UDC 004.2:004.315 https://doi.org/10.20535/2786-8729.6.2025/333919

METHOD FOR ON-LINE ACCELERATION OF DEPENDENT OPERATION CHAINS USING REDUNDANT CODE ON FPGA WITH SYSTEM OF LINEAR EQUATIONS EXAMPLE

Illia Verbovskyi *
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/0009-0008-4782-4281

Valerii Zhabin
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute", Kyiv, Ukraine
https://orcid.org/0000-0003-0377-3394

* Corresponding author: illyaverb@gmail.com

This study examines methods for accelerating the execution of dependent operation chains in on-line mode through parallel processing of operands at the bit level in redundant code on field-programmable gate arrays (FPGA). The object of research is the hardware implementation of the Thomas algorithm for solving systems of linear equations with tridiagonal matrices on FPGA platforms. The aim is to develop a method for accelerating dependent operation chains in on-line mode using redundant code with minimization of pin count requirements. The methodology employs algorithmic analysis, hardware modeling using Active HDL, performance evaluation based on timing characteristics and resource utilization on Altera Cyclone III EP3C5E144 platform, with verification performed using Quartus.

The results reveal bottlenecks in traditional FPGA implementations of the Thomas algorithm and demonstrate that the proposed optimized method provides over threefold performance improvement while maintaining constant pin count regardless of operand bit depth. The developed computing module architecture enables bit-wise parallel data processing and supports a modified version of the Thomas algorithm adapted for on-line operation. The scientific novelty lies in combining redundant code with on-line computation techniques to simultaneously achieve computational acceleration and hardware implementation simplification. The practical value is determined by the applicability of the proposed approach to resource-constrained FPGA platforms, ensuring efficient implementation of computationally intensive algorithms with dependent operation chains.

Keywords: FPGA, on-line computing, digit-by-digit processing, tridiagonal systems, redundant number systems.

1. Introduction

Modern approaches to accelerating computing in computer systems increasingly rely on hardware solutions, particularly FPGAs, which offer high performance with low power consumption. However, traditional methods of implementing FPGA computational algorithms face significant limitations related to the need for a large number of contacts for parallel data input and output, as well as the inability to combine data-dependent sequences effectively.

The acceleration of computational algorithms on reconfigurable hardware platforms is an important area in computer architecture and computational mathematics. This research addresses limitations in current parallel computing methods that affect many scientific and engineering applications.

Parallel arithmetic methods implemented in operating modules (OM) enable the acceleration of calculations in parallel systems from the program level down to the operational level. Nonetheless, it is not possible to speed up the execution of chains of dependent operations in partial overlapping

mode. Until the OM in the chain receives an operand for its operation, it cannot begin executing it. This means the execution of dependent operations in the chain halts. The final result of the calculations is obtained only after completing all operations in the sequence of data-dependent operations.

Therefore, implementing parallel branches of algorithms in different OMs does not accelerate the receipt of the final result. Analyzing the efficiency of performing computational algorithms on FPGAs using on-line calculation techniques and a redundant number system will be examined through an example of solving systems of linear algebraic equations (SLE) with the Thomas algorithm.

This problem is essential because data-dependent computational sequences are common in mathematical modeling, numerical analysis, and scientific computing. The difficulty in parallelizing such operations creates a bottleneck that limits the performance of computational systems across multiple scientific domains.

At various stages of this method, it becomes necessary to perform chains of data-dependent operations that can be executed in a partially overlapped manner over time when using a redundant number system. This method is widely used across different fields of science and technology, from modeling complex physical processes to computer graphics problems. The Thomas algorithm is one of the most efficient algorithms for solving such systems, with a linear time complexity of $O(n)$, compared to the estimated $O(n^3)$ complexity of the standard Gaussian method [1].

Research into optimizing data-dependent computational algorithms on FPGA platforms using advanced numerical techniques is a relevant scientific direction. Addressing these limitations in parallel computation methods is necessary for improving computational capabilities. This improvement is needed in various fields of science and technology. Therefore, studies in this area are both timely and essential.

2. Literature review and problem statement

The problems of hardware implementation of numerical methods are actively explored in modern scientific literature. In particular, the issues related to optimizing FPGA calculations are discussed in the works [2–3], which demonstrate the effectiveness of the on-line mode for computing functions. Research [4] shows the advantages of this approach when working with various functions.

Modern research on FPGA-based numerical methods actively investigates optimization through redundant numeral systems (RNS) and online arithmetic. Recent studies demonstrate that integrating RNS variants, such as R-RNS, into high-performance architectures can decrease latency and energy consumption [5], while enhanced FPGA-based redundant adders further improve timing efficiency [6]. RRNS designs also boost fault tolerance without excessive hardware cost [7].

The implementation of various methods for solving SLE on an FPGA is examined in papers [8–10], where the authors mainly focus on classical algorithms, such as the Gaussian method and LU decomposition. However, insufficient attention is given to specific methods for sparse matrices, especially tridiagonal ones. Particular focus should be on studies [11–14], which are dedicated to using redundant numeral systems to optimize calculations.

