American Multinomial Option Pricing on FPGA using OneAPI

Aidan O Mahony
Department of Electrical and Electronic Engineering
University College Cork
Cork, Ireland
103837793@umail.ucc.ie

Gil Zeidan
Programmable Solutions Group
Intel Corporation
Massachusetts
USA
gilbert.zeidan@intel.com

Bernard Hanzon
Department of Mathematics
University College Cork
Cork, Ireland
b.hanzon@ucc.ie

Emanuel Popovici
Department of Electrical and Electronic Engineering
University College Cork
Cork, Ireland
e.popovici@ucc.ie

***Abstract*—This paper describes a new method for pricing American options that utilizes the OneAPI framework and a Field-Programmable Gate Array (FPGA) device. By using a multinomial model based on Pascals Simplex, the method is able to price American options with improved performance over traditional methods. The use of the OneAPI framework allows for efficient parallelization of the computations on the FPGA, resulting in improved performance and faster pricing of the options. Our analysis indicates that for 7 assets, the method can process approximately 53.74 options per second at the maximum tree depth of 15, showcasing its potential for real-world applications. The results of the method are validated against existing pricing models, demonstrating its accuracy and viability for use in real-world option pricing applications.**

***Index Terms*—Fintech accelerators, American options, OneAPI, Field-Programmable Gate Arrays (FPGAs), Multi asset pricing, Digital Systems**

I. Introduction

Options are derivatives giving the holder the right, but not the obligation, to buy or sell an asset at a predetermined price on or before a specified date. Option pricing calculates a financial option's theoretical value. FPGAs, programmable integrated circuits, can speed up option pricing through parallel processing capabilities.

Well-known option pricing models include Black-Scholes [1], Binomial model [2], and Monte Carlo simulation [3]. Option pricing on FPGA involves implementing a pricing model on an FPGA device for real-time calculations [4], which is particularly beneficial for market makers and traders [5]. This paper focuses on valuing options based on multiple underlying assets using a recombining multinomial tree approach.

The paper is structured as follows: Section 2 reviews related research; Section 3 gives an overview of financial options and the Binomial model, as well as the theory for pricing derivatives on multiple assets using recombining multinomial trees. Section 4 outlines the hardware accelerator and OneAPI kernels for reconfigurable option pricing. Section 5 discusses the test environment and results, and Section 6 concludes.

II. Related Work

This section reviews existing research on accelerating option pricing for American options on multiple assets, emphasizing the authors' unique approach. The focus is on alternative methods for calculating option pricers on multiple options and binomial-tree pricing methods.

Previous research explores valuation formulas for diverse American options on multiple assets with both convex and nonconvex payoffs [6], Monte Carlo simulation techniques [7] [8] with associated convergence issues [9], and quasi-Monte Carlo methods [10]. These studies also touch on binomial-tree-based American pricing, with applications for single assets [11] [12] [13] [14].

Our novel approach uses FPGAs and Pascal's Simplex for American Options, offering superior performance and accuracy. It has been applied to two-asset European options [15] and multinomial pricing for European options using FPGAs [16]. Notably, this method also presents significant advancements in terms of latency and energy efficiency over existing binomial-tree model implementations on FPGAs [17].

A. Differences between models

The Black-Scholes model and the binomial option model are used to price options but differ in assumptions and calculation methods. The Black-Scholes model assumes continuous time and geometric Brownian motion, while the binomial option model is discrete-time and uses a tree-based approach. The former prices European options, and the latter prices both European and American options.

B. American Binomial Option Price Model

The American and European binomial option pricing models differ in the exercise of the option. The European model assumes exercise at the expiration date, while the American model allows exercise any time before expiration. The American model is more complex due to early exercise decisions.

The American Binomial Option Price Model calculates expected option values at each time point until expiration. It is widely used for its simplicity and applicability to various option types and underlying assets. However, it has limitations and may not always provide accurate prices.

