

Information systems. Computer sciences. Issues of information security
Информационные системы. Информатика. Проблемы информационной безопасности

UDC 004.2

<https://doi.org/10.32362/2500-316X-2025-13-3-44-53>

EDN QWXGNC



RESEARCH ARTICLE

Method for designing specialized computing systems on the basis of hardware and software cooptimization

Ilya E. Tarasov[@],
Peter N. Sovietov,
Daniil V. Lulyava,
Nikita A. Duksin

MIREA – Russian Technological University, Moscow, 119454 Russia[@] Corresponding author, e-mail: tarasov_i@mirea.ru

• Submitted: 03.11.2024 • Revised: 13.02.2025 • Accepted: 20.03.2025

Abstract

Objectives. Pipelining is an effective method for increasing the clock frequency of digital circuits. At the same time, balancing the pipeline stages during circuit synthesis at the register transfer level does not yet guarantee a balanced topological implementation of such a pipeline in terms of signal propagation delays according to the selected technological basis. This is due to the specifics of the algorithms for placing and routing components of digital devices, which are not capable of optimizing solutions in a strict mathematical sense in an acceptable time. In practice, approaches for developing digital devices combine manual control of topological constraints that set general rules for placing components with automatic optimization for localized fragments of the circuit are used to obtain results close to optimal. Pipeline circuits are based on a simple connection diagram of individual stages to demonstrate the effect of using topological design constraints on their example. On the basis of pipeline structures, a number of algorithms can be implemented to effectively complement programmable processor devices and provide hardware acceleration of some tasks. The present work develops methodological recommendations for managing topological design constraints in the implementation of pipeline computing structures based on programmable logic devices (PLD) with field-programmable gate array (FPGA) architecture.

Methods. The work is based on accepted methods for designing and modeling digital systems.

Results. Based on the analysis, modifications to a 32-bit CORDIC transcendental function computation pipeline were developed. By adding design constraints on the placement of register groups corresponding to the pipeline stages a significant increase in the clock frequency can be achieved as compared to automatic placement to reduce the running time of the tracing algorithms. The resulting effect is systematically reproduced in several implemented versions of the pipeline.

Conclusions. The presented recommendations can be used to control the clock frequency and number of stages of pipeline computing structures while simultaneously reducing the time of one iteration and routing of a module based on PLD with FPGA architecture.

Keywords: PLD, pipeline, constraints, CORDIC

For citation: Tarasov I.E., Sovietov P.N., Lulyava D.V., Duksin N.A. Method for designing specialized computing systems on the basis of hardware and software cooptimization. *Russian Technological Journal*. 2025;13(3):44–53. <https://doi.org/10.32362/2500-316X-2025-13-3-44-53>, <https://www.elibrary.ru/QWXGNC>

Financial disclosure: The authors have no financial or proprietary interest in any material or method mentioned.

The authors declare no conflicts of interest.

НАУЧНАЯ СТАТЬЯ

Управление топологическими ограничениями при реализации конвейерных вычислительных структур на базе программируемых логических интегральных схем

И.Е. Тарасов[@],
П.Н. Советов,
Д.В. Люлява,
Н.А. Дуксин

МИРЭА – Российский технологический университет, Москва, 119454 Россия

[@] Автор для переписки, e-mail: tarasov_j@mirea.ru

• Поступила: 03.11.2024 • Доработана: 13.02.2025 • Принята к опубликованию: 20.03.2025

Резюме

Цели. Конвейеризация является эффективным приемом повышения тактовой частоты цифровых схем. При этом балансировка стадий конвейера при синтезе схемы на уровне регистровых передач еще не гарантирует сбалансированную по задержкам распространения сигнала топологическую реализацию такого конвейера в выбранном технологическом базисе. Это обусловлено спецификой алгоритмов размещения и трассировки компонентов цифровых устройств, которые не позволяют получать оптимальные решения в строгом математическом смысле за приемлемое время. В практике разработки цифровых устройств применяются подходы, основанные на комбинации ручного управления топологическими ограничениями, задающими общие правила размещения компонентов, и автоматической оптимизации для локализованных фрагментов схемы, которая в этом случае позволяет получать результаты, близкие к оптимальным. Конвейерные структуры имеют простую схему соединений отдельных стадий, что позволяет продемонстрировать на их примере эффект от применения топологических проектных ограничений. В то же время, на базе конвейерных структур возможна реализация ряда алгоритмов, эффективно дополняющих программируемые процессорные устройства и обеспечивающие аппаратное ускорение некоторых задач. Цель работы – разработка методических рекомендаций по управлению топологическими проектными ограничениями при реализации конвейерных вычислительных структур на базе программируемых логических интегральных схем (ПЛИС) с архитектурой field-programmable gate array (FPGA).

Методы. Использованы методы проектирования и моделирования цифровых систем.