Studies on FPGA implementations of the Thomas algorithm for tridiagonal systems demonstrate gains from batching, vectorization, and heterogeneous integration with OpenCL [15–16]. Advanced Thomas-Thomas and Thomas-PCR schemes with high-bandwidth memory (HBM) outperform GPU counterparts in speed and power efficiency [17].

Nonetheless, these papers primarily analyze individual arithmetic operations rather than their combinations as part of high-level algorithms like the Thomas algorithm. Therefore, there is a need for a comprehensive study of the potential. This study should combine an on-line computational mode with a redundant number system. The goal is to improve the implementation of the FPGA Thomas algorithm.

3. The aim and objectives of the study

The aim of the study is to develop a method for speeding up the execution of dependent operation chains in on-line mode through parallel processing of operands at the bit level in redundant code on an FPGA, while reducing the number of required pins. This will be demonstrated by implementing the Thomas algorithm for solving systems of linear equations. The objectives of the study are:

– To analyze the computational structure of dependent operation chains in the Thomas algorithm and assess the limitations of conventional FPGA implementations regarding pin count and processing speed.

– To create an optimized method that allows parallel processing of operands at the bit level using redundant code representation in on-line mode, while minimizing the FPGA pins needed for data input and output operations.

To achieve the stated objectives, it is important to examine the computational structure of the Thomas algorithm with an emphasis on dependent operation chains, analyze the properties of redundant number systems that support bit-level parallel processing, and explore on-line computation techniques that help reduce the number of FPGA pins required.

4. The study materials and methods for accelerating dependent operation chains in on-line mode

4.1. Research methodology and approach

The study employs a combined theoretical and experimental approach to develop and validate the proposed method. The research methodology includes: algorithmic analysis method for identifying computational bottlenecks and dependencies in the Thomas algorithm structure; hardware modeling approach using FPGA development tools to simulate and test the proposed architecture; performance evaluation methodology based on timing analysis and resource utilization metrics; comparative analysis method to assess the efficiency of the proposed approach against conventional implementations.

The research is conducted using Active HDL with VHDL code for FPGA implementation and Quartus for functional verification. The target hardware platform is an Altera Cyclone III EP3C5E144 FPGA selected for its balanced performance and resource characteristics suitable for the proposed algorithm implementation.

4.2. Thomas algorithm for tridiagonal matrices

Considering a tridiagonal system of linear algebraic equations of the form:

$$\begin{bmatrix} b_1 & c_1 & & & 0 \\ a_2 & b_2 & c_2 & & \\ & a_3 & b_3 & \ddots & \\ & & \ddots & \ddots & c_{n-1} \\ 0 & & & a_n & b_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_n \end{bmatrix}, \quad (1)$$

where a, b, c – matrix coefficients, d – right vector, x – solution vector.

The Thomas algorithm consists of two stages:

– *Straight stroke* – calculation of factors

$$c'_1 = \frac{c_1}{b_1};\ c'_i = \frac{c_i}{b_i - c'_{i-1}a_i},\ i = \overline{2, n-1}, \quad (2)$$

$$d'_1 = \frac{d_1}{b_1};\ d'_i = \frac{d_i - d'_{i-1}a_i}{b_i - c'_{i-1}a_i},\ i = \overline{2, n}. \quad (3)$$

– *Reverse* – calculation of the desired values x

$$x_n = d'_n, \tag{4}$$

$$x_i = d'_i - c'_i x_{i+1};\ i = n-1, n-2, \dots, 1. \tag{5}$$

The algorithm works well for tridiagonal systems but has chains of data-dependent operations, which lowers its efficiency on computing modules that support at least the level of parallelism in operations. To combine dependent operations, parallel calculation at the bit level is necessary.

The analysis of dependent operation chains in the Thomas algorithm reveals the need for alternative arithmetic representations that can support bit-level parallelism while reducing hardware resource requirements.

4.3. Redundant number system

Quasi-canonical redundant numeral systems, which differ from canonical numeral systems by only one additional digit, can be considered quite convenient. For example, at the base k = 2, the canonical system has numbers $x \in \{0,1\}$, and quasi-canonical – $x \in \{-1,0,1\}$. For example, to compute a fractional-rational function for rational values m і r:

$$Z = \frac{\sum_{s=1}^{\frac{m}{2}} \prod_{k=2s-1}^{2s} X_k}{\sum_{j=1}^{r} Y_j}, \tag{6}$$

the values of the arguments and the result of the function will be:

$$X_k = \sum_{i=1}^{n} x_{ki} 2^{q-i},\ (0 \le X_k < 2^q), \tag{7}$$

$$Y_j = \sum_{i=1}^{n} y_{ji} 2^{q-i},\ \left(2^{q-1} \le Y_j < 2^q\right), \tag{8}$$

$$Z = \sum_{i=1}^{n} z_i 2^{q-i},\ \left(0 \le Z < \frac{m2^q}{r}\right),\ z_i \in \{-1, 0, 1\}, \tag{9}$$

where $x_{ki}, y_{ji} \in \{0, 1\}$ – operand digits, q – place of the comma [18].

