks)	Moderate (general)	Very High (parallel tasks)
Flexibility	High (custom hardware)	Very High (general)	Moderate (parallel optimized)
Latency	Very Low	Moderate	Low to Moderate
Programming	Complex (hardware knowledge)	Easy (high-level languages)	Moderate (requires parallelism)
Development Time	Long (custom design)	Short (established tools)	Moderate (specialized tools)
Cost	High (initial, low volume)	Low to Moderate	Moderate to High
Parallelism	High (custom pipelines)	Low to Moderate	Very High
Use Case	Specialized, real-time	General-purpose	Parallel processing, AI/ML

4. Review of FPGA Implementations for Option Pricing

4.1. Black–Scholes Model

The Black–Scholes Model, introduced by Fischer Black, Myron Scholes [15], and later expanded by Robert Merton [16], is a fundamental mathematical model for pricing European-style options. It assumes that the price of the underlying asset follows a geometric Brownian motion with constant drift and volatility, no transaction costs or taxes, and continuous trading. The model provides a formula for calculating the price of call and put options based on variables such as the current price of the underlying asset, strike price, risk-free interest rate, time to expiration, and asset volatility. Despite its simplifying assumptions, the Black–Scholes Model is widely used for its robustness and forms the basis for modern financial theory and risk management strategies. It has significantly impacted the growth of options markets and financial engineering.

4.1.1. Black–Scholes Model on FPGA

The Black–Scholes algorithm has been deployed onto a Xilinx Virtex-II Pro architecture, partitioning the model across a processor and FPGA, which outperformed the Mathematica software by completing computations in approximately 1% of the time [17] giving a 3400× speedup over the ARM processor implementation and 312× speedup over the Mathematica implementation. In [18], the authors introduced an FPGA-based pricing model for complex options, such as ‘Asian’ options, using a parameterized Monte Carlo version of the Black–Scholes model. Their FPGA solution demonstrated a speedup of 270× for a single FPGA compared to a Xeon blade CPU and 5400× speedup for 16 FPGAs compared to a Xeon blade CPU.

Another study presented a hardware architecture for generating random vectors applied to Delta-Gamma Value-at-Risk simulations and Black–Scholes option pricing models. This FPGA solution’s scalability was emphasized, showing a generation rate 200 times that of a single Opteron 2.2 GHz [19]. Indeed, in the previous example it is noted that a single FPGA can replace the equivalent of 33 quad-core computers (132 CPU cores) in a cluster, thereby significantly reducing the heat, power, space, and capital outlay required for a 32U rack to just one computer with an FPGA. The authors of [20] explored computational systems for enhancing investment decision speed and accuracy, introducing a novel FPGA cluster approach called SMILE. The SMILE architecture with 32 nodes is approximately five times faster than a traditional computing cluster with the same number of nodes. To achieve similar response times to the 32-node SMILE cluster, the traditional computing cluster (ALTAMIRA) needs to scale up to 256 processors.

An OpenCL-based kernel was proposed for European Stock Option pricing using the Monte Carlo Black–Scholes algorithm on an FPGA device. This FPGA-based approach significantly outperformed CPU and GPU architectures, achieving computational times in milliseconds [21] (exact times were not reported). Ref. [22] presented a high-performance Black–Scholes system for pricing European call options on an Altera Stratix-V FPGA, emphasizing the FPGA’s advantages of dedicated hardware and efficient complex computations. Exact speedup comparisons were not possible for this paper.

The industrial readiness of High-Level Synthesis (HLS) tools for FPGAs, particularly for pricing Black–Scholes and Heston model-based options has been illustrated by showing how HLS tools outperformed traditional CPU, GPU, and Xeon Phi implementations, endorsing their viability in business scenarios [2]. The authors present up to a 221-times speedup compared to a sequential CPU implementation. Another project implemented a structured product pricer on an FPGA, achieving speed-ups of 550 to 1450 times compared to a one-core software solution [23].

The authors of [24] introduced a design flow for efficient hardware accelerators for option pricing on an FPGA platform, outperforming most manually-designed engines and software implementations by a 2× factor. Another methodology for option pricing tasks on data center FPGAs, emphasizing power efficiency improvements (30% more floating point operations per Joule of energy) in computational finance applications was presented

in [25]. And finally, a thesis [26] focused on FPGA-based accelerators for computational finance models, with FPGAs achieving 4× to 5× more operations per Watt than a GPU.

An implementation the Black-Scholes formula on a Pynq-Z1 FPGA is presented in [27], but the results indicated inefficiencies (slowdown) in the FPGA solution compared to a Python execution on the ARM. An emphasis on the high potential of specialized architectures for accelerating financial product pricing on commercially available FPGAs is presented in [28], which provided a speedup of between 550× to 1450×. The research in [29] introduced an efficient hardware structure for the Black-Scholes option pricing model, outpacing CPU and GPU implementations in terms of throughput. This research shows a 365× speedup over similar CPU implementations and 2.6× speedup over GPU implementations with comparable chip manufacturing processes. They also report a throughput-power efficiency speedup of 3293 times compared to a CPU and 59.4 times compared to a GPU.

The paper [30] introduced a parallelized calculation framework for computing implied volatility based on the Black-Scholes Model, with the FPGA implementation showcasing at least 4× to 5× speedup over traditional methods. Lastly, ref. [11] discussed the implementation of the Black-Scholes-Merton options pricing formula on various hardware platforms (presenting an estimated 298× speedup for Black-Scholes), highlighting that both GPU and FPGA performances were primarily limited by the interconnect.

4.1.2. Key Observations

Studies such as those by [17,18,21] have showcased the remarkable speed and efficiency gains achieved by FPGA-based systems, often outperforming their counterparts by orders of magnitude.

Another observation is the increasing prominence of High-Level Synthesis (HLS) tools in the area of FPGA-based financial computations. Tools like HLS allow for a more seamless transition from high-level programming languages to hardware descriptions, simplifying the design process and making FPGA platforms more accessible to developers without deep hardware expertise. Ref. [2] and others have demonstrated the industrial readiness of these tools, emphasizing their potential to outperform traditional computational platforms. However, it is also worth noting that not all FPGA implementations guarantee superior performance. As highlighted by the study from [27], inefficiencies, especially those related to communication times with the FPGA, can sometimes lead to sub-optimal results, underscoring the importance of meticulous design and optimization.

Lastly, the choice of the FPGA platform and the specific optimization techniques employed play an important role in determining performance outcomes. As seen in the works of [26,29], the selection of the FPGA platform, whether datacenter-class or embedded-class, and the integration of specific algorithms and structures can significantly influence the computational speed and efficiency.

As evidenced by studies like [26,29,30], there is a growing emphasis on harnessing the full potential of FPGA platforms, whether they are datacenter-class or embedded-class. The integration of High-Level Synthesis (HLS) tools and platforms such as OpenCL, as highlighted by [2,25], underscores the industry's move towards making FPGA development more accessible and efficient. Furthermore, the consistent pursuit of power efficiency, as seen in [25], indicates a broader trend towards sustainable and energy-efficient computational finance solutions. Table 2 shows a consolidated summary of notable findings.

Table 2. Summary of notable findings from various studies on FPGA implementations of the Black–Scholes model.

Study	Notable Findings
Tandon (2003) [17]	Implemented Black–Scholes on Xilinx Virtex-II Pro, achieving a 3400× speedup over ARM processor and 312× over Mathematica.
Baxter et al. (2007) [18]	FPGA-based pricing model for ‘Asian’ options, showing a 270× speedup for a single FPGA and 5400× for 16 FPGAs compared to a Xeon blade CPU.
Thomas et al. (2007) [19]	Hardware architecture for generating random vectors for Delta–Gamma Value-at-Risk and Black–Scholes, achieving 200× the generation rate of a single Opteron 2.2 GHz CPU. Equivalent to 33 quad-core CPUs.
Bruce (2008) [11]	Implemented Black–Scholes–Merton options pricing on various hardware platforms, estimating a 298× speedup for Black–Scholes. Highlighted interconnect limitations for both GPU and FPGA.
Castillo et al. (2009) [20]	SMILE architecture with 32 nodes, 5× faster than a traditional computing cluster with the same number of nodes. Traditional cluster needs 256 processors to match SMILE’s response times.
Patel et al. (2016) [21]	OpenCL-based kernel for European Stock Option pricing on FPGA, significantly outperforming CPU and GPU architectures, achieving computational times in milliseconds.
Choo et al. (2013) [22]	High-performance Black–Scholes system on Altera Stratix-V FPGA for pricing European call options.
Inggs (2015) [2]	Demonstrated HLS tools’ industrial readiness for Black–Scholes and Heston models, achieving up to 221× speedup over sequential CPU implementation.
Guerrero et al. (2015) [23]	Structured product pricer on FPGA, achieving speed-ups of 550 to 1450 times compared to a one-core software solution.
Pham et al. (2016) [24]	Design flow for efficient hardware accelerators for option pricing on FPGA, outperforming most manually-designed engines and software implementations by 2×.
Inggs (2016) [25]	Emphasized power efficiency improvements (30% more floating point operations per Joule of energy) in computational finance on FPGA.
Ma (2019) [26]	FPGA-based accelerators for computational finance, achieving 4× to 5× more operations per Watt than GPU.
Rodrigues et al. (2019) [28]	Specialized architectures for financial product pricing on FPGA, providing a speedup of 550× to 1450×.
de Castro (2019) [27]	results indicated inefficiencies (slowdown) in the FPGA solution compared to a Python execution on the ARM.
Li et al. (2021) [29]	Efficient hardware structure for Black–Scholes on FPGA, showing a 365× speedup over CPU implementations and 2.6× over GPU. Throughput-power efficiency speedup of 3293× compared to CPU and 59.4× compared to GPU.
Wang et al. (2022) [30]	Parallelized framework for computing implied volatility on FPGA, showing 4× to 5× speedup over traditional methods.

4.2. Binomial and Trinomial Tree Methods

The Binomial and Trinomial Tree Methods are numerical techniques used to price options by discretizing the continuous price movements of the underlying asset. Developed by John Cox, Stephen Ross, and Mark Rubinstein in 1979 [31], the binomial method models the asset price movements over discrete time intervals, allowing for two possible outcomes

(up or down) at each step. The trinomial method extends this by introducing a third possible outcome (no change), improving accuracy for more complex options. These methods are particularly useful for American options, which can be exercised at any time before expiration. By constructing a recombining tree of possible prices and working backward from the option's expiration, these methods determine the option's fair value at each node.

4.2.1. Binomial and Trinomial Tree Methods on FPGA

The implementation of binomial option pricing models on FPGAs, particularly for American options, has been a topic of interest in recent years. One such study utilized OpenCL to achieve this, aiming to reduce energy consumption while maintaining high parallelism. The results revealed that the optimal FPGA implementation could evaluate over 2000 options/s with an average power consumption of less than 20 W [32].