Результаты. На основе проведенного анализа разработаны модификации конвейерного вычислителя 32-разрядного преобразования CORDIC для вычисления трансцендентных функций. Установлено, что добавление проектных ограничений по размещению групп регистров, соответствующих стадиям конвейера, позволяет существенно повысить тактовую частоту по сравнению с автоматическим размещением

и уменьшить время работы алгоритмов трассировки. Полученный эффект систематически воспроизводится в нескольких реализованных вариантах конвейера.

Выводы. Рассмотренные рекомендации позволяют управлять тактовой частотой и количеством стадий конвейерных вычислительных структур при одновременном уменьшении времени одной итерации размещения и трассировки модуля на базе ПЛИС.

Ключевые слова: ПЛИС, конвейер, проектные ограничения, CORDIC

Для цитирования: Тарасов И.Е., Советов П.Н., Люлява Д.В., Дуксин Н.А. Управление топологическими ограничениями при реализации конвейерных вычислительных структур на базе программируемых логических интегральных схем. *Russian Technological Journal*. 2025; 13(3):44–53. <https://doi.org/10.32362/2500-316X-2025-13-3-44-53>, <https://www.elibrary.ru/QWXGNC>

Прозрачность финансовой деятельности: Авторы не имеют финансовой заинтересованности в представленных материалах или методах.

Авторы заявляют об отсутствии конфликта интересов.

INTRODUCTION

High performance computing systems are designed as a combination of general purpose and specialized subsystems. At the architectural design stage, it is necessary to identify tasks to be solved by specialized subsystems that complement the work of general purpose processors. Such identified tasks should be both in high demand and either too inefficient to be solved by the central processing unit or represent an unacceptable load. In digital electronic devices and systems, algorithms for digital signal processing [1], calculation of hash functions in information protection subsystems [2], acceleration of artificial neural networks [3], etc., are often implemented based on specialized computing devices. In the present paper, approaches to the design of a pipeline calculator are considered on a number of examples.

When developing a computing device that transforms the input vector $\vec{x}$ into the output vector $\vec{y}$, the transformation of the function given in the high-level input language into a sequence of actions is performed at each stage of the transformation. For this purpose, a synthesizer developed in the Specialized Computer Systems Laboratory at RTU MIREA is used [4]. The output of the synthesizer comprises a text in the hardware description language, which forms registers at the stages of the pipeline and combinational logic nodes between them to perform the transformations $f_1, f_2, f_3, \dots, f_n$. A similar approach is used in a number of synthesizers [5]¹. However, for the developed software product there is a possibility of synthesis control based on feedback formed by analyzing the results of component placement and tracing. In this case, the maximum value of the signal propagation delay between the stages of the pipeline determines the minimum

period of the clock frequency signal. The components of this delay should be determined for the topological basis in such a way that the synthesizer can evenly distribute the signal propagation delay between the pipeline stages.

The signal propagation delay between registers of a programmable logic device (PLD) using field programmable gate array (FPGA) architecture is defined as follows²:

$$t = t_{\text{logic}} + t_{\text{route}}, \quad (1)$$

where t_{logic} is the propagation delay determined by the combinational elements; t_{route} is the propagation delay determined by the PLD trace circuitry.

For achieving high clock frequency and uniform distribution of total signal delay between all stages of the pipeline, the synthesizer should evaluate the components defined by the combinational elements as well as those defined by the trace circuits. Once the signal transformations have been distributed to the combinational logic nodes, the resulting register transfer level (RTL) representation of the pipeline is passed to the PLD computer-aided design (CAD) system, which performs the placement of the circuit components and tracing of the interconnects. In this case, suboptimal component placement introduces additional signal delay that violates the uniformity of delay distribution across the pipeline stages.

The stages in the development of a pipeline computing device are shown in Fig. 1.

In order to achieve a high clock frequency of the pipeline operation, it is necessary to estimate the components of the signal propagation delays and eliminate the negative effects of suboptimal mutual placement of the interconnected components on the PLD chip. While the placement optimization problem

¹ <https://docs.amd.com/r/en-US/ug1399-vitis-hls/HLS-Programmers-Guide>. Accessed October 10, 2024.

² <https://docs.xilinx.com/r/en-US/ug906-vivado-design-analysis/Timing-Analysis>. Accessed October 10, 2024.

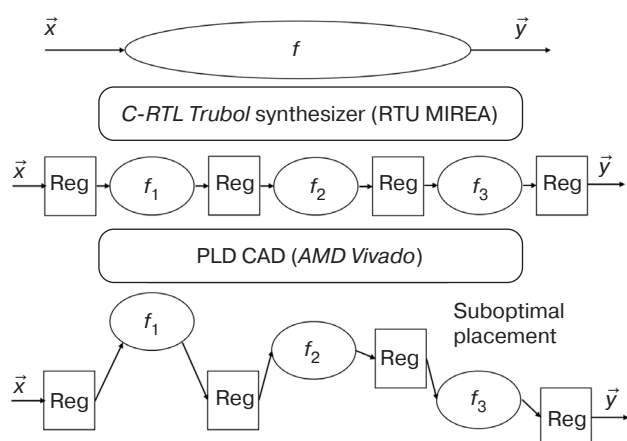


Fig. 1. The stages in the development of a pipelined computing device.