In redundant numeral systems, when performing addition and subtraction, transfers or borrowings in higher digits do not occur. The output digit at the OM cannot change during transmission between the OMs.

The implementation of redundant number systems provides the foundation for on-line computation techniques that enable early initiation of dependent operations with minimized pin count.

4.4. Principles of on-line computing

Additionally, in on-line mode, all basic algebraic operations are executed in OM from higher digits. This enables you to begin performing operations immediately after receiving the highest bits of operands, without waiting for the lower bits. This fact allows dependent operations to be combined, which speeds up the calculation of the final result.

Let the partial operands on the i-th bar be denoted as: be denoted as:

$$X_{ki} = x_{k1} x_{k2} \dots x_{ki} 00 \dots 00, \tag{10}$$

$$Y_{ji} = y_{j1} y_{j2} \dots y_{ji} 00 \dots 00, \tag{11}$$

and the partial result as

$$Z_i = z_1 z_2 \ldots z_i 00 \ldots 00, \tag{12}$$

then, at each i-th cycle of the calculation, a transformation is performed:

$$N_i = 2R_{i-1} + F_i, \tag{13}$$

$$R_i = N_i - z_i, \tag{14}$$

where N_i, R_i – internal variables, F_i – function from partial operands, and z_i determined by the conditions:

$$z_i = \begin{cases} -1, & N_i < -2^{-1} \\ 0, & -2^{-1} \le N_i < 2^{-1} \\ 1, & 2^{-1} \le N_i \end{cases}. \tag{15}$$

For basic arithmetic operations, the function F_i is as follows:

$$F_i = 2^{-p}(x_i \pm y_i), \tag{16}$$

$$F_i = 2^{-p}(x_i Y_i + y_i X_{i-1}), \tag{17}$$

where p is the delay parameter, which determines the number of cycles required to start forming the result from the higher digits. Performing dependent operations in partial overlapping mode allows the use of operations within the redundant number system. These systems differ from canonical systems because the number of distinct digits is greater than the base of the numeral system.

4.5. General structure of the computing module

To implement the Thomas algorithm in on-line mode, a system of interconnected operating modules has been developed.

Each module includes: input buffers for the accumulation of partial operands; a function calculation unit F_i; a block for the formation of the result digit z_i; registers for storing intermediate variables; an output buffer to pass the result to the next module. The interaction between modules is organized according to the principle of a pipeline. In this system, the results of some operations are transmitted to the input of others. This transmission happens in parallel with the calculation of subsequent bits.

4.6. Optimized algorithm for the on-line mode

To adapt the Thomas algorithm to the on-line mode of calculations, a modified algorithm has been developed. This algorithm optimizes the order of operations for the maximum time combination. The key feature of the proposed approach is the possibility of starting the calculation immediately. Specifically, coefficients can be calculated after obtaining the first significant digits of coefficients. This happens without waiting for them to complete their full calculation. Similarly, the reverse stroke of the algorithm can be started even before the complete completion of the forward stroke.

The operating tree presented in Fig. 1 illustrates the hierarchical organization of computational operations and demonstrates how the modified algorithm enables parallel execution of dependent operations. The structure shows that operations at different levels can be initiated as soon as the required input data becomes available, eliminating the need to wait for complete results from previous stages.

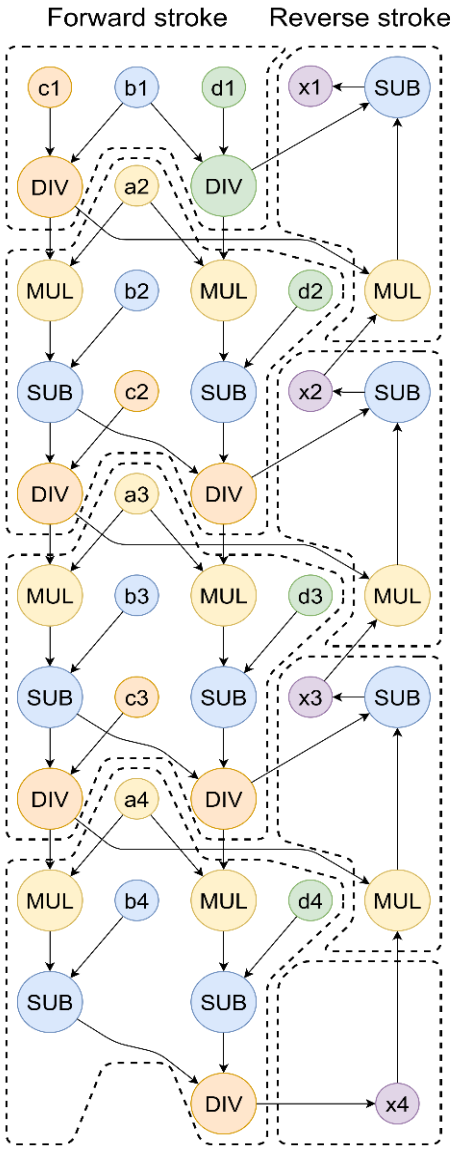

Fig. 1. Operating tree of the modified Thomas algorithm showing distribution of operations by levels and dependencies between forward and reverse strokes

This approach significantly reduces the overall computation time while maintaining the algorithmic correctness of the Thomas method.