A parallel pipelined architecture designed for accelerating financial computations using binomial-tree pricing models was introduced, which was specifically tailored to handle concurrent requests for option valuations. When implemented on an xc4vsx55 FPGA, this architecture was found to be over 250 times faster than a Core2 Duo processor operating at 2.2 GHz. Moreover, it outperformed an Nvidia Geforce 7900GTX processor with 24 pipelines running at 650 MHz by more than twice the speed, suggesting a promising avenue for future optimizations in performance, area, and power [33].

Another innovative approach introduced a scalable, reconfigurable hardware architecture for pricing American options using the Binomial Lattice algorithm. This method provided double precision floating point pricing and could evaluate up to $N = 64,000$ time steps. Notably, it achieved a 73× speedup over an optimized CPU implementation, emphasizing its efficiency on a per-asset basis [34].

Further research explored two parallel pipelined architectures on an FPGA for accelerating option pricing models using both binomial and trinomial trees. These architectures were capable of performing over 100 times faster than a Core2 Duo processor and twice as fast as a non-CUDA Nvidia GPU. The study also reported on power consumption where their solution which operates at 160 MHz consumes 5.7 Watts, making it approximately 6-times more efficient than the Geforce 8600GTS, which requires 71 Watts, in terms of calculation speed per watt [35].

Beyond binomial and trinomial trees, there has been work on FPGA-based parallel processors optimized for solving the Black–Scholes partial derivative equation, a staple in financial option pricing. One such processor achieved a 5× speedup compared to a 2 GHz dual-core Intel CPU, showcasing its flexibility and scalability for various applications [36].

In the area of Monte Carlo option pricing, a novel FPGA-based method was proposed that utilized a discrete-space random walk over a binomial lattice. This method was hypothesized to enhance parallelization and performance without compromising accuracy. Experimental results confirmed this, showing a 50× improvement in throughput compared to existing FPGA methods [37].

The energy efficiency of FPGAs in financial computations, especially in the binomial option pricing model for American options, was explored using Altera's OpenCL implementation. The results from this study were promising, specifically considering energy efficiency, the FPGA implementation is more than 5 times more energy efficient than the software implementation [32].

A systolic hardware architecture designed for pricing American options using the binomial-tree model on FPGAs demonstrated a linear increase in latency with the size of the binomial tree. This architecture outperformed previous works by achieving a 65× improvement in option latency [38].

Another study introduced a formal mathematical framework for the binomial-tree model, which facilitated FPGA-based solutions for various binomial-tree problems. The reported performance ranges from 1.4× throughput compared with a hand-tuned systolic

design, and up to to 9.1× and 5.6× improvement when compared to scalar and vector architectures [39].

Comparing the performance and energy efficiency of OpenCL-accelerated computations across different platforms, one study found that while GPUs might offer superior throughput, FPGAs displayed better performance portability and energy efficiency in some cases. More specifically the authors report a 68% device peak performance for the FPGA compared to 20% for an NVIDIA GPU. Also, they report up to a 1.4× improvement of energy efficiency for some cases [40].

The potential of FPGA technology in multi-asset option pricing was also explored. One study presented an architecture for a two-asset European option pricer based on Pascal's simplex, which demonstrated the efficiency of hardware acceleration for multi-asset option pricing [10] requiring an estimated 39.3 Watts. This study leverages pipelining to reduce latency of computing option pricing by 25×. Another study introduced a multi-asset option pricer using Pascal's simplex and a recombining multinomial tree approach, achieving computation speeds up to 43 times faster than a software-based general-purpose processor [7].

4.2.2. Key Observations

The binomial and trinomial tree methods, when implemented on FPGAs, consistently outperform traditional CPU-based methods in terms of speed and efficiency [33–35]. The introduction of parallel pipelined architectures, systolic hardware designs, and formal mathematical frameworks showcases the ongoing innovation in this field [33,38,39]. Furthermore, the exploration into multi-asset option pricing using FPGA technology underscores the expanding horizons of its applications, with studies achieving remarkable computation speeds and efficiencies [7,10].

As computational demands in the financial sector grow, there is an evident gravitation towards FPGA-based solutions due to their inherent parallelism and energy efficiency [32]. The consistent exploration and development of parallel architectures, from pipelined to systolic designs, highlight the industry's commitment to optimizing performance and throughput [33,38]. Additionally, the increasing integration of OpenCL frameworks with FPGA implementations signifies a move towards more accessible and adaptable hardware programming paradigms, allowing for broader applications and easier adaptability across various financial models [7,40]. Table 3 summarizes the notable findings from this section.

Table 3. Summary of notable findings from various studies on FPGA implementations of binomial and trinomial tree methods.

Study	Notable Findings
Morales et al. (2014) [32]	FPGA implementation evaluated over 2000 options/s with <20 W power consumption.
Jin et al. (2008) [33]	250× faster than Core2 Duo, 2× faster than Nvidia Geforce 7900GTX.
Wynnyk and Magdon-Ismail (2009) [34]	73× speedup over optimized CPU implementation.
Jin et al. (2009) [35]	100× faster than Core2 Duo, 2× faster than non-CUDA Nvidia GPU, 6× more efficient in power consumption.
Chatziparaskevas et al. (2012) [36]	5× speedup compared to 2 GHz dual-core Intel CPU.
Tavakkoli et al. (2014) [38]	65× improvement in option latency
Fabry et al. (2017) [37]	50× improvement in throughput for Monte Carlo option pricing.
Morales et al. (2014) [32]	5× more energy efficient than software implementation.

Table 3. Cont.

Study	Notable Findings
Tavakkoli et al. (2017) [39]	1.4× throughput vs. hand-tuned systolic design, up to 9.1× and 5.6× improvement vs. scalar and vector architectures.
Minhas et al. (2018) [40]	68% device peak performance for FPGA vs. 20% for NVIDIA GPU, up to 1.4× energy efficiency improvement.
Mahony et al. (2020) [10]	Two-asset European option pricer, 25× latency reduction, 39.3 W power consumption.
Mahony et al. (2022) [7]	Multi-asset option pricer, 43× faster than software-based general-purpose processor.

4.3. Monte Carlo Simulations

Monte Carlo Simulations are numerical techniques used in financial modeling to estimate the value of options and other derivatives by simulating the random paths of the underlying asset's price [41]. These simulations rely on repeated random sampling to compute their results, making them particularly useful for pricing complex options where analytical solutions are difficult or impossible to obtain. The method involves generating numerous potential future price paths for the underlying asset using stochastic processes, then calculating the payoff for each path and averaging the results to obtain an estimated option price. Monte Carlo simulations are highly flexible and can handle various payoff structures and market conditions, including American options and path-dependent options. Their accuracy increases with the number of simulations, though they can be computationally intensive.

4.3.1. Monte Carlo Simulations on FPGA

The advancements in Field-Programmable Gate Arrays (FPGAs) have significantly revolutionized the acceleration of Monte Carlo simulations, especially in the domain of financial applications.

In the early stages of this evolution, ref. [42] made a significant contribution by introducing a hardware accelerator for Monte Carlo simulations on FPGAs. This innovation achieved a remarkable 50-fold speedup for financial applications using the BGM interest rate model. This pioneering work paved the way for further exploration, as evidenced by [43], which demonstrated a 12-fold speedup. This exploration was deepened by [44], which presented a modular framework and achieved speedups between 8 and 71 times for two discrete-time random walks. Ref. [45] introduced the 'HyperStreams' abstraction, achieving a 146× acceleration for European option pricing. The Monte-Carlo simulation engine's implementation on the FPGA-based Maxwell supercomputer further emphasized this potential. This engine, as discussed in [46,47], outperformed software solutions by factors ranging from 340 to 750 times. Additionally, ref. [48] documented a 20× speedup by implementing the LSMC method for American option pricing on an FPGA with a considerable energy savings (>20:1). The versatility of FPGAs in financial computations was further showcased by [49], which introduced an FPGA accelerator for QMC derivative pricing, surpassing the performance of a 3 GHz processor by over 50 times.

Describing an FPGA hardware architecture, ref. [50] focuses on expediting the Least-Squares Monte Carlo (LSMC) method for American option pricing. The FPGA-based solution achieved remarkable speed-ups: 25× in path generation and 18× in regression phases compared to its software counterpart. This resulted in an overall speed-up of 20× and an energy efficiency of 54×. Although FPGA design takes longer than software development, leveraging existing IP cores and the evolution of reconfigurable hardware present promising avenues for diverse algorithms. The authors also planned to look into heterogeneous computing platforms and fine-tune arithmetic types and word lengths for enhanced precision and efficiency.

Several studies have highlighted the adaptability and efficiency of FPGAs. Ref. [51] introduced Contessa, a high-level language tailored for Monte Carlo simulations in financial computing. This language facilitated a direct route to FPGA-accelerated performance, showcasing significant speedup (maximum 62×) and reduced power consumption (a tenth of the power consumption, reported by the authors, when compared to a CPU implementation. Precise figures were not available). Ref. [52] presented a dynamic scheduling Monte Carlo simulation framework designed for multi-accelerator heterogeneous clusters, achieving a 44-fold speedup. They also report that their cluster with 8 Virtex-5 xc5v1x330t FPGAs and 8 Tesla C1060 GPUs provides 19.6 times improved energy efficiency compared to a cluster with 16 AMD Phenom 9650 quad-core 2.4 GHz CPUs for the asset simulation application. The exploration of fixed-point arithmetic for Monte Carlo-based European option pricing on FPGAs by [53] revealed that the error introduced was negligible (comparing 26-bit fixed point implementation to 30-bit implementation), thus enhancing throughput without compromising accuracy.

The energy efficiency and performance superiority of FPGAs were further highlighted by various studies. Ref. [54] presented the first FPGA-based accelerator for option pricing using the Heston model, emphasizing the energy efficiency and performance superiority of FPGAs for advanced Monte Carlo simulations and saving 89% of the energy and providing around twice the speed. Ref. [55] looked into the potential of FPGAs as hardware accelerators for European option pricing using Monte Carlo simulations, showcasing substantial speed-up and energy savings. For a Monte Carlo simulation with $n = 100,000$, the FPGA-based ASP-4P configuration consumes only 171 μJ , compared to 631,878 μJ for a CPU-based system, demonstrating over 3686-times more energy efficiency.

Recent advancements have further solidified the importance of FPGAs in this domain. Ref. [56] introduced CloudiFi, a cloud-native framework that significantly improved end-to-end response time and scalability for Monte-Carlo European Option Pricing workloads. Their early results indicate up to 485× gains in microservice response time. Ref. [57] showcased a design that was 690-times faster than a software implementation when concurrently mapping and customizing computationally-intensive tasks onto an FPGA-based computing cluster.