C. Recombining Multinomial Trees Based on Pascal's Simplex

The method in [18] extends the binomial tree method for pricing options on multiple assets, reducing computational intensity. Using monomial ordering as described in [19], the method calculates option prices by determining the asset price vector at the final time on each of the $\binom{T+2}{2}$ nodes, denoted as ν, and computing option values iteratively. This approach, leveraging Pascal's Simplex, allows for efficient parallelization on FPGA, improved performance, accurate pricing, efficient hardware utilization, and support for early exercise in American options.

For a European option, the value at node ν at time $t = |\nu| < T$ is determined by successor nodes' values. For American options, the acceleration device generates asset price vectors, calculates option prices at the final time, and iteratively computes option values for decreasing values of $|\nu|$ with $|\nu| \leq T$.

III. Proposed FPGA architecture for financial options

A. Intel OneAPI and DPC++

Intel OneAPI is a software development platform that provides a range of tools, libraries, and APIs for various domains [20]. It allows developers to write code optimized for Intel hardware and deploy it across different platforms.

DPC++ is a programming model and language extension for parallel programming on CPUs, GPUs, and other accelerators. Based on SYCL, DPC++ enables developers to express data parallelism using C++ constructs and provides libraries for common parallel algorithms.

B. Design of the Multinomial Option Pricer

In our proposed FPGA architecture for financial options, we leveraged the inherent parallelism of Field-Programmable Gate Arrays (FPGAs) to significantly accelerate the pricing of American options. By implementing a multinomial model based on Pascal's Simplex on the FPGA, we were able to generate the option price simplex in parallel, overcoming the computational limitations of traditional methods and achieving superior performance in option pricing calculations.

The option pricer, illustrated in Figure 1 with inputs described in Table I, uses a parallelized multinomial options pricing model for American options. The design consists of several kernels that perform specific operations and work together to achieve high performance and all computations are carried out with a 64-bit precision. Furthermore, a number of the kernels (stock value, payout, sorter, option value) can be grouped together into a single compute block which then in turn can be duplicated to create a pipeline. This pipeline improves performance.

The first set of kernels computes the stock price tree. The producer kernel takes the initial stock price, volatility, and interest rate as inputs and generates a set of payoff values for each of the tree leaves, i.e., terminal nodes. The loop mux kernel computes the maximum value of the option, taking into account the option exercise possibility at each node, which is the American part of the option pricing algorithm.

The next set of kernels (in the compute blocks) computes the option values. Combined, these produce the option value for a set of options at a particular node in the tree. The option value kernel calculates the option value for each node in the price tree as per Equation 1.

$$OptionValue = \sum_{i=0}^{N-1} \left(payout[i] * \frac{1}{e^{r\frac{1}{N}}} * q_i \right) \qquad (1)$$

The final set of kernels deals with output operations. Multi-tier buffer kernels handle buffering of the option values. The channel DDR kernels move data to the DDR memory for storage. The sorter kernels sort the data in DDR memory for improved performance of subsequent kernels. Additional details on these kernels are presented in Table II.

Kernel	Description
d	Tree depth
N	Number of assets
Operation	Whether to call/put on max/min
end_node_values	defined as a N-element vector containing $(d_{N,0})^N Z_N, \cdots, (d_{0,0})^N Z_0$ where the assets are multiplied by the first elements in the direction vector to the power of N and Z_X represents S_0 for each of the assets.
init_step	defined as a NxN element matrix containing $\frac{d_{N-1,N}}{d_{N-1,N-1}}, \ldots, \frac{d_{0,1}}{d_{0,0}}$
mult_coeff	N-element multiplicative coefficient vector containing $(1/R) * q_N$,$(1/R) * q_{N-1}$,$\cdots$,(1/R)q_0 where R is defined as $e^{r(1/N)}$ (r is the risk neutral rate)
share_coeff	N-element share coefficient vector containing $\frac{1}{d_{N,1}}, \ldots, \frac{1}{d_{0,1}}$
strike_vector	N-element vector containing the strike prices K.

TABLE I. Inputs to the FPGA. These inputs are precomputed from the traditional option price parameters, i.e., ρ, σ etc. Section II in [18] explains the direction vectors $d_{i,j}$, etc, in more detail.