4.7. Implementation features for floating-point numbers

To handle floating-point numbers in on-line mode, a specialized unit has been developed that processes the order and mantissa separately. The order is fully transmitted before the start of the mantissa transmission, allowing the circuit to be preconfigured for correct operand alignment [19]. The processing algorithm is based on the technique presented in [19], with modifications to improve efficiency specifically for operations related to the Thomas algorithm. An example of how the system functions with similar methods can be found in [20].

5. Results of investigating the acceleration of dependent operation chains execution in on-line mode

5.1. Analysis of computational structure and bottlenecks identification

To assess the computational characteristics of dependent operation chains in the Thomas algorithm, a systematic analysis was conducted using VHDL models of the developed computing

modules. The analysis identified key bottlenecks in conventional FPGA implementations, particularly the sequential nature of coefficient calculations and the linear increase in pin count requirements with operand bit depth.

The main evaluation criteria established for bottleneck identification were: number of cycles required to calculate the result; number of FPGA logic elements involved; maximum operating frequency; system throughput (number of operations per unit of time); pin count requirements for data input/output.

5.2. Development and validation of the optimized implementation method

The developed method integrating on-line arithmetic techniques with redundant number systems was implemented and validated through synthesis and simulation in the FPGA design environment using test vectors of different dimensions based on Altera Cyclone III EP3C5E144.

Scientific novelty: The proposed method uniquely combines bit-level parallel processing in redundant code with on-line computation to achieve simultaneous acceleration of dependent operations and minimization of pin count requirements.

5.2.1. Performance comparison results

The proposed on-line approach is compared with the standard sequential implementation of the Thomas algorithm based on conventional binary arithmetic as described in [1, 10]. The classical approach follows the traditional two-phase execution: complete forward elimination followed by complete backward substitution, implemented using standard arithmetic units with parallel operand processing [8, 15]. The simulation results for various system parameters are shown in Fig. 2.

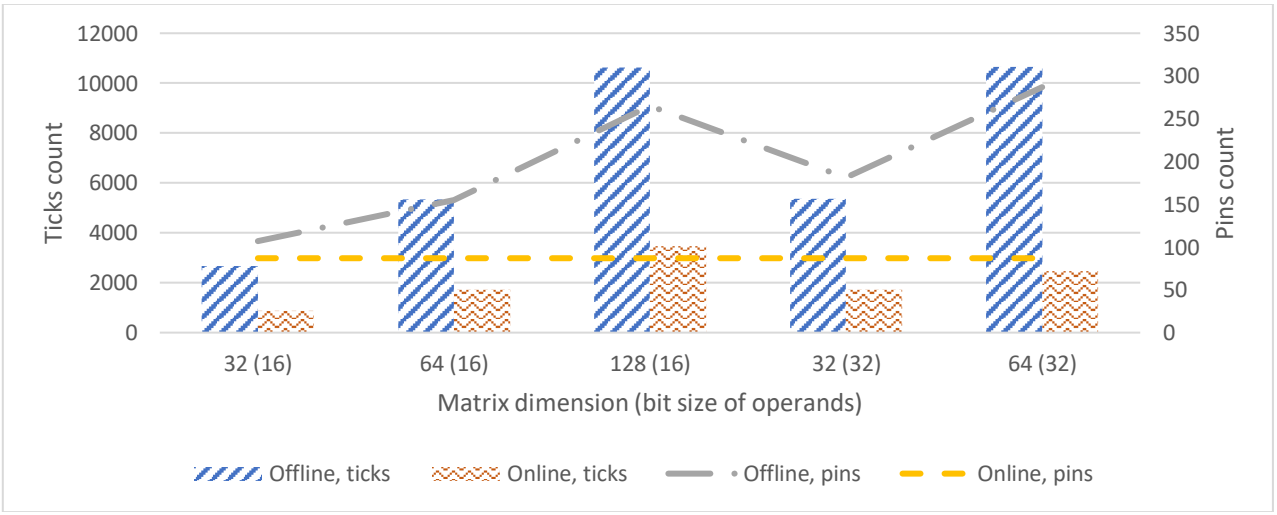

Fig. 2. Compare the time and number of pins for offline and on-line modes.

Figure 2 demonstrates the performance comparison between the classical sequential approach (offline mode) and the proposed method (on-line mode). The graph shows results for different matrix dimensions and operand bit sizes. The blue diagonal-striped bars represent execution time in clock cycles for the offline approach, while the orange zigzag-patterned bars show the lower execution times for the on-line method. The gray dot-dashed line shows pin count requirements for the offline approach, which varies with operand counts, while the yellow dashed line demonstrates constant pin count requirements for the on-line method across all configurations.

As can be seen from the results, the use of on-line mode allows acceleration more than three times compared to the classical sequential approach [1, 9]. Moreover, this advantage is maintained with an increase in both the dimensionality of the system and the bit depth of the operands, demonstrating the scalability of the proposed method.