Other significant contributions include ref. [58]’s high-performance design for interest rate derivative pricing (58× speed-up compared to software implementation on CPU and more than two orders of magnitude more energy efficient), and ref. [59]’s massively parallelized Quasi-Monte Carlo simulation engine (2× to 3× speed-up). They also report their power consumption measurements also show FPGAs to be 336× more energy efficient than CPUs, and 16× more energy efficient than GPUs. Ref. [60] provided a comparative study between FPGA and GPU systems, highlighting FPGA’s superior performance in specific applications. They reported for power consumption of 4× greater efficiency compared to an Intel Xeon 5138 dual-core processor. Ref. [61] proposed a mixed precision methodology resulting in a 170× faster than quad-core implementations on software and also demonstrated energy efficiency improvements by being up to 170 times more energy-efficient compared to quad-core CPU implementations and up to 5.5 times more energy-efficient compared to NVIDIA Tesla C2070 GPU implementations for various Monte Carlo simulation applications. Additionally, ref. [62] introduced a multi-level Monte Carlo FPGA architecture where they showed a reduction of computational complexity compared to single-level methods by up to 50% and also that their design can simulate up to 100 millions of time steps in an asset path simulation with less than 3.6 W. Refs. [63,64] both emphasized the capabilities of reconfigurable hardware in financial computations, showing various speed-ups ranging from 28× to 149× with an 18.6 energy efficiency improvement over a comparable CPU in [63]. Refs. [65,66] showcased significant speed advantages in their respective studies compared to software (800× and 4200×, respectively) and the power consumption of the FPGA design is 2.8 times lower than the software (a 2.5 GHz Xeon E5420) and 6.9 times lower than the GPU design (1.3 GHz Tesla C1060 GPU). Ref. [67] introduced a novel methodology for MLMC simulations showing a 3–9× speed-up on the same platform, and

ref. [68] highlighted the benefits of a high-level synthesis approach showing a speed-up of 3.89× for a Black–Scholes implementation. Ref. [69] introduced a hardware framework for dynamic loading showing a speed-up of between 30× and 120× with up to 30-times greater energy efficiency compared to software-only execution on CPUs, and ref. [70] investigated FPGA-based accelerators using high-level synthesis and showed a 2× speedup compared to a K4200 GPU for a Black–Scholes implementation and a 2× speedup for an Asian option calculation compared to a GTX960 GPU, they also demonstrated significant power efficiency, consuming only 3 watts compared to approximately 100 watts for a GPU. Ref. [71] presented a high-level FPGA design toolflow, and ref. [72] proposed *HyPER*, a modular hybrid system which is 3.4× faster and 36× more power efficient than a CPU implementation for various option pricing calculations. Finally, in [73] the Monte Carlo simulation for option pricing using a Xilinx Virtex-5 achieved a 46.2× speed-up and a 14.4× improvement in energy efficiency compared to a CPU, processing 256 million elements in 3.73 s while consuming 70.31 Joules.

4.3.2. Key Observations

The evolution of Field-Programmable Gate Arrays (FPGAs) has been instrumental in advancing Monte Carlo simulations, particularly within the financial sector. Early research, such as that by [42], laid the foundation by introducing hardware accelerators that achieved significant speedups. This trajectory was furthered by studies like [43,44], which explored the potential of FPGAs in various financial applications.

Option pricing emerged as a prominent area of focus, with works like [45,46] showcasing the capabilities of FPGAs in achieving remarkable accelerations. The versatility of FPGAs was further highlighted by innovations such as the Contessa language by [51] and the dynamic scheduling framework presented by [52].

Recent advancements, such as the CloudiFi framework by [56], emphasize the growing importance of FPGAs in modern cloud-based computational finance. Indeed, the exploration of fixed-point arithmetic for Monte Carlo simulations by [53] and the energy efficiency studies by [54] underscore the adaptability and efficiency of FPGAs.

In conclusion, the collective contributions from various researchers have solidified the role of FPGAs in accelerating Monte Carlo simulations, with each study adding a unique perspective to this evolving domain.

Following the key observations, several distinct trends emerge in the utilization of FPGAs for Monte Carlo simulations and financial computations. The initial stages of research, as seen in studies like [42], were primarily centered around exploiting the computational power of FPGAs. This phase was marked by direct hardware accelerations, aiming for raw speedups.

As the research evolved, there was a clear gravitation towards domain-specific applications, with option pricing emerging as a focal point. This specialization, highlighted by works such as [45,47], went beyond mere computational advantages, introducing innovative methodologies and abstractions tailored for financial computations.

Another discernible trend has been the emphasis on sustainability and adaptability. With the global push towards green computing, studies like [54,55] have underscored the energy efficiency of FPGAs, portraying them as both powerful and environmentally responsible computational tools.

In more recent years, the trend has shifted towards integration, scalability, and user accessibility. Frameworks like CloudiFi, introduced by [56], exemplify the industry's move towards cloud-native FPGA solutions. Concurrently, the exploration of high-level languages and toolflows, as seen in contributions from [51,71], suggests an effort to democratize FPGA-based solutions, making them more approachable for a broader audience. Table 4 provides a summary of the findings from this section.

Table 4. Summary of notable findings from various studies on FPGA implementations of Monte Carlo simulations.

Study	Notable Findings
Zhang et al. (2005) [42]	Hardware accelerator achieving 50× speedup for financial applications.
Anlauf (2005) [43]	12× speedup for Monte Carlo simulations on FPGAs.
Bower et al. (2006) [44]	Modular framework with speedups between 8× to 71×.
Morris et al. (2007) [45]	‘HyperStreams’ abstraction, 146× acceleration for European option pricing.
Tian et al. (2008) [46]	Monte Carlo engine, 340× to 750× speedup.
Tian and Benkrid (2009) [48]	LSMC method for American options, 20× speedup and >20:1 energy savings.
Woods et al. (2008) [49]	QMC derivative pricing, 50× performance over 3 GHz processor.
Tian et al. (2012) [50]	LSMC method, 25× path generation, 18× regression, 20× overall, 54× energy efficiency.
Thomas (2010) [51]	Contessa language, up to 62× speedup, significant power savings.
Anson et al. (2010) [52]	Dynamic scheduling, 44× speedup, 19.6× energy efficiency.
Tian et al. (2010) [53]	Fixed-point arithmetic, enhanced throughput with negligible error.
De Schryver et al. (2011) [54]	Heston model accelerator, 89% energy savings, 2× speedup.
Hegner et al. (2012) [55]	ASP-4P configuration, 3686× energy efficiency.
Diamantopoulos et al. (2021) [56]	CloudiFi framework, up to 485× improvement.
Liu et al. (2010) [57]	690× faster for intensive tasks on FPGA-based cluster.
Tian et al. (2010) [58]	Interest rate derivative pricing, 58× speedup, more energy efficient.
Betkaoui et al. (2010) [60]	Comparative study, FPGA 4× greater power efficiency than Intel Xeon.
Chow et al. (2012) [61]	Mixed precision, 170× speedup, significant energy efficiency improvements.
De Schryver et al. (2013) [62]	Multi-level Monte Carlo, 50% complexity reduction, <3.6 W power.
Jin (2013) [63]	Reconfigurable hardware, 28× to 149× speedup, 18.6× energy efficiency.
De Schryver et al. (2014) [72]	800× speed advantage, 2.8× lower power consumption.
Jong (2014) [66]	4200× speed up over CPU
Omland et al. (2015) [67]	MLMC simulations, 3–9× speed-up.
Choi et al. (2016) [68]	High-level synthesis, 3.89× speedup for Black–Scholes.
Lomuscio et al. (2016) [69]	Dynamic loading, 30× to 120× speedup, up to 30× energy efficiency.
Muslim et al. (2017) [70]	High-level synthesis, 2× speedup, significant power efficiency.
Setetemela et al. (2018) [71]	High-level FPGA design toolflow.
Brugger et al. (2014) [72]	HyPER, 3.4× faster, 36× more power efficient.
Toft et al. (2014) [73]	Monte Carlo simulation on Xilinx Virtex-5, 46.2× speedup, 14.4× energy efficiency.

4.4. Finite Difference

Finite Difference Methods are numerical techniques used to solve differential equations by approximating them with difference equations. In financial modeling, they are commonly applied to price options, particularly for solving the Black–Scholes partial differential equation [74]. These methods involve discretizing the continuous time and asset price into a grid and calculating option prices at each grid point by iterating backward from the option's expiration to the present. The explicit finite difference method updates option prices based on known values from previous time steps, while the implicit method solves a system of linear equations at each time step, offering greater stability for larger time steps.

4.5. Finite Difference on FPGA

A framework for the application of financial explicit finite difference methods on reconfigurable hardware, with a focus on efficient option pricing, is proposed in [75]. The study highlights a 24-fold speed increase using Virtex-6 FPGA technology. However, according to the authors, the potential of this approach on other hardware technologies remains an open question.

Introducing a parallel pipelined architecture, the work in [76] aims to accelerate financial computations using the finite difference method on both reconfigurable hardware and GPUs. This architecture is capable of handling simultaneous option pricing requests with impressive throughput. The design, when implemented on an xc4vlx160 FPGA, achieved over 12 times the speed of a Pentium 4 processor and was 9.4-times more energy-efficient than a GPU. Future directions for this research encompass optimizations on modern hardware cores, synergies between FPGA and GPU, and the exploration of automated domain-specific strategies.

In the study by [77], a novel strategy is put forth to enhance explicit finite difference option pricing by dynamically reconfiguring constants. This approach supports both complete and partial runtime reconfigurations. An analysis on a Virtex-6 XC6VLX760 device showed that partial reconfiguration can yield a speed-up of 4.7 times compared to a static design, while full reconfiguration was found to be less advantageous due to extended reconfiguration durations.

Lastly, the technical report by [78] examines power consumption on various computing platforms, notably desktop computers and FPGA boards, utilizing a Hall effect-based current sensor. The application chosen for this power assessment is a Monte Carlo simulation tailored for European option pricing. The report further mentions that with the combined FPGA and CPU system consumes less than 1/100th of the energy required by the CPU alone.

Key Observations

The explicit finite difference methods, when implemented on such hardware, have shown significant speed-ups, often outperforming traditional CPU-based solutions. Notably, the dynamic reconfiguration of constants, as discussed in some of the studies, offers a flexible approach to optimization, allowing for both full and partial runtime adjustments. This adaptability, however, comes with its own set of challenges, especially when considering the extended durations associated with full reconfigurations. Another observation is the emphasis on energy efficiency. As computational demands grow, the energy consumption of these processes becomes a critical concern. FPGA-based solutions, in many instances, have demonstrated superior energy efficiency compared to GPUs and traditional processors.