IV. Results

We used the Intel DevCloud, a cloud-based platform with access to various Intel hardware configurations, including FPGAs, for development and testing purposes. This allowed us to run experiments on a powerful FPGA device without any hardware investment.

We used the following versions: Target Family, Device, Board Stratix 10, 1SX280HN2F43E2VG, pac_s10_usm. The SYCL version used was 2023.0.0, and the Quartus version was 19.2.0 Build 57 Pro.

A. Accuracy Analysis

We evaluate our accuracy against the results presented in [21]. In that paper the author presents a number of simulation prices for multiple assets, up to 15, for multiple different approaches, e.g, Finite Difference (FD) [22]. In Figure 2, we present our results compared to to results from [21] for Min-Puts with parameter values: $S_i(0) = 100$, $K = 100$, $T = 0.5$, $r = 0.06$, and $\sigma_i = 0.6$ which are the same as the parameters in the referenced papers. Our results are broadly inline with the

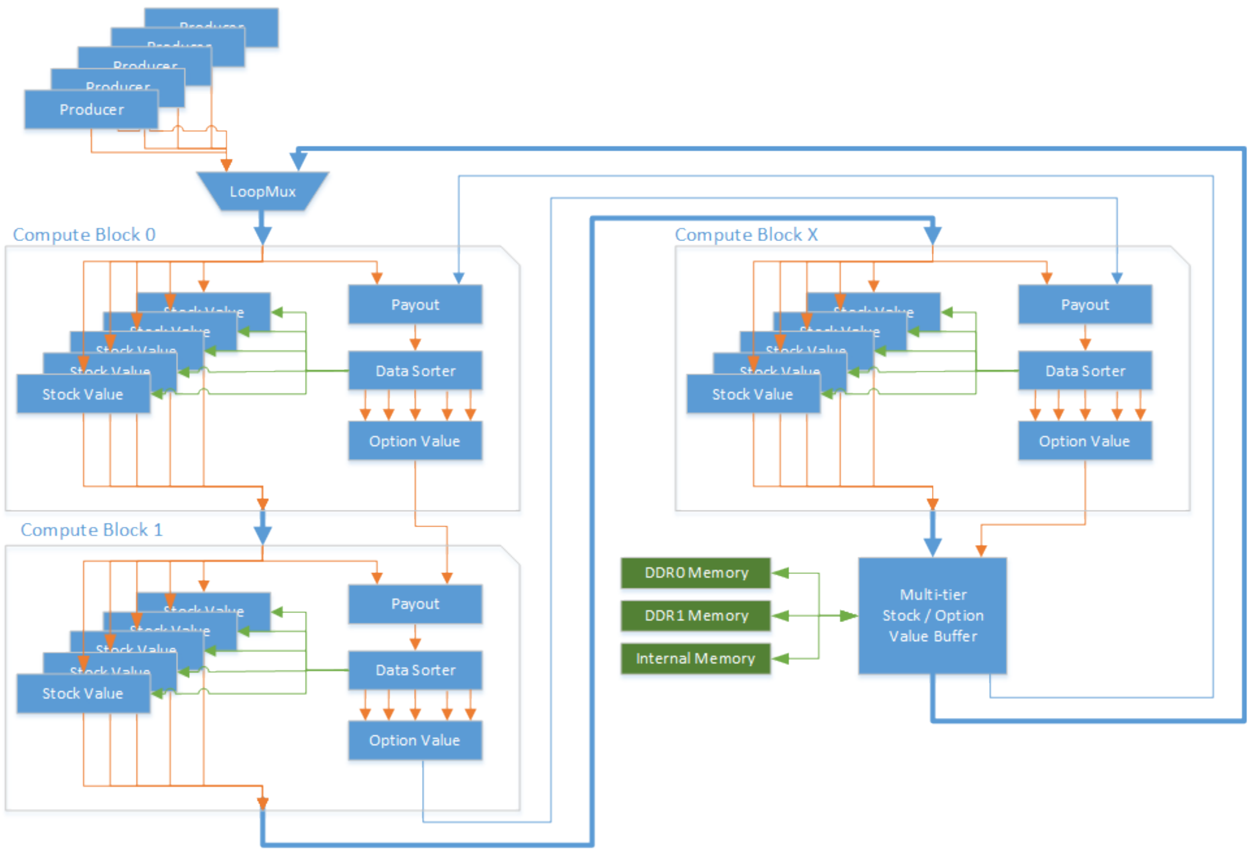

Fig. 1. High-Level Architecture of the American Option Pricer

FD and MC results however we observed an average variance of between ± 0.05 and ± 0.3.

We also took note of the early exercise depth which is dependent on the input values, for some input values there was an early exercise depth, for others a late exercise depth. We have also tested our implementation for up to seven assets, which yields graphs that are similarly aligned but not exact matches to the finite difference and Monte Carlo results.

B. Latency Analysis

Our measurements for the execution of our implementation for two assets we note the relationship appears to be polynomial, as the latency more than doubles as the tree depth increases from 500 to 1000, and continues to increase at a similar rate as the tree depth grows beyond 1000. At a tree depth of 16000, the latency is over 220 million microseconds, or more than 220 seconds.