5.2.2. Hardware resource optimization

Hardware cost estimation was carried out by synthesizing the developed modules for the target FPGA and comparing with conventional implementations using standard arithmetic units as reported in [8, 10, 15]. The traditional approach employs parallel arithmetic units with full-width operand processing, requiring extensive interconnection networks and large register files [16]. The results of the comparison are presented in Fig. 3, which shows the relative number of resources used for different implementations.

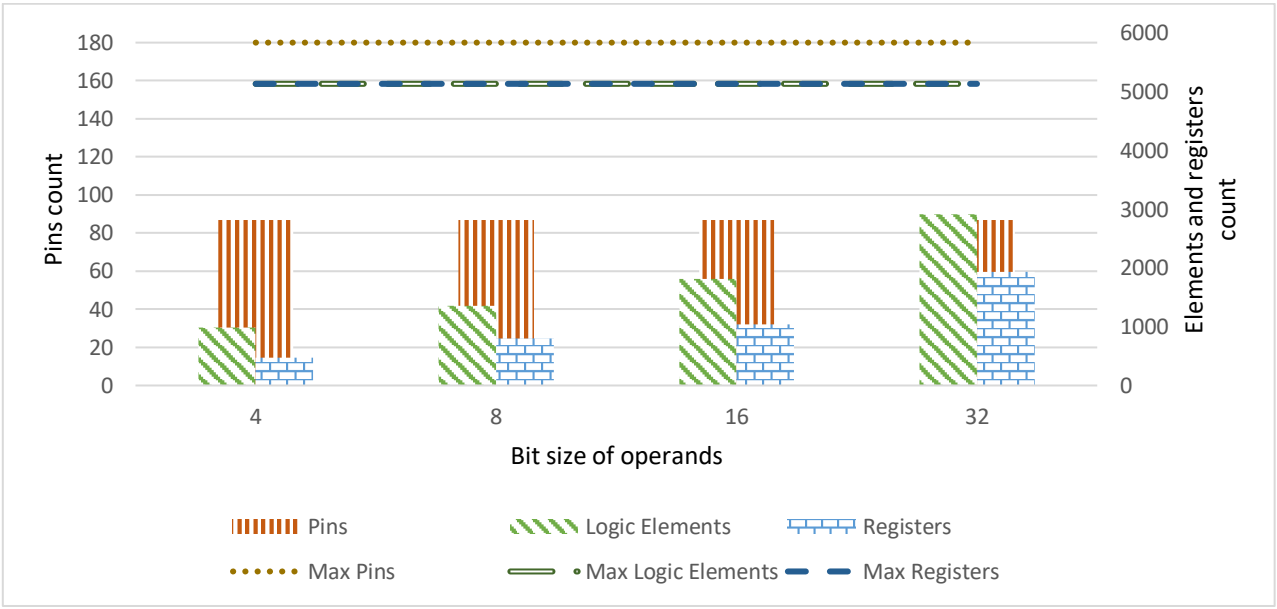

Fig. 3. Comparison of the element base for the implementation of the proposed approach

Figure 3 presents hardware resource utilization across different operand bit sizes (4, 8, 16, and 32 bits). The graph shows three types of resources: brown solid with dots bars represent pin count requirements, green diagonal-striped bars show logic elements usage, and blue bricked bars indicate register count. The horizontal lines represent maximum available resources: dotted yellow line for maximum pins (180), dot-dashed green line for maximum logic elements (5136), and dashed blue line for maximum registers (5136). The results demonstrate that pin count requirements remain relatively constant across different bit sizes for the proposed approach, while logic elements and registers scale moderately with operand width.

An important advantage of the proposed approach is the significant reduction and stabilization in the number of I/O pins required compared to traditional parallel implementations [15, 16]. As shown in Fig. 3, the pin count remains approximately constant at around 87 pins regardless of operand bit size, whereas classical approaches following conventional FPGA design practices [8, 17] would require linearly increasing pin counts. The logic elements usage grows moderately from 983 elements for 4-bit operands to 2914 elements for 32-bit operands, remaining well below the available resources limit.

5.2.3. Latency analysis and timing characteristics

Analysis of time characteristics showed that for operands with n digits and delay parameter p, the total number of cycles required to calculate the result by the Thomas algorithm is: for the classical approach $28(3n + p) + 25$; for the on-line mode: $23n + 12p + 6$.

A time diagram illustrating the distribution of operations in time for the two approaches is presented in Fig. 4. The diagram shows operations measured in clock cycles, with timing dependencies based on the bit width of values obtained after computations, demonstrating how the proposed on-line method enables partial overlapping of dependent operations compared to the sequential execution pattern of the conventional approach.