The trajectory of research in this domain indicates a growing interest in automating the deployment of financial algorithms on reconfigurable hardware. This trend underscores the importance of ease of implementation and adaptability in rapidly evolving financial markets. Furthermore, there is a noticeable shift towards exploring synergies between different hardware platforms, such as FPGAs and GPUs. Such collaborations could potentially harness the strengths of both platforms, leading to even more optimized solutions. The focus on domain-specific strategies and the fine-tuning of arithmetic types and word

lengths also suggest a move towards more tailored and precise computational solutions. Lastly, as energy consumption becomes a focal point in computational research, we can anticipate further studies benchmarking and optimizing the energy profiles of reconfigurable hardware in financial computations. Table 5 shows the summary of these studies.

Table 5. Summary of notable findings from various studies on FPGA implementations of finite difference methods.

Study	Notable Findings
Jin et al. (2011) [75]	24× speed increase using Virtex-6 FPGA.
Jin et al. (2009) [76]	12× speed of Pentium 4, 9.4× more energy-efficient than GPU on xc4vlx160 FPGA.
Becker et al. (2011) [77]	Dynamic reconfiguration on Virtex-6 XC6VLX760, 4.7× speed-up for partial reconfiguration.
Albicocco et al. (2012) [78]	Combined FPGA and CPU system consuming less than 1/100th of the energy of CPU alone.

4.6. Heston Model

The Heston Model, developed by Steven Heston in 1993 [79], is a widely used stochastic volatility model for pricing options. Unlike the Black–Scholes model, which assumes constant volatility, the Heston Model allows for stochastic (randomly varying) volatility, making it more realistic for financial markets where volatility tends to fluctuate over time. The model is particularly noted for its ability to capture the “volatility smile”, a pattern where implied volatility varies with the strike price and maturity of the option.

4.6.1. Heston Model on FPGA

Using the Heston model as a representative problem-model pairing [80], the efficient exploration of the vast design space is illustrated. The paper presents a detailed methodology for the design and evaluation of optimal hardware accelerators. Through case studies, it introduces a new hardware random number generator for various distributions and a specialized hardware accelerator for computing European barrier option prices. Furthermore, a unique benchmark is proposed for a balanced assessment and comparison of different accelerator implementations in financial mathematics, however at the time of publishing the authors were still working on an FPGA implementation.

In a comparative study between constructing hardware in a Scala Embedded Language (Chisel) and VHDL, the Heston Model was implemented [81]. The results indicated a 30% reduction in code size with Chisel and similar hardware utilization as VHDL. However, certain challenges like the absence of floating point support and complications with XILINX IP cores suggest that the current version of Chisel 2.0 may not be ideal for arithmetic-heavy models like Heston. Nevertheless, Chisel’s potential for future enhancements is evident.

Another study introduced a heterogeneous computing platform integrating GPU and FPGA for power-efficient high-performance computing [82]. This research analyzed various applications, including the Heston Model for option pricing. The findings underscored the importance of aligning computation tasks with the right computing architectures to achieve optimal performance and power efficiency. The reported efficiency of the FPGA compared to the GPU was 1.84×10^9 Ops/Joule.

In a recent paper, the Heston stochastic volatility model, in conjunction with the Longstaff and Schwartz path reduction, was applied for market risk analysis on a Xilinx Alveo U280 FPGA [83]. By implementing efficiency-centric techniques, the algorithm was optimized. The outcomes displayed a notable performance boost on the FPGA compared to two 24-core Intel Xeon Platinum CPUs, with performance improvements ranging from 8 to 185 times. The difference between the initial and the optimized algorithm was a staggering 320 times, underscoring the exceptional computational performance and energy efficiency. They also report a power efficiency of 2.6379×10^9 Ops/Joule.

4.6.2. Key Observations

The consistent mention of the Heston model across studies indicates its significance in financial mathematics and hardware acceleration research. Additionally, the emphasis on power efficiency and performance optimization showcases the industry's drive towards more sustainable and efficient computational solutions.

There is a clear trend towards the exploration of heterogeneous computing platforms, combining the strengths of different architectures like GPU and FPGA. The push for efficiency-driven techniques and the continuous evolution of embedded languages like Chisel also highlight the industry's commitment to innovation and optimization. Table 6 provides a summary of this section.

Table 6. Summary of notable findings from various studies on FPGA implementations of the Heston Model.

Study	Notable Findings
De Schryver et al. (2011) [80]	Detailed methodology for designing and evaluating optimal hardware accelerators for the Heston model; introduced new hardware random number generator and accelerator for European barrier option prices.
Stumm (2013) [81]	Comparative study between Chisel and VHDL for the Heston Model; 30% reduction in code size with Chisel, but challenges with floating point support and XILINX IP cores.
Wu et al. (2014) [82]	Heterogeneous computing platform integrating GPU and FPGA; analyzed Heston Model for option pricing; FPGA 1.84×10^9 Ops/Joule efficiency compared to GPU.
Klaisoongnoen et al. (2022) [83]	Applied Heston model on Xilinx Alveo U280 FPGA; 8 to 185 times performance improvement over two 24-core Intel Xeon Platinum CPUs; 2.6379×10^9 Ops/Joule power efficiency.

4.7. Quadrature Methods

Quadrature Methods are numerical techniques used to evaluate integrals, which can be applied to option pricing in financial models [84]. These methods provide a way to compute the expected payoff of an option by integrating over all possible future paths of the underlying asset's price. They are particularly useful for pricing options with complex features, such as path dependency and early exercise options, where analytical solutions are difficult to obtain.

4.7.1. Quadrature Methods on FPGA

An FPGA-accelerated architecture is introduced for pricing complex options using quadrature methods. Offering a 32.8 times speedup and an 8.3-times increase in power efficiency over a Tesla C1060 GPU, the study by Anson [85] suggests that future endeavors will explore the use of the latest FPGAs, the automation of efficient FPGA and GPU implementations, and the application of hardware designs optimized based on quadrature methods to diverse areas such as electromagnetic problem solutions and photon distribution calculations.

Presenting a framework for automatically generating hardware accelerators for finance applications, Jin [86] specifically addresses European option pricing problems via the quadrature method using a high-level software language (C#). The results indicate that the automatically generated accelerators on an xc4vlx160 FPGA can run over 18-times faster and be up to 143-times more power efficient than a Pentium 4 processor in single precision arithmetic. In double precision arithmetic, these accelerators are 7-times faster and 77-times more power efficient.

A novel parallel architecture for accelerating quadrature methods for pricing multi-dimensional options, such as discrete barrier, Bermudan, and American options, is in-

roduced in another study. Tested on various platforms, including reconfigurable logic and a compute unified device architecture (CUDA)-based graphics processing unit (GPU), the findings [87] show that a 100 MHz Virtex-4 xc4vlx160 FPGA design outperforms an optimized multi-threaded software implementation running on a Xeon W3504 dual-core CPU by being 4.6 times faster and 25.9 times more energy-efficient. Additionally, it is 2.6 times faster and 25.4 times more energy-efficient than a GPU with similar silicon process technology.

Considering the integration grid density and the mantissa size of floating-point operators, research presented in [88] introduces a precision optimization methodology for quadrature computation on reconfigurable hardware. The proposed approach achieves up to 6 times faster performance compared to FPGA designs with double precision arithmetic. Moreover, it surpasses an i7-870 quad-core CPU by up to 15.1 times in speed and 234.9 times in energy efficiency, and a Tesla C2070 GPU by 1.2 times in speed and 42.2 times in energy efficiency.

4.7.2. Key Observations

The consistent mention of quadrature methods as a preferred technique for option pricing highlights its significance in the domain. Power efficiency, alongside speedup, emerges as a critical metric, with multiple studies showcasing the superiority of FPGA implementations over traditional CPUs and GPUs in both these aspects.

There is a clear inclination towards automating the generation of hardware accelerators which leverages high-level software languages for this purpose. This trend indicates a move towards making FPGA implementations more accessible and less reliant on low-level hardware design expertise. Another discernible trend is the exploration of precision optimization methodologies. Such methodologies aim to strike a balance between computational accuracy and performance, optimizing for specific application needs. The consistent comparison against GPUs and CPUs also suggests a competitive environment where FPGAs are continually being benchmarked against other prevalent architectures to demonstrate their advantages. Table 7 summarizes these studies.

Table 7. Summary of notable findings from various studies on FPGA implementations of Quadrature Methods.

Study	Notable Findings
Anson (2009) [85]	FPGA architecture for complex options pricing; 32.8× speedup and 8.3× power efficiency over Tesla C1060 GPU.
Jin (2009) [86]	Automated hardware accelerators for European options via quadrature method; 18× faster and 143× more power efficient (single precision); 7× faster and 77× more power efficient (double precision) than Pentium 4.
Anson (2011) [87]	Parallel architecture for multi-dimensional options; Virtex-4 xc4vlx160 FPGA 4.6× faster and 25.9× more energy-efficient than Xeon W3504 dual-core CPU; 2.6× faster and 25.4× more energy-efficient than a comparable GPU.
Tse (2012) [88]	Precision optimization for quadrature computation; up to 6× faster than FPGA designs with double precision; 15.1× faster and 234.9× more energy-efficient than i7-870 CPU; 1.2× faster and 42.2× more energy-efficient than Tesla C2070 GPU.

4.8. Miscellaneous Models

The hardware implementation for Collateralized Debt Obligations (CDOs) pricing using the One-Factor Gaussian Copula (OFGC) model has been explored and was 63 times faster than a software implementation running on a 3.4 GHz Intel Xeon processor in [89]. Further studies have looked into hardware implementations for both One-Factor and Multi-

Factor Gaussian Copula models, achieving increases between 64 and 71 times compared to corresponding software running on a 3.4 GHz Intel Xeon processor [90,91].

Next we look at papers that look into risk analysis, VaR estimation, and related topics. Efficient FPGA-based methods have been proposed for generating vectors used in modeling correlations between stochastic time-series [92] which was 26 times faster than a quad Opteron 2.6 GHz SMP. Additionally, optimized hardware designs of a Multivariate Gaussian random number generator (MVGRNG) for FPGA platforms have been presented, focusing on financial applications like Value-at-Risk (VaR) estimation [93] and reported an improvement in performance of up to 96% was reported for VaR calculation while up to 81% improvement was observed for option pricing. A suite of hardware accelerators for financial applications, specifically for risk valuation models, has also been demonstrated by [94]. In the previous example, the authors show a for the Black–Scholes and Black-76 accelerators can, respectively, process data up to $\times 348$ and $\times 297$ times faster than the software implementation. They also show a Binomial accelerator, which could not be fully pipelined due to limited resources, but could process data up to $\times 38$ times faster than the software implementation.