We also plotted the latency for each of the supported asset cases, i.e., 2 to 7, for the maximum depth supported for 7 assets which is a tree of depth 15. This is presented in Figure 3. We observe a similar trend in the latency for each of the 3-7 case as the 2 asset case.

C. Resource Utilization Analysis

Table III provides information about the computational requirements of the option pricing implementation for different numbers of assets, taking into account the limitations of

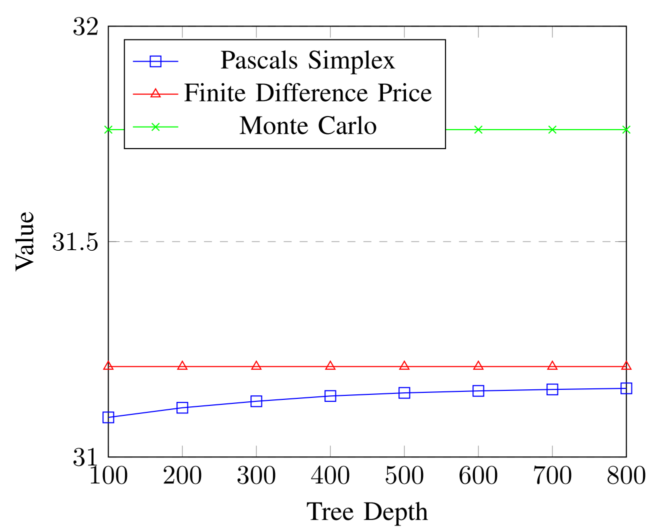

Fig. 2. Comparison of Pascals Simplex, Finite Difference Price, and Monte Carlo methods for valuing an option on three financial assets

DDR memory. The table shows that as the number of assets increases from two to seven, the number of Application Logic Units (ALUTs), Flip-Flops (FFs), Random Access Memories (RAMs), Digital Signal Processors (DSPs), and Multipliers (MLABs) also increases, indicating that the computational resources required for the implementation increase as the

Kernel	Description	Kernel	Description
producer	Generates a set of payoff values for each of the tree leaves (i.e., terminal nodes) based on the input option parameters	loopMux	Computes the payoff value for each node of the tree by recursively traversing the tree and applying the option pricing formula. Returns option price to host
stockValue	Computes the stock price for each node of the tree based on the initial stock price and the tree depth	optionValue	Compute the value of an American option at the current point in the tree. Exercise early if appropriate.
dataSorter	Reorders the tree nodes to optimize memory access patterns	payout	Computes potential profit if option is exercised
writeMultiTierBuffer	Writes the reordered tree data to a multi-tier buffer	readMultiTierBuffer	Reads the reordered tree data from the multi-tier buffer
channelDDR	Reads data from a pipe and writes it to a designated buffer in DDR memory	sorterChannelBuffer	Sorts the tree data in a designated buffer in DDR memory using a FIFO buffer