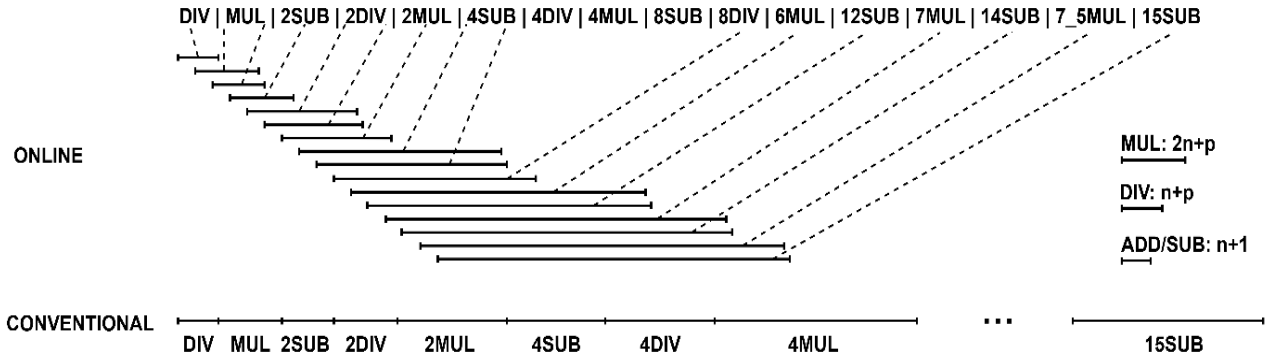

Fig. 4. Time diagram of operations for conventional and on-line approaches to solving a problem.

6. Discussion of results of the dependent operation chain acceleration investigation

6.1. Interpretation of performance improvement results

The obtained results demonstrate significant performance improvements that the fundamental characteristics of the proposed approach can explain. The more than threefold acceleration achieved through on-line mode implementation is attributed to the elimination of sequential dependencies between operations. In conventional implementations, each operation in the Thomas algorithm must wait for the complete calculation of its predecessor, creating computational bottlenecks. The proposed method overcomes this limitation by enabling partial overlapping of dependent operations through bit-level parallel processing in redundant code.

The consistency of performance gains across different system dimensions and operand bit depths indicates that the approach scales effectively. This scalability stems from the inherent properties of redundant number systems, where carry propagation is eliminated, allowing each digit position to be processed independently. The on-line computation mode further enhances this by enabling operations to commence as soon as the most significant digits become available.

6.2. Analysis of hardware resource optimization

The constant pin count requirement regardless of operand bit depth offers a notable architectural benefit. This is due to the sequential nature of digit transmission in online mode, where operands are sent bit-by-bit instead of in parallel. Traditional designs need separate pins for each bit of all operands, causing a linear increase in pin count. The proposed method fundamentally alters this approach by reusing the same communication channels for each digit transmission.

The hardware resource utilization results indicate efficient use of FPGA logic elements. The elimination of complex carry-propagation circuits typical in conventional arithmetic units offsets the slight increase in logic complexity. This trade-off proves advantageous for the target application domain.

6.3. Limitations and constraints analysis

Despite the significant advantages, the proposed method has certain limitations that must be acknowledged. The approach requires a delay parameter p for proper operation, which introduces initial latency. However, this delay becomes negligible for larger problem sizes where $n >> p$. Additionally, the method is optimized explicitly for data-dependent operation chains and may not provide similar benefits for fully parallelizable algorithms.

The implementation complexity increases due to the need for specialized arithmetic units capable of redundant number system operations. This may require additional design effort and verification compared to standard arithmetic implementations.

6.4. Comparison with existing approaches

The results demonstrate clear advantages over conventional FPGA implementations of the Thomas algorithm. The achieved performance improvements surpass those reported in recent

literature for similar tridiagonal system solvers [8, 15, 16]. The unique combination of on-line arithmetic with redundant number systems addresses gaps identified in the literature review, where previous work focused on either hardware optimization or arithmetic system improvements separately. A detailed quantitative comparison is presented in Table 1, where speed-up times are calculated relative to the traditional offline method to provide consistent baseline for performance evaluation across different approaches.

Table 1. Comparison of the proposed method with conventional approaches reported in recent literature (for 128*128 size matrix with 16-bit operands).

Method	Clock Cycles	Speed-up Times	Pin Count	FPGA Platform
High-level synthesis approach [8]	≈8400	1.269	≈400	Xilinx Zynq
High throughput solver [15]	≈5310	2.007	≈350	Xilinx Ultrascale
Heterogeneous scalable solver [16]	≈7600	1.402	≈380	Intel Arria 10
Proposed on-line method	**3456**	**3.083**	**87**	**Altera Cyclone III**
Traditional off-line method	10656	baseline	287	Altera Cyclone III

6.5. Prospects for further research

The successful application of the proposed method to the Thomas algorithm opens several promising research directions:

– Extension to other numerical algorithms: The approach could be adapted for other data-dependent computational sequences, such as iterative solvers and recursive algorithms.

– Advanced redundant number systems: Investigation of higher-radix redundant systems could potentially further improve performance and reduce hardware complexity.

–Hybrid architectures: Combining the proposed method with emerging FPGA architectures featuring dedicated DSP blocks and embedded processors could yield additional performance benefits.

– Scalability studies: Research into implementing the approach on larger FPGA families and multi-FPGA systems could address more complex computational problems.

– Real-time applications: The predictable timing characteristics make the method suitable for real-time computational applications, warranting further investigation in this domain.