The performance and energy consumption of GPUs and FPGAs through the implementation of the Black–Scholes and Heston option pricing models have been explored [95] where they show that energy per computation by using an FPGA can be from 5.9% to 9.8% (depending on the algorithm) as much as that by using an GPU and performance can be from $1.71\times$ to $2.56\times$ as fast as the GPU. Novel methods for pricing multi-dimensional American options on heterogeneous CPU/FPGA systems have been introduced [96,97] showing a $2\times$ speedup compared to state-of-the-art and a significant power saving (Zynq FPGA consumes only 7 kJ, while the low power Atom needs 614 kJ). Furthermore, the implementation of an Asian option pricing application on a high-performance computing (HPC) system has been explored [98] showing between $4\times$ and $9.2\times$ speedup compared to a multi-threaded software implementation and power efficiency improvements of up to 45 times compared to traditional CPU implementations. Tools like OXiGen have been designed for translating high-level code functions into optimized FPGA-based dataflow kernels, with applications in option pricing [99,100] where it demonstrated a $88.1\times$ compared to a single-threaded software implementations. Research has also been conducted on accelerating the computationally-demanding Gaussian Copula Model (GCM) in financial analysis [101,102] where they authors were able to achieve a $4\times$ speedup compared to a software implementation on CPU.

FPGA-accelerated approaches have been introduced for processing market data feeds directly from the network [103] where they showed a $12\times$ improvement over the current (at the time) real-world rate. Other research has highlighted the ability of FPGA IP libraries to significantly reduce (by a factor of 2) latency in electronic trading [104]. They also mention that the Celoxica AMDC accelerator card, which uses a Xilinx Virtex 5 LX110T FPGA device, was measured to draw less than 15 watts of power from its host server. Additionally, FPGA hardware designs have been introduced to accelerate, via a $4\times$ latency reduction in comparison to the conventional software based approach, High Frequency Trading (HFT) applications [105]. An automation framework for Iterative Stencil Loops (ISLs) has also been presented, with potential applications in financial data processing [106]. A novel high-performance computing approach has been introduced for predicting financial market trends, particularly options [107].

Additionally, a high-performance FPGA-based tridiagonal solver, optimized for hardware acceleration, has been introduced for its application in derivatives pricing problems [108] where they showed a $36\times$ speed-up and $16\times$ speed-up for the fixed-point and floating-point designs compared to a CPU implementation and power efficiency of up to 16 times greater than that of a GPU. Special purpose processing elements have been described for optimizing investment strategies in financial securities [109] where the authors demonstrate a speedup of more than 17,000 compared to a high-performance PC. They also report the power consumption of the FPGA (ranging from 12 W to 45 W) is

significantly lower compared to the CPU and GPU used in the study. The CPU, a quad-core Core i7 870, has a thermal design power (TDP) of 95 W, while the GPU, a Tesla C2075, has a TDP of 225 W. Novel optimization methodologies have been presented for reducing hardware resource consumption in stencil-based numerical procedures on reconfigurable hardware [110] where they show a 2× speedup as well. Automation optimization algorithms have been introduced for reconfigurable designs targeting FPGAs [111]. The authors show that an optimized design is over 100× faster and up to 200 times more power efficient than the basic configuration. OpenCL compilation frameworks have been designed for generating high-performance hardware on FPGAs [112,113]. The authors showed a 1.2× speedup compared to a handcrafted implementation. The performance-programmability gap in FPGA programming has been addressed by employing OpenCL in other papers also, e.g., [114] where a 1.22× performance improvement was demonstrated.

Key Observations

The hardware acceleration capabilities of FPGAs have been leveraged in various financial applications, ranging from option pricing to high-frequency trading. Notably, the adaptability of FPGAs allows for fine-tuning and optimization, which is particularly beneficial for financial models that require high precision and low latency. Furthermore, the introduction of tools and frameworks, such as OpenCL compilation frameworks and ARDEGO, have simplified the process of programming and optimizing FPGAs, making them more accessible for financial computations.

The increasing complexity of financial models and the demand for real-time processing have driven the shift towards FPGA-based systems. As observed, there is a growing trend in utilizing FPGA for high-frequency trading, where latency is a critical factor. Additionally, the development of FPGA IP libraries and specific processing elements indicates a move towards more specialized and tailored hardware solutions for financial applications. The exploration of hybrid-core computing and the emphasis on energy efficiency also suggest a broader trend in the financial industry, where both performance and sustainability are becoming paramount. Lastly, the continuous efforts in automating and optimizing FPGA implementations, as seen with tools like OXiGen and Senju, highlight the industry's commitment to harnessing the full potential of FPGA technology in the coming years. Table 8 shows a summary of these miscellaneous methods.

Table 8. Summary of notable findings from various studies on FPGA implementations of miscellaneous models.

Study	Notable Findings
Kaganov et al. (2008) [89]	CDO pricing using OFGC model; 63× faster than software on Intel Xeon.
Kaganov et al. (2011) [91]	One-Factor and Multi-Factor Gaussian Copula models; 64–71× faster than software on Intel Xeon.
Thomas et al. (2008) [92]	Generating vectors for modeling correlations; 26× faster than quad Opteron 2.6 GHz SMP.
Saiprasert et al. (2010) [93]	MVGRNG for VaR estimation; up to 96% improvement in performance.
Stamoulias et al. (2017) [94]	Hardware accelerators for risk valuation; Black–Scholes and Black-76 up to 348× and 297× faster, Binomial up to 38× faster.
Ma et al. (2016) [95]	Black–Scholes and Heston models; FPGA 5.9%–9.8% energy per computation, 1.71–2.56× faster than GPU.
Varela et al. (2015) [96]	Multi-dimensional American options on CPU/FPGA; 2× speedup, Zynq FPGA consumes only 7 kJ vs. 614 kJ for Atom.
Nestorov et al. (2017) [98]	Asian option pricing on HPC; 4–9.2× speedup, 45× power efficiency improvement.

Table 8. *Cont.*

Study	Notable Findings
Peeverelli et al. (2018) [99]	OXiGen tool for FPGA-based kernels; 88.1× speedup over single-threaded software.
Ibraev (2020) [101]	Gaussian Copula Model acceleration; 4× speedup compared to CPU.
Morris et al. (2009) [103]	Market data feed processing; 12× improvement over real-world rate.
Lockwood et al. (2012) [104]	FPGA IP libraries for electronic trading; 2× latency reduction, Celoxica AMDC accelerator card < 15 W power.
Leber et al. (2011) [105]	High Frequency Trading applications; 4× latency reduction vs. software.
Del Sozzo et al. (2023) [106]	Senju automation framework for ISLs; potential applications in financial data processing.
Lázaro García (2013) [107]	HPC approach for predicting market trends; significant speedup for options.
Palmer (2014) [108]	FPGA-based tridiagonal solver; 36× speedup (fixed-point), 16× speedup (floating-point), 16× power efficiency over GPU.
Starke et al. (2012) [109]	Optimizing investment strategies; speedup > 17,000× compared to high-performance PC, FPGA 12–45 W vs. CPU 95 W and GPU 225 W.
Jin et al. (2012) [110]	Optimization for stencil-based numerical procedures; 2× speedup.
Kurek (2014) [111]	Automation optimization algorithms
Czajkowski et al. (2012) [112]	OpenCL compilation frameworks; 1.2× speedup compared to handcrafted implementation.
Krommydas et al. (2016) [114]	OpenCL for FPGA programming; 1.22× performance improvement.

5. Challenges and Future Directions

In this section, we examine the challenges surrounding the use of FPGAs and future directions for development and research in this space.

5.1. Challenges

The development and optimization of FPGA-based systems highlight several challenges in the field. While these systems have consistently shown superiority in many applications, not all implementations guarantee optimal performance. Efficient design is particularly critical, especially in terms of communication times with the FPGA, which remains a complex challenge. Furthermore, as computational demands increase, managing energy consumption becomes increasingly important. Ensuring that FPGAs maintain energy efficiency in all scenarios is a significant ongoing struggle for both researchers and developers. Additionally, despite advances in High-Level Synthesis (HLS) tools, making FPGA platforms universally accessible and user-friendly is still a hurdle, particularly for those without extensive hardware expertise. Moreover, although there is a trend towards heterogeneous computing, achieving integration of FPGAs with other platforms like GPUs without compromising performance continues to be problematic. Lastly, with the growing complexity of financial models, the development of domain-specific strategies and the fine-tuning of arithmetic types and word lengths tailored to specific financial computations present persistent challenges.

5.2. Future Directions

Looking forward, the continuous evolution of FPGA-based architectures is anticipated. Future research could examine innovative areas such as parallel pipelined architectures, systolic hardware designs, and the application of formal mathematical frameworks. The research area for FPGA applications in finance is also expanding, with potential explorations

into multi-asset option pricing and high-frequency trading on the horizon. Further emphasis on user accessibility is critical; future directions might include the development of more intuitive frameworks and high-level languages to make FPGA-based solutions more approachable for a broader audience. Additionally, exploring deeper synergies between different hardware platforms could harness the strengths of FPGAs, GPUs, and other architectures more effectively. Automating the generation of hardware accelerators is another promising area, potentially making FPGA implementations more accessible and reducing the reliance on low-level hardware design expertise. As financial models become increasingly complex, there is a growing need for specialized FPGA IP libraries and specific processing elements tailored for financial applications. Finally, the exploration of hybrid-core computing and a continued emphasis on energy efficiency could play pivotal roles in shaping the future of FPGA applications in finance.

6. Conclusions

The integration of Field Programmable Gate Arrays (FPGAs) into financial computation has proven to be highly effective, particularly in the field of option pricing, where their ability to accelerate complex models has been extensively documented. Our comprehensive review of 99 studies highlights significant speed and efficiency gains achievable with FPGAs, which often exceed those of traditional computing solutions by several orders of magnitude. For example, FPGAs can deliver performance gains ranging from 270 to 5400 times faster than conventional CPU implementations depending on the specific option pricing model employed.

Despite these advances, several challenges hinder the universal adoption of FPGAs. Key among these are the difficulties of FPGA programming and the need for optimized design frameworks that enhance both power efficiency and ease of integration with existing computing infrastructures. Also, adoption trends for FPGAs, GPUs, and CPUs reveal a sharp contrast: while GPUs have become the standard for high-performance computing (HPC) due to their massively parallel architectures and have seen significant adoption in both general-purpose and AI-specific tasks, FPGAs have not kept pace with this growth. Despite their potential for specific applications with irregular data types and high energy efficiency, FPGAs face challenges such as low memory bandwidth, complex programming requirements, and high development times, which limit their adoption compared to GPUs and CPUs [115]. Future research should thus focus on simplifying the programming environment and enhancing the design tools that facilitate the broader application of FPGAs in financial systems. The importance of these improvements is underscored by the fact that FPGAs have demonstrated energy efficiency improvements of up to 30% more floating-point operations per Joule compared to traditional systems.

Looking ahead, the path for FPGAs in finance appears strong, with substantial opportunities for growth in areas such as high-frequency trading and risk analysis. Addressing the current technological constraints will be crucial in maintaining the relevance of FPGAs in an industry characterized by rapid innovation and escalating performance demands. The ongoing collaboration among academics, developers, and financial experts is essential to harness the full capabilities of FPGA technology, ensuring it continues to be a transformative force in financial computation.