TABLE II. Detailed Overview of Kernels Employed in the Option Pricing Algorithm

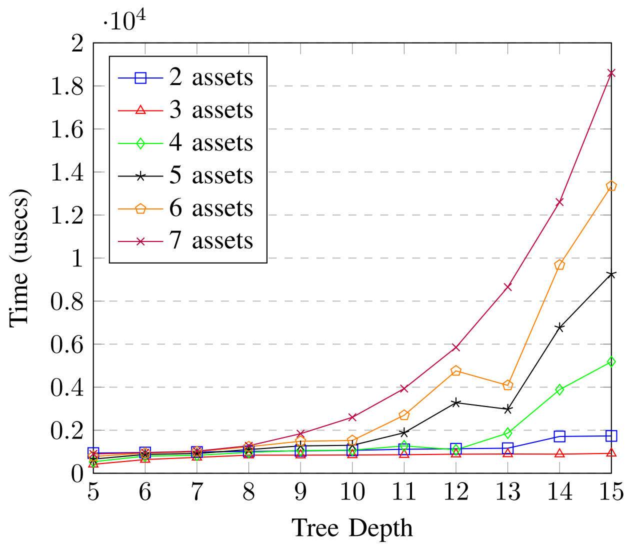

Fig. 3. Latency Comparison for American Option Pricing with Two to Seven Underlying Assets

complexity of the pricing model increases. However, the implementation only supports up to seven assets due to the constraints of DDR memory hence we never require 100% of the FPGA resources.

Assets	ALUTs	FFs	RAMs	DSPs	MLABs
2	703364	1361642	4407	1822	803
3	746235	1419122	4781	1831	908
4	789006	1476602	5122	1847	1165
5	844080	1552208	5580	1863	1436
6	906236	1638163	6010	1879	1800
7	906236	1638163	6010	1879	1800

TABLE III. Resource Utilization Analysis for Option Pricing Across Different Numbers of Underlying Assets

D. Comparative Analysis

In comparison to other FPGA-based implementations for American option pricing, our proposed architecture uniquely supports multiple assets and leverages the inherent parallelism of FPGAs with a multinomial model based on Pascal's Simplex. This approach significantly outperforms traditional methods. For instance, the LSMC method on a Xilinx Virtex-4 XC4VSX55 [13] and the binomial option pricing model using OpenCL [14] both report a 20x speed-up and the ability to evaluate over 2000 options/s with less than 20W power usage. Adapting this implementation for two assets necessitates calculating all possible outcome combinations, which could potentially lead to a computation speed of approximately 3 options per second for tree depth $N = 1024$ compared to the simplex approach which achieves 40 options per second for the same N.

V. Conclusion

The use of FPGAs for American multinomial option pricing can significantly expedite the calculation process. By leveraging parallel processing capabilities, FPGAs reduce the time required for determining theoretical option values, a crucial tool for risk management and informed investment decisions.

Intel OneAPI is a robust platform suitable for developing FPGA applications, including option pricing models. OneAPI allows developers to use a common set of tools, libraries, and APIs, optimized for Intel hardware and deployable across various platforms.

The use of FPGAs and OneAPI provides a fast and efficient solution for pricing American multinomial options. This combination is valuable to market makers and traders who need to quickly assess the value of these financial instruments. With ongoing research and development, there is potential to unlock even greater performance and precision, solidifying the role of FPGA in option pricing within the financial industry.

VI. Acknowledgements

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.

REFERENCES

[1] F. Black and M. Scholes, "The pricing of options and corporate liabilities," *Journal of political economy*, vol. 81, no. 3, pp. 637–654, 1973.

[2] J. C. Cox, S. A. Ross, and M. Rubinstein, "Option pricing: A simplified approach," *Journal of financial Economics*, vol. 7, no. 3, pp. 229–263, 1979.

[3] P. P. Boyle, "Options: A monte carlo approach," *Journal of financial economics*, vol. 4, no. 3, pp. 323–338, 1977.