The demonstrated effectiveness of integrating on-line computation with redundant arithmetic suggests potential for broader impact across scientific computing applications requiring acceleration of sequential computational processes.

Conclusions

Based on the conducted research, the following conclusions can be drawn corresponding to the stated objectives:

1. Analysis of computational structure and bottleneck identification: The computational characteristics of dependent operation chains in the Thomas algorithm have been systematically analyzed, revealing that sequential dependencies between coefficient calculations represent the primary bottleneck limiting parallelization efficiency on FPGA platforms. The analysis showed that conventional implementations exhibit linear growth in pin count requirements with operand bit depth, and sequential execution constraints hinder the effective acceleration of data-dependent operations.

2. Development of optimized implementation method: An optimized method integrating on-line arithmetic techniques with redundant number systems has been successfully developed and validated. The method enables parallel processing of operands at the bit level while maintaining constant pin count requirements regardless of operand bit depth. Experimental validation

demonstrates over threefold performance improvement compared to conventional sequential implementations, confirming the effectiveness of combining on-line computation mode with redundant code representation for accelerating dependent operation chains on FPGA platforms.

Scientific novelty: The developed method combines bit-level parallel processing in redundant code with on-line computation techniques to simultaneously achieve acceleration of data-dependent operations and minimization of hardware pin requirements.

Practical value: The proposed approach provides a viable solution for implementing computationally intensive algorithms with dependent operation chains on resource-constrained FPGA platforms, offering significant performance improvements while reducing hardware complexity and pin count requirements.

References

[1] V. S. Ryaben'kii and S. V. Tsynkov, "Introduction," in *A Theoretical Introduction to Numerical Analysis*, Chapman Hall/CRC, 2006, pp. 1–19. https://doi.org/10.1201/9781420011166-1.

[2] J. Piñeiro, M. D. Ercegovac, and J. D. Bruguera, "Algorithm and architecture for logarithm, exponential, and powering computation," *IEEE Trans. Comput.*, vol. 53, no. 9, pp. 1085–1096, Sep. 2004. https://doi.org/10.1109/TC.2004.53.

[3] I. Dychka, V. Zhabin, and V. Zhabina, "Analysis of on-line computation effectiveness in redundant number system," in *Proc. IEEE First Int. Conf. System Anal. & Intell. Comput. (SAIC)*, 2018, pp. 1–6. https://doi.org/10.1109/SAIC.2018.8516877.

[4] J. Bajard, S. Kla, and J. Muller, "BKM: A new hardware algorithm for complex elementary functions," *IEEE Trans. Comput.*, vol. 43, no. 8, pp. 955–963, Aug. 1994. https://doi.org/10.1109/12.295857.

[5] S. Mousavi, D. Rahmati, S. Gorgin, and A. Lee, "Enhancing efficiency in computational intensive domains via redundant residue number systems," in *Proc. 21st Int. SoC Des. Conf. (ISOCC)*, Sapporo, Japan, Aug. 19–22, 2024, pp. 330–331. https://doi.org/10.1109/isocc62682.2024.10762680.

[6] S. R. Sahu, B. K. Bhoi, and M. Pradhan, "Improved redundant binary adder realization in FPGA," *J. Circuits Syst. Comput.*, vol. 30, no. 8, pp. 2150287-1–2150287-23, Jun. 2021. https://doi.org/10.1142/s021812662150287x.

[7] Y. Zhang, "An FPGA implementation of redundant residue number system for low-cost fast speed fault-tolerant computations," M.Eng. thesis, Nanyang Technol. Univ., Singapore, 2018. https://doi.org/10.32657/10220/47113.

[8] H. Meng, K. Wakabayashi, and T. Kuroda, "A scalable linear equation solver FPGA using high-level synthesis," in *Proc. 24th Workshop Synthesis System Integration Mixed Inf. Technol. (SASIMI)*, Taipei, Taiwan, 2022, pp. 145–150. [Online]. Available: https://sasimi.jp/new/sasimi2022/files/archive/pdf/p145_B-7.pdf. Accessed: May 1, 2025.

[9] S. M. Perera and N. Bebiano, "A low-cost and numerically stable algorithm to solve tridiagonal systems via quasiseparable matrices," *Research Square*, pp. 1–22, Aug. 2023. https://doi.org/10.21203/rs.3.rs-3200350/v1.

[10] M. D. Ercegovac and M. Muller, "Arithmetic processor for solving tridiagonal systems of linear equations," in *Proc. 40th Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, Oct. 29–Nov. 1, 2006, pp. 337–340. https://doi.org/10.1109/acssc.2006.354763.

[11] V. I. Zhabin, V. I. Korneĭchuk, and V. P. Tarasenko, "Computation of rational functions for many arguments," *Autom. Remote Control*, vol. 38, no. 12, pp. 1864–1871, 1978.

[12] J. T. Butler and T. Sasao, "Redundant multiple-valued number systems," Naval Postgraduate School, Monterey, CA, USA, Tech. Rep., 1997. [Online]. Available: http://pi.314159.ru/butler1.pdf. Accessed: Aug. 1, 2025.