Author Contributions: Conceptualization, A.O.M.; methodology, A.O.M.; software, A.O.M.; validation, E.P., B.H. and A.O.M.; formal analysis, A.O.M.; investigation, A.O.M.; resources, E.P.; data curation, A.O.M.; writing—original draft preparation, A.O.M.; writing—review and editing, E.P. and B.H.; visualization, A.O.M.; supervision, E.P. and B.H.; project administration, A.O.M. All authors have read and agreed to the published version of the manuscript.

Funding: This paper has been supported in part by Dell Technologies, Intel Programmable Solutions Group, Science Foundation Ireland under grant 07/MI/008, and the SFI INSIGHT Centre for Data Analytics under grant 12-RC-2289-P2.

Data Availability Statement: Dataset available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

- Gandhare, S.; Karthikeyan, B. Survey on FPGA Architecture and Recent Applications. In Proceedings of the 2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 30–31 March 2019; pp. 1–4. [CrossRef]
- Inggs, G.; Fleming, S.; Thomas, D.B.; Luk, W. Is High Level Synthesis Ready for Business? An Option Pricing Case Study. In *FPGA Based Accelerators for Financial Applications*; De Schryver, C., Ed.; Springer International Publishing: Cham, Switzerland, 2015; pp. 97–115. [CrossRef]
- Kitchenham, B.; Brereton, O.P.; Budgen, D.; Turner, M.; Bailey, J.; Linkman, S. Systematic literature reviews in software engineering—a systematic literature review. *Inf. Softw. Technol.* **2009**, *51*, 7–15. [CrossRef]
- Budgen, D.; Brereton, P. Performing systematic literature reviews in software engineering. In Proceedings of the 28th International Conference on Software Engineering, Shanghai, China, 20–28 May 2006; pp. 1051–1052.
- Mahony, A.O.; Popovici, E. A systematic review of blockchain hardware acceleration architectures. In Proceedings of the 2019 30th Irish Signals and Systems Conference (ISSC), Maynooth, Ireland, 17–18 June 2019; pp. 1–6.
- Zhang, C.; Yu, H.; Zhou, Y.; Jiang, H. High-Performance and Energy-Efficient FPGA-GPU-CPU Heterogeneous System Implementation. In *Advances in Parallel & Distributed Processing, and Applications*; Arabnia, H.R., Deligiannidis, L., Grimaila, M.R., Hodson, D.D., Joe, K., Sekijima, M., Tinetti, F.G., Eds.; Springer: Cham, Switzerland, 2021; pp. 477–492.
- Mahony, A.O.; Zeidan, G.; Hanzon, B.; Popovici, E. A parallel and pipelined implementation of a pascal-simplex based multi-asset option pricer on FPGA using OpenCL. *Microprocess. Microsyst.* **2022**, *90*, 104508. [CrossRef]
- Monteiro, A.M.; Santos, A.A. Parallel computing in finance for estimating risk-neutral densities through option prices. *J. Parallel Distrib. Comput.* **2023**, *173*, 61–69. [CrossRef]
- Panova, E.; Volokitin, V.; Gorshkov, A.; Meyerov, I. Black-Scholes Option Pricing on Intel CPUs and GPUs: Implementation on SYCL and Optimization Techniques. In Proceedings of the Supercomputing: 8th Russian Supercomputing Days, RuSCDays 2022, Moscow, Russia, 26–27 September 2022; Revised Selected Papers; Springer: Cham, Switzerland, 2022; pp. 48–62.
- Mahony, A.O.; Zeidan, G.; Hanzon, B.; Popovici, E. A parallel and pipelined implementation of a pascal-simplex based two asset option pricer on FPGA using openCL. In Proceedings of the 2020 IEEE Nordic Circuits and Systems Conference (NorCAS), Virtual, 27–28 October 2020; pp. 1–6.
- Bruce, R.; Setoain, J.; Chamberlain, R.; Devlin, M.; Badia, R.M. Implementing Closed-Form Expressions on FPGAs Using the NAL, with Comparison to CUDA GPU and Cell BE Implementations. 2008. Available online: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1baa510cce4a140fd3d9025ad2d7935b49e64e27> (accessed on 13 January 2024).
- Xiong, C.; Xu, N. Performance Comparison of BLAS on CPU, GPU and FPGA. In Proceedings of the 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 11–13 December 2020; Volume 9, pp. 193–197. [CrossRef]
- Elshakhs, Y.S.; Deliparaschos, K.M.; Charalambous, T.; Oliva, G.; Zolotas, A. A Comprehensive Survey on Delaunay Triangulation: Applications, Algorithms, and Implementations Over CPUs, GPUs, and FPGAs. *IEEE Access* **2024**, *12*, 12562–12585. [CrossRef]
- Nurvitadhi, E.; Sheffield, D.; Sim, J.; Mishra, A.; Venkatesh, G.; Marr, D. Accelerating binarized neural networks: Comparison of FPGA, CPU, GPU, and ASIC. In Proceedings of the 2016 International Conference on Field-Programmable Technology (FPT), Xi'an, China, 7–9 December 2016; pp. 77–84.
- Black, F.; Scholes, M. The pricing of options and corporate liabilities. *J. Political Econ.* **1973**, *81*, 637–654. [CrossRef]
- Merton, R.C. Theory of rational option pricing. *Bell J. Econ. Manag. Sci.* **1973**, *4*, 141–183. [CrossRef]
- Tandon, S. A Programmable Architecture for Real-Time Derivative Trading. Master's Thesis, University of Edinburgh, Edinburgh, UK, 2003.
- Baxter, R.; Booth, S.; Bull, M.; Cawood, G.; Perry, J.; Parsons, M.; Simpson, A.; Trew, A.; McCormick, A.; Smart, G.; et al. Maxwell-a 64 FPGA supercomputer. In Proceedings of the Second NASA/ESA Conference on Adaptive Hardware and Systems (AHS 2007), Edinburgh, Scotland, 5–8 August 2007; pp. 287–294.
- Thomas, D.B.; Luk, W. Sampling from the multivariate Gaussian distribution using reconfigurable hardware. In Proceedings of the 15th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM 2007), Napa, CA, USA, 23–25 April 2007; pp. 3–12.
- Castillo, J.; Bosque, J.L.; Castillo, E.; Huerta, P.; Martínez, J.I. Hardware accelerated montecarlo financial simulation over low cost fpga cluster. In Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing, Rome, Italy, 23–29 May 2009; pp. 1–8.
- Patel, C.; Srikanth, M.; Kumar, K.C.; Sivanantham, S. Monte-Carlo Black-Scholes Implementation using OpenCL Standard. *Indian J. Sci. Technol.* **2016**, *8*, 1–5. [CrossRef]
- Choo, C.; Malhotra, L.; Munjal, A. FPGA-Based Design of Black Scholes Financial Model for High Performance Trading. *J. Inf. Commun. Converg. Eng.* **2013**, *11*, 190–198. [CrossRef]
- Guerrero, M.A.B. MultiCorePricer: A Monte-Carlo Pricing Engine for Financial Derivatives. Ph.D. Thesis, Swiss Federal Institute of Technology Zurich, Zurich, Switzerland, 2015.

24. Pham, N.K.; Aung, K.M.M.; Kumar, A. Automatic framework to generate reconfigurable accelerators for option pricing applications. In Proceedings of the 2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig), Cancun, Mexico, 30 November–2 December 2016; pp. 1–8.
25. Inggs, G. Algorithmic Trading: A brief, computational finance case study on data centre FPGAs. *arXiv* **2016**, arXiv:1607.05069.
26. Ma, L. Low Power and High Performance Heterogeneous Computing on FPGAs. Ph.D. Thesis, Politecnico di Torino, Torino, Italy, 2019.
27. de Castro, M.C.S. Implementação e Avaliação de Co-Processadores para BlackScholes em FPGA. *Cad. IME-Série Inform.* **2019**, *42*, 7.
28. Rodrigues, A.; Moreira, V. MultiCorePricer: A Monte-Carlo Pricing Engine for Financial Derivatives 2. In *Topics in Computational Finance*; Volume number. 2019/2, Operations Management and Research and Decision Sciences Book Series 2019; University of Madeira: Funchal, Portugal, 2019; p. 27.
29. Li, Y.; Zhang, L.; Dai, Y.; Sun, Y. A Word-length Optimized Parallel Framework for Accelerating Option Pricing Model. In Proceedings of the 2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), Haikou, China, 20–22 December 2021; pp. 654–661.
30. Wang, S.; Huan, H.; Wong, S.F.; Yen, J. FPGA based Implied Volatility Calculation with Multi-section Method. In Proceedings of the 2022 IEEE 20th International Conference on Industrial Informatics (INDIN), Perth, Australia, 25–28 July 2022; pp. 507–514.
31. Cox, J.C.; Ross, S.A.; Rubinstein, M. Option pricing: A simplified approach. *J. Financ. Econ.* **1979**, *7*, 229–263. [\[CrossRef\]](#)
32. Morales, V.M.; Horrein, P.; Baghdadi, A.; Hochapfel, E.; Vaton, S. Energy-efficient FPGA implementation for binomial option pricing using OpenCL. In Proceedings of the 2014 Design, Automation Test in Europe Conference Exhibition (DATE), Dresden, Germany, 24–28 March 2014; pp. 1–6. [\[CrossRef\]](#)
33. Jin, Q.; Thomas, D.B.; Luk, W.; Cope, B. Exploring reconfigurable architectures for binomial-tree pricing models. In Proceedings of the Reconfigurable Computing: Architectures, Tools and Applications, International Workshop on Applied Reconfigurable Computing, London, UK, 26–28 March 2008; pp. 245–255.
34. Wynnyk, C.; Magdon-Ismail, M. Pricing the american option using reconfigurable hardware. In Proceedings of the 2009 International Conference on Computational Science and Engineering, Vancouver, BC, Canada, 29–31 August 2009; Volume 2, pp. 532–536.
35. Jin, Q.; Thomas, D.B.; Luk, W.; Cope, B. Exploring reconfigurable architectures for tree-based option pricing models. *ACM Trans. Reconfigurable Technol. Syst. (TRETs)* **2009**, *2*, 21. [\[CrossRef\]](#)
36. Chatziparaskevas, G.; Brokalakis, A.; Papaefstathiou, I. An FPGA-based parallel processor for Black-Scholes option pricing using finite differences schemes. In Proceedings of the Conference on Design, Automation and Test in Europe, Dresden, Germany, 12–16 March 2012; pp. 709–714.
37. Fabry, P.; Thomas, D. Efficient reconfigurable architecture for pricing exotic options. *ACM Trans. Reconfig. Technol. Syst. (TRETs)* **2017**, *10*, 29. [\[CrossRef\]](#)
38. Tavakkoli, A.; Thomas, D.B. Low-latency option pricing using systolic binomial trees. In Proceedings of the 2014 International Conference on Field-Programmable Technology (FPT), Shanghai, China, 10–12 December 2014; pp. 44–51.
39. Tavakkoli, A.; Thomas, D.B. A high-level design framework for the automatic generation of high-throughput systolic binomial-tree solvers. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2017**, *26*, 341–354. [\[CrossRef\]](#)
40. Minhas, U.I.; Woods, R.; Karakostas, G. Exploring functional acceleration of OpenCL on FPGAs and GPUs through platform-independent optimizations. In *Applied Reconfigurable Computing. Architectures, Tools, and Applications, Proceedings of the International Symposium on Applied Reconfigurable Computing, Santorini, Greece, 2–4 May 2018*; Springer: Cham, Switzerland, 2018; pp. 551–563.
41. Boyle, P.P. Options: A monte carlo approach. *J. Financ. Econ.* **1977**, *4*, 323–338. [\[CrossRef\]](#)
42. Zhang, G.; Leong, P.H.W.; Ho, C.H.; Tsoi, K.H.; Cheung, C.C.; Lee, D.U.; Cheung, R.C.; Luk, W. Reconfigurable acceleration for Monte Carlo based financial simulation. In Proceedings of the 2005 IEEE International Conference on Field-Programmable Technology, Singapore, 11–14 December 2005; pp. 215–222.
43. Anlauf, J.K. *Pricing of Derivatives by Fast, Hardware-Based Monte-Carlo Simulation*; Technical Report, Working Paper; Institute for Information, University of Bonn: Bonn, Germany, 2005.
44. Bower, J.A.; Thomas, D.B.; Luk, W.; Mencer, O. A reconfigurable simulation framework for financial computation. In Proceedings of the 2006 IEEE International Conference on Reconfigurable Computing and FPGAs (ReConFig 2006), San Luis Potosi, Mexico, 27–29 September 2006; pp. 1–9.
45. Morris, G.W.; Aubury, M. Design space exploration of the European option benchmark using hyperstreams. In Proceedings of the 2007 International Conference on Field Programmable Logic and Applications, Amsterdam, The Netherlands, 27–29 August 2007; pp. 5–10.
46. Tian, X.; Benkrid, K.; Gu, X. High performance Monte-Carlo based option pricing on FPGAs. *Eng. Lett.* **2008**, *16*, 3.
47. Tian, X.; Benkrid, K. Design and implementation of a high performance financial Monte-Carlo simulation engine on an FPGA supercomputer. In Proceedings of the 2008 International Conference on Field-Programmable Technology, Taipei, Taiwan, 7–10 December 2008; pp. 81–88.