[4] C. Leber, B. Geib, and H. Litz, "High frequency trading acceleration using FPGAs," in *2011 21st International Conference on Field Programmable Logic and Applications*, pp. 317–322, IEEE, 2011.

[5] R. Martin, "Wall street's quest to process data at the speed of light," *Information Week*, vol. 4, no. 21, p. 07, 2007.

[6] M. Broadie and J. Detemple, "The valuation of American options on multiple assets," *Mathematical Finance*, vol. 7, no. 3, pp. 241–286, 1997.

[7] M. Broadie and P. Glasserman, "Pricing American-style securities using simulation," *Journal of economic dynamics and control*, vol. 21, no. 8-9, pp. 1323–1352, 1997.

[8] L. Andersen and M. Broadie, "Primal-dual simulation algorithm for pricing multidimensional American options," *Management Science*, vol. 50, no. 9, pp. 1222–1234, 2004.

[9] S. Tezuka, "Financial applications of monte carlo and quasi-monte carlo methods," *LECTURE NOTES IN STATISTICS-NEW YORK-SPRINGER VERLAG-*, pp. 303–332, 1998.

[10] J. W. Wan, K. Lai, A. W. Kolkiewicz, and K. Tan, "A parallel quasi-monte carlo approach to pricing American options on multiple assets," *International Journal of High Performance Computing and Networking*, vol. 4, no. 5/6, pp. 321–330, 2006.

[11] M. Broadie and J. Detemple, "American option valuation: new bounds, approximations, and a comparison of existing methods," *The Review of Financial Studies*, vol. 9, no. 4, pp. 1211–1250, 1996.

[12] M. Gaudenzi and F. Pressacco, "An efficient binomial method for pricing American options," *Decisions in Economics and finance*, vol. 26, pp. 1–17, 2003.

[13] X. Tian and K. Benkrid, "American option pricing on reconfigurable hardware using least-squares monte carlo method," in *2009 International Conference on Field-Programmable Technology*, pp. 263–270, 2009.

[14] V. M. Morales, P.-H. Horrein, A. Baghdadi, E. Hochapfel, and S. Vaton, "Energy-efficient FPGA implementation for binomial option pricing using OpenCL," in *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1–6, 2014.

[15] A. O. Mahony, G. Zeidan, B. Hanzon, and E. Popovici, "A parallel and pipelined implementation of a pascal-simplex based two asset option pricer on FPGA using OpenCL," in *2020 IEEE Nordic Circuits and Systems Conference (NorCAS)*, pp. 1–6, IEEE, 2020.

[16] A. O. Mahony, G. Zeidan, B. Hanzon, and E. Popovici, "A parallel and pipelined implementation of a pascal-simplex based multi-asset option pricer on FPGA using OpenCL," *Microprocessors and Microsystems*, vol. 90, p. 104508, 2022.

[17] A. Tavakkoli and D. B. Thomas, "Low-latency option pricing using systolic binomial trees," in *2014 International Conference on Field-Programmable Technology (FPT)*, pp. 44–51, 2014.

[18] D. Sierag and B. Hanzon, "Pricing derivatives on multiple assets: recombining multinomial trees based on Pascal's simplex," *Annals of Operations Research*, pp. 1–27, 2017.

[19] D. Cox, J. Little, and D. O'Shea, *Ideals, varieties, algorithms: An introduction to computational algebraic geometry and commutative algebra, UTM*, pp. 52–58. Springer-Verlag, New York, 1992.

[20] K. Obata, H. M. Waidyasooriya, and M. Hariyama, "Implementation of an FPGA-oriented complex number computation library using Intel OneAPI DPC++," in *2022 IEEE 65th International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 1–4, IEEE, 2022.

[21] L. C. Rogers, "Monte carlo valuation of American options," *Mathematical Finance*, vol. 12, no. 3, pp. 271–286, 2002.

[22] P. Hartley, "Pricing a multi-asset American option," tech. rep., Working paper, University of Bath, 2000.