[13] V. I. Zhabin, I. A. Zhukov, I. A. Klymenko, and V. V. Tkachenko, *Prykladna teoriia tsyfrovykh avtomativ: Navchalnyi posibnyk*, 2nd ed. Kyiv: NAU, 2009. [Online]. Available: https://comsys.kpi.ua/wp-content/uploads/2025/02/prykladna-teoriya-tsyfrovykh-avtomativ.pdf. Accessed: Aug. 1, 2025.

[14] V. Y. Zhabyn, V. Y. Korneichuk, and V. P. Tarasenko, "Some machine methods for computing rational functions of many arguments," *Autom. Telemechanics*, vol. 38, no. 12, pp. 145–154, 1977.

[15] K. Kamalakkannan, G. R. Mudalige, I. Z. Reguly, and S. A. Fahmy, "High throughput multidimensional tridiagonal system solvers on FPGAs," in *Proc. Int. Conf. Supercomput.*, Virtual Event, 2022, pp. 1–13. https://doi.org/10.1145/3524059.3532371.

[16] H. J. Macintosh, J. E. Banks, and N. A. Kelson, "Implementing and evaluating an heterogeneous, scalable, tridiagonal linear system solver with opencl to target fpgas, gpus, and cpus," *Int. J. Reconfigurable Comput.*, vol. 2019, pp. 1–13, Oct. 2019. https://doi.org/10.1155/2019/3679839.

[17] T. Kuo and C. Wu, "FPGA implementation of a novel multifunction modulo (2n ± 1) multiplier using radix-4 booth encoding scheme," *Appl. Sci.*, vol. 13, no. 18, pp. 1–12, Sep. 2023, Art. no. 10407. Accessed: Aug. 16, 2025. [Online]. Available: https://doi.org/10.3390/app131810407.

[18] I. Verbovskyi and V. Zhabin, "Improving the efficiency of functions computation in on-line mode on FPGA," in *Proc. Int. Conf. Secur. Fault Tolerance Intell.*, Kyiv, Ukraine, 2022, pp. 1–8. Accessed: Aug. 1, 2025. [Online]. Available: https://icsfti-proc.kpi.ua/article/view/281001.

[19] V. Zhabin, V. Zhabina, and O. Verba, "Asynchronous on-line float-point computations in systems with direct connections between computation units," in *Proc. IEEE 2nd Int. Conf. System Anal. & Intell. Comput.*, 2019, pp. 1–5.

[20] V. I. Zhabin, V. I. Korneichuk, V. P. Tarasenko, and A. A. Shcherbyna, "Strukturnyi sposib shvydkoho vyrishennia system rivnian alhebry z trydiahonalnoiu matrytseiu," Tech. Rep., Feb. 1979.

УДК 004.2:004.315

МЕТОД ОНЛАЙН-ПРИСКОРЕННЯ ЗАЛЕЖНИХ ЛАНЦЮГІВ ОПЕРАЦІЙ ІЗ ВИКОРИСТАННЯМ НАДЛИШКОВОГО КОДУ НА ПЛІС НА ПРИКЛАДІ СИСТЕМИ ЛІНІЙНИХ РІВНЯНЬ

Ілля Вербовський
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0009-0008-4782-4281

Валерій Жабін
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0000-0003-0377-3394

У дослідженні розглянуто метод прискорення виконання залежних ланцюгів операцій у неавтономному режимі шляхом паралельної обробки операндів на бітовому рівні в надлишковому коді на програмованих логічних інтегральних схемах (FPGA). Об'єктом дослідження є апаратна реалізація алгоритму Томаса для розв'язання систем лінійних рівнянь з тридіагональною матрицею на платформах ПЛІС. Мета роботи полягає у розробленні методу прискорення залежних ланцюгів операцій у неавтономному режимі з використанням надлишкового коду та мінімізацією кількості елементів вводу-виводу. Методологія включає алгоритмічний аналіз, апаратне моделювання з використанням Active HDL, оцінку продуктивності на основі часових характеристик та використання ресурсів на платформі Altera Cyclone III EP3C5E144 з верифікацією за допомогою Quartus.

Результати показують вузькі місця традиційних реалізацій алгоритму Томаса на ПЛІС та демонструють, що запропонований оптимізований метод забезпечує підвищення продуктивності втричі при збереженні сталої кількості виводів незалежно від розрядності операндів. Розроблена архітектура обчислювального модуля дозволяє реалізувати паралельну побітову обробку даних та підтримує модифіковану версію алгоритму Томаса, адаптовану для роботи в неавтономному режимі. Наукова новизна полягає в поєднанні надлишкового коду з неавтономними методами обчислень, що дозволяє одночасно досягти прискорення обчислень і спрощення апаратної реалізації. Практична цінність визначається застосовністю запропонованого підходу до FPGA з обмеженими ресурсами, що забезпечує ефективну реалізацію обчислювально складних алгоритмів із залежними ланцюгами операцій.

Ключові слова: ПЛІС, неавтономні обчислення, порозрядна обробка, тридіагональні системи, надлишкові системи числення.