48. Tian, X.; Benkrid, K. American option pricing on reconfigurable hardware using Least-Squares Monte Carlo method. In Proceedings of the 2009 International Conference on Field-Programmable Technology, Sydney, Australia, 9–11 December 2009; pp. 263–270. [\[CrossRef\]](#)
49. Woods, N.A.; VanCourt, T. FPGA acceleration of quasi-Monte Carlo in finance. In Proceedings of the 2008 International Conference on Field Programmable Logic and Applications, Heidelberg, Germany, 8–10 September 2008; pp. 335–340.
50. Tian, X.; Benkrid, K. Implementation of the longstaff and schwartz american option pricing model on fpga. *J. Signal Process. Syst.* **2012**, *67*, 79–91. [\[CrossRef\]](#)
51. Thomas, D.B. Acceleration of financial monte-carlo simulations using fpgas. In Proceedings of the 2010 IEEE Workshop on High Performance Computational Finance, New Orleans, LA, USA, 14 November 2010; pp. 1–6.
52. Anson, H.; Thomas, D.B.; Tsoi, K.H.; Luk, W. Dynamic scheduling Monte-Carlo framework for multi-accelerator heterogeneous clusters. In Proceedings of the 2010 International Conference on Field-Programmable Technology, Beijing, China, 8–10 December 2010; pp. 233–240.
53. Tian, X.; Benkrid, K. Fixed-point arithmetic error estimation in Monte-Carlo simulations. In Proceedings of the 2010 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 13–15 December 2010; pp. 202–207.
54. De Schryver, C.; Shcherbakov, I.; Kienle, F.; Wehn, N.; Marxen, H.; Kostiuk, A.; Korn, R. An energy efficient FPGA accelerator for monte carlo option pricing with the heston model. In Proceedings of the 2011 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 30 November–2 December 2011; pp. 468–474.
55. Hegner, J.S.; Sindholt, J.; Nannarelli, A. Design of power efficient fpga based hardware accelerators for financial applications. In Proceedings of the NORCHIP 2012, Copenhagen, Denmark, 12–13 November 2012; pp. 1–4.
56. Diamantopoulos, D.; Polig, R.; Ringlein, B.; Purandare, M.; Weiss, B.; Hagleitner, C.; Lantz, M.; Abel, F. Acceleration-as-a- μ Service: A Cloud-native Monte-Carlo Option Pricing Engine on CPUs, GPUs and Disaggregated FPGAs. In Proceedings of the 2021 IEEE 14th International Conference on Cloud Computing (CLOUD), Chicago, IL, USA, 5–10 September 2021; pp. 726–729.
57. Liu, Q.; Todman, T.; Tsoi, K.H.; Luk, W. Convex models for accelerating applications on FPGA-based clusters. In Proceedings of the 2010 International Conference on Field-Programmable Technology, Beijing, China, 8–10 December 2010; pp. 495–498.
58. Tian, X.; Benkrid, K. Libor market model simulation on an FPGA parallel machine. In Proceedings of the 2010 VI Southern Programmable Logic Conference (SPL), Pernambuco, Brazil, 24–26 March 2010; pp. 9–14.
59. Tian, X.; Benkrid, K. High-performance quasi-Monte Carlo financial simulation: FPGA vs. GPP vs. GPU. *ACM Trans. Reconfig. Technol. Syst. (TRETs)* **2010**, *3*, 26. [\[CrossRef\]](#)
60. Betkaoui, B.; Thomas, D.B.; Luk, W. Comparing performance and energy efficiency of FPGAs and GPUs for high productivity computing. In Proceedings of the 2010 International Conference on Field-Programmable Technology, Beijing, China, 8–10 December 2010; pp. 94–101.
61. Chow, G.C.T.; Tse, A.H.T.; Jin, Q.; Luk, W.; Leong, P.H.; Thomas, D.B. A mixed precision Monte Carlo methodology for reconfigurable accelerator systems. In Proceedings of the ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, 22–24 February 2012; pp. 57–66.
62. de Schryver, C.; Torruella, P.; Wehn, N. A multi-level Monte Carlo FPGA accelerator for option pricing in the Heston model. In Proceedings of the 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE), Melbourne, Australia, 18–22 March 2013; pp. 248–253.
63. Jin, Q. Optimising Financial Computation for Reconfigurable Hardware. Ph.D. Thesis, Imperial College London, London, UK, 2013.
64. Sanchez-Roman, D.; Moreno, V.; Lopez-Buedo, S.; Sutter, G.; Gonzalez, I.; Gomez-Arribas, F.J.; Aracil, J. FPGA acceleration using high-level languages of a Monte-Carlo method for pricing complex options. *J. Syst. Archit.* **2013**, *59*, 135–143. [\[CrossRef\]](#)
65. De Jong, M.; Sima, V.M.; Bertels, K.; Thomas, D. FPGA-accelerated Monte-Carlo integration using stratified sampling and Brownian bridges. In Proceedings of the 2014 International Conference on Field-Programmable Technology (FPT), Shanghai, China, 10–12 December 2014; pp. 68–75.
66. de Jong, M. Hardware Acceleration of Monte-Carlo Integration in Finance. Master's Thesis, TU Delft, Delft, The Netherlands, 2014.
67. Omland, S.; Hefter, M.; Ritter, K.; Brugger, C.; De Schryver, C.; Wehn, N.; Kostiuk, A. Exploiting Mixed-Precision Arithmetics in a Multilevel Monte Carlo Approach on FPGAs. In *FPGA Based Accelerators for Financial Applications*; De Schryver, C., Ed.; Springer International Publishing: Cham, Switzerland, 2015; pp. 191–220. [\[CrossRef\]](#)
68. Choi, J.; Lian, R.L.; Brown, S.; Anderson, J. A unified software approach to specify pipeline and spatial parallelism in FPGA hardware. In Proceedings of the 2016 IEEE 27th International Conference on Application-Specific Systems, Architectures and Processors (ASAP), London, UK, 6–8 July 2016; pp. 75–82. [\[CrossRef\]](#)
69. Lomuscio, A.; Nannarelli, A.; Re, M. FPGA Acceleration by Dynamically-Loaded Hardware Libraries. *Energy* **2016**, *1*, 3.
70. Muslim, F.B.; Ma, L.; Roozmeh, M.; Lavagno, L. Efficient FPGA implementation of OpenCL high-performance computing applications via high-level synthesis. *IEEE Access* **2017**, *5*, 2747–2762. [\[CrossRef\]](#)
71. Setetemela, K.; Winberg, S. Systematic design of an ideal toolflow for accelerating big data applications on FPGA platforms. In Proceedings of the 2018 IEEE 9th International Conference on Mechanical and Intelligent Manufacturing Technologies (ICMIMT), Cape Town, South Africa, 10–13 February 2018; pp. 202–206.
72. Brugger, C.; de Schryver, C.; Wehn, N. HyPER: A runtime reconfigurable architecture for monte carlo option pricing in the Heston model. In Proceedings of the 2014 24th International Conference on Field Programmable Logic and Applications (FPL), Munich, Germany, 2–4 September 2014; pp. 1–8. [\[CrossRef\]](#)

73. Toft, J.K.; Nannarelli, A. Energy efficient fpga based hardware accelerators for financial applications. In Proceedings of the 2014 NORCHIP, Tampere, Finland, 27–28 October 2014; pp. 1–6.
74. Schwartz, E.S. The valuation of warrants: Implementing a new approach. *J. Financ. Econ.* **1977**, *4*, 79–93. [[CrossRef](#)]
75. Jin, Q.; Luk, W.; Thomas, D.B. Unifying finite difference option-pricing for hardware acceleration. In Proceedings of the 2011 21st International Conference on Field Programmable Logic and Applications, Chania, Greece, 5–7 September 2011; pp. 6–9.
76. Jin, Q.; Thomas, D.B.; Luk, W. Exploring reconfigurable architectures for explicit finite difference option pricing models. In Proceedings of the 2009 International Conference on Field Programmable Logic and Applications, Prague, Czech Republic, 31 August–2 September 2009; pp. 73–78.
77. Becker, T.; Jin, Q.; Luk, W.; Weston, S. Dynamic constant reconfiguration for explicit finite difference option pricing. In Proceedings of the 2011 International Conference on Reconfigurable Computing and FPGAs, Cancun, Mexico, 30 November–2 December 2011; pp. 176–181.
78. Albicocco, P.; Papini, D.; Nannarelli, A. *Direct Measurement of Power Dissipated by Monte Carlo Simulations on CPU and FPGA Platforms*; IMM Technical Report 2012-18; Technical University of Denmark: Kongens Lyngby, Denmark, 2012.
79. Heston, S.L. A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Rev. Financ. Stud.* **1993**, *6*, 327–343. [[CrossRef](#)]
80. de Schryver, C.; Jung, M.; Wehn, N.; Marxen, H.; Kostiuik, A.; Korn, R. Energy efficient acceleration and evaluation of financial computations towards real-time pricing. In Proceedings of the Knowledge-Based and Intelligent Information and Engineering Systems: 15th International Conference, KES 2011, Kaiserslautern, Germany, 12–14 September 2011; Proceedings, Part IV; pp. 177–186.
81. Stumm, C. *Investigate the Hardware Description Language Chisel—A Case Study Implementing the Heston Model*; Technische Universität Kaiserslautern: Kaiserslautern, Germany, 2013.
82. Wu, Q.; Ha, Y.; Kumar, A.; Luo, S.; Li, A.; Mohamed, S. A heterogeneous platform with GPU and FPGA for power efficient high performance computing. In Proceedings of the 2014 International Symposium on Integrated Circuits (ISIC), Singapore, 10–12 December 2014; pp. 220–223.
83. Klaisoongnoen, M.; Brown, N.; Brown, O.T. Low-power option Greeks: Efficiency-driven market risk analysis using FPGAs. In Proceedings of the International Symposium on Highly-Efficient Accelerators and Reconfigurable Technologies, Tsukuba, Japan, 9–10 June 2022; pp. 95–101.
84. Andricopoulos, A.D.; Widdicks, M.; Duck, P.W.; Newton, D.P. Universal option valuation using quadrature methods. *J. Financ. Econ.* **2003**, *67*, 447–471. [[CrossRef](#)]
85. Anson, H.; Thomas, D.B.; Luk, W. Accelerating quadrature methods for option valuation. In Proceedings of the 2009 17th IEEE Symposium on Field Programmable Custom Computing Machines, Napa, CA, USA, 5–7 April 2009; pp. 29–36.
86. Jin, Q.; Thomas, D.B.; Luk, W. Automated application acceleration using software to hardware transformation. In Proceedings of the 2009 International Conference on Field-Programmable Technology, Sydney, Australia, 9–11 December 2009; pp. 411–414.
87. Anson, H.; Thomas, D.; Luk, W. Design exploration of quadrature methods in option pricing. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2011**, *20*, 818–826.
88. Tse, A.H.; Chow, G.C.; Jin, Q.; Thomas, D.B.; Luk, W. Optimising performance of quadrature methods with reduced precision. In Proceedings of the Reconfigurable Computing: Architectures, Tools and Applications: 8th International Symposium, ARC 2012, Hong Kong, China, 19–23 March 2012; Proceedings 8; pp. 251–263.
89. Kaganov, A.; Chow, P.; Lakhany, A. FPGA acceleration of Monte-Carlo based credit derivative pricing. In Proceedings of the 2008 International Conference on Field Programmable Logic and Applications, Heidelberg, Germany, 8–10 September 2008; pp. 329–334.
90. Kaganov, A. *Hardware Acceleration of Monte-Carlo Structural Financial Instrument Pricing Using a Gaussian Copula Model*; University of Toronto: Toronto, ON, Canada, 2008.
91. Kaganov, A.; Lakhany, A.; Chow, P. FPGA acceleration of multifactor CDO pricing. *ACM Trans. Reconfig. Technol. Syst. (TRETs)* **2011**, *4*, 20. [[CrossRef](#)]
92. Thomas, D.B.; Luk, W. Multivariate Gaussian random number generation targeting reconfigurable hardware. *ACM Trans. Reconfig. Technol. Syst. (TRETs)* **2008**, *1*, 12. [[CrossRef](#)]
93. Saiprasert, C.; Bouganis, C.S.; Constantinides, G.A. Design of a financial application driven multivariate gaussian random number generator for an FPGA. In Proceedings of the Reconfigurable Computing: Architectures, Tools and Applications: 6th International Symposium, ARC 2010, Bangkok, Thailand, 17–19 March 2010; Proceedings 6; pp. 182–193.
94. Stamoulias, I.; Kachris, C.; Soudris, D. Hardware accelerators for financial applications in HDL and High Level Synthesis. In Proceedings of the 2017 International Conference on Embedded Computer Systems: Architectures, Modeling, and Simulation (SAMOS), Samos, Greece, 16–20 July 2017; pp. 278–285.
95. Ma, L.; Muslim, F.B.; Lavagno, L. High performance and low power Monte Carlo methods to option pricing models via high level design and synthesis. In Proceedings of the 2016 European Modelling Symposium (EMS), Pisa, Italy, 28–30 November 2016; pp. 157–162.
96. Varela, J.A.; Brugger, C.; de Schryver, C.; Wehn, N.; Tang, S.; Omland, S. Exploiting the brownian bridge technique to improve longstaff-schwartz american option pricing on FPGA systems. In Proceedings of the 2015 International Conference on ReConFigurable Computing and FPGAs (ReConFig), Riviera Maya, Mexico, 7–9 December 2015; pp. 1–6.

97. Varela, J.A.; Brugger, C.; Tang, S.; Wehn, N.; Korn, R. Pricing High-Dimensional American Options on Hybrid CPU/FPGA Systems. In *FPGA Based Accelerators for Financial Applications*; De Schryver, C., Ed.; Springer International Publishing: Cham, Switzerland, 2015; pp. 143–166. [\[CrossRef\]](#)
98. Nestorov, A.M.; Reggiani, E.; Palikareva, H.; Burovskiy, P.; Becker, T.; Santambrogio, M.D. A scalable dataflow implementation of curran's approximation algorithm. In Proceedings of the 2017 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Orlando, FL, USA, 29 May–2 June 2017; pp. 150–157.
99. Peverelli, F.; Rabozzi, M.; Del Sozzo, E.; Santambrogio, M.D. OXiGen: A tool for automatic acceleration of c functions into dataflow FPGA-based kernels. In Proceedings of the 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), Vancouver, BC, Canada, 21–25 May 2018; pp. 91–98.
100. Peverelli, F.; Rabozzi, M.; Cardamone, S.; Del Sozzo, E.; Thom, A.J.; Santambrogio, M.D.; Di Tucci, L. Automated acceleration of dataflow-oriented c applications on FPGA-based systems. In Proceedings of the 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), San Diego, CA, USA, 28 April–1 May 2019; p. 313.
101. Ibraev, S. Acceleration of the Strike Calculation for Foreign Exchange Options Using FPGA. 2020. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3540371 (accessed on 17 January 2024).
102. Ibraev, S.; Deng, M. FPGA Application to Option Price Inversion. 2022. Available online: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4124011 (accessed on 21 January 2024).
103. Morris, G.W.; Thomas, D.B.; Luk, W. FPGA accelerated low-latency market data feed processing. In Proceedings of the 2009 17th IEEE Symposium on High Performance Interconnects, New York, NY, USA, 25–27 August 2009; pp. 83–89.
104. Lockwood, J.W.; Gupte, A.; Mehta, N.; Blott, M.; English, T.; Vissers, K. A low-latency library in FPGA hardware for high-frequency trading (HFT). In Proceedings of the 2012 IEEE 20th Annual Symposium on High-Performance Interconnects, Santa Clara, CA, USA, 22–24 August 2012; pp. 9–16.
105. Leber, C.; Geib, B.; Litz, H. High frequency trading acceleration using FPGAs. In Proceedings of the 2011 21st International Conference on Field Programmable Logic and Applications, Chania, Greece, 5–7 September 2011; pp. 317–322.
106. Del Sozzo, E.; Conficconi, D.; Santambrogio, M.D.; Sano, K. Senju: A Framework for the Design of Highly Parallel FPGA-based Iterative Stencil Loop Accelerators. In Proceedings of the 2023 ACM/SIGDA International Symposium on Field Programmable Gate Arrays, Monterey, CA, USA, 12–14 February 2023; p. 233.
107. Lázaro García, H. Design and Implementation of a Multi-Level Monte Carlo Accelerator for Option Pricing on the Zynq-7000 EPP. Bachelor Thesis, Carlos III University of Madrid, Madrid, Spain, 2013.
108. Palmer, S. Accelerating Implicit Finite Difference Schemes Using a Hardware Optimized Tridiagonal Solver for FPGAs. *arXiv* **2014**, arXiv:1402.5094.
109. Starke, C.; Grossmann, V.; Wienbrandt, L.; Schimmmler, M. An FPGA implementation of an investment strategy processor. *Procedia Comput. Sci.* **2012**, *9*, 1880–1889. [\[CrossRef\]](#)
110. Jin, Q.; Becker, T.; Luk, W.; Thomas, D. Optimising explicit finite difference option pricing for dynamic constant reconfiguration. In Proceedings of the 22nd International Conference on Field Programmable Logic and Applications (FPL), Oslo, Norway, 29–31 August 2012; pp. 165–172.
111. Kurek, M.; Becker, T.; Chau, T.C.; Luk, W. Automating Optimization of Reconfigurable Designs. In Proceedings of the 2014 IEEE 22nd Annual International Symposium on Field-Programmable Custom Computing Machines, Boston, MA, USA, 11–13 May 2014; pp. 210–213. [\[CrossRef\]](#)
112. Czajkowski, T.S.; Aydonat, U.; Denisenko, D.; Freeman, J.; Kinsner, M.; Neto, D.; Wong, J.; Yiannacouras, P.; Singh, D.P. From OpenCL to high-performance hardware on FPGAs. In Proceedings of the 22nd International Conference on Field Programmable Logic and Applications (FPL), Oslo, Norway, 29–31 August 2012; pp. 531–534.
113. Czajkowski, T.S.; Neto, D.; Kinsner, M.; Aydonat, U.; Wong, J.; Denisenko, D.; Yiannacouras, P.; Freeman, J.; Singh, D.P.; Brown, S.D. OpenCL for FPGAs: Prototyping a compiler. In Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA), Las Vegas, NV, USA, 16–19 July 2012; p. 1.
114. Krommydas, K.; Helal, A.E.; Verma, A.; Feng, W.C. *Bridging the Performance-Programmability Gap for FPGAs Via Opencl: A Case Study with Opencwarfs*; Technical Report; Department of Computer Science, Virginia Polytechnic Institute & State University: Blacksburg, VA, USA, 2016.
115. de Castro, M.; Vilariño, D.L.; Torres, Y.; Llanos, D.R. The Role of Field-Programmable Gate Arrays in the Acceleration of Modern High-Performance Computing Workloads. *Computer* **2024**, *57*, 66–76. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.