The *C-RTL Trubol* synthesizer is a software developed by the authors. The synthesizer is a tool for creating a description in the electronic CAD format

can be formulated in the strict mathematical sense, it has no practical solution in an acceptable time due to the burgeoning complexity of optimization algorithms for a general formulation. In practice, the PLD-based design uses design constraints that specify the areas for placing groups of components (so-called topological design constraints). For groups of components (called P-blocks) placed in this way, optimization by CAD algorithms can be performed in a reasonable time but with suboptimal results. Technical methods for controlling design constraints are specified in the AMD UltraFast™ Design Methodology Guide for FPGAs and Systems-on-a-Chip³. Research into the use of topological design constraints as reflected in a number of publications and dates back to 2011 [6] is driven by the development of design tools such as *Xilinx PlanAhead* [7, 8]. At present, the use of design constraints is still being used in network packet processing [9] and digital signal processing [10].

CONTROL OF THE SYNTHESIS OPERATIONS OF THE PIPELINE FUNCTIONAL UNITS

We consider the following sequence of design of a pipelined computational structure. A set of test pipeline chains containing nodes with the appropriate logic function is created to estimate the delay determined by combinational logic. For these nodes, the pipeline is synthesized and placed, and the experimental estimate is written into the data structure passed to the *C-RTL* synthesizer. As the synthesizer is an original design, appropriate modifications are introduced to account for delays introduced by external sources.

³ <https://docs.amd.com/r/en-US/ug949-vivado-design-methodology>. Accessed October 10, 2024.

It should be noted that the use of PLD does not require the evaluation of a wide range of possible arithmetic and logic operations. For this purpose, since bitwise operations are performed based on truth tables, while addition and subtraction operations can be carried out using special “fast carry chain” nodes, it is sufficient to estimate the delay caused by these two classes of operations.

Given an architectural pattern and certain delay parameters, the design sequence of the pipeline computing structure shown in Fig. 2 can be adopted.

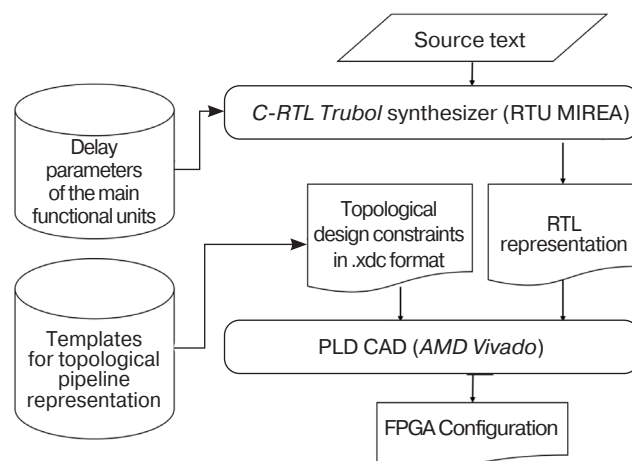


Fig. 2. The sequence of the automated design of the pipeline computing structure

The input to the developed sequence is assumed to be the program source code in a problem-oriented high-level language, as well as the design constraints formed on the basis of the study of the hardware platform characteristics. The synthesis uses the delay parameters of the main functional nodes that have been preliminarily evaluated in the process of pipeline synthesis according to a predetermined scheme that explicitly distinguishes the circuit fragments for evaluation. The dedicated synthesizer generates an RTL representation of the module in the hardware description language, which is complemented by the design constraints file in .xdc or .sdc format (depending on the CAD system used). This file is created by parameterizing one of the developed templates for the topological pipeline representation.

FORMATION OF TOPOLOGICAL DESIGN CONSTRAINTS FOR PIPELINE COMPUTATION STRUCTURE

When placing a pipeline in PLD, it is necessary to specify the rules for placing its individual components. In PLD CAD systems, it is common practice to have a hierarchical (modular) placement mode whose placement algorithms use the project modules at the RTL level as localized project units to place their components

as compactly as possible. At the same time, the designer can accompany the project with special project constraint files that control the processes of the Implementation group, such as timing analysis and placement. The corresponding constraint language command groups deal with timing constraints and topological (area) constraints.

The command for describing design constraints within a single stage has the following form:

```
create_pb <pblock name>
< X-axis pblock coordinate >
< Y-axis pblock coordinate >
< pblock width> < pblock height>
< list of elements associated with pblock>
```

In this example, PLD on-chip boundaries with specified coordinates are set for triggers whose names match the template (Figure 3). The size of the on-chip boundaries and coordinates are precisely matched to the overall topology of the chip along with the number of registers and logic elements involved in each stage. The name template is selected based on the RTL level description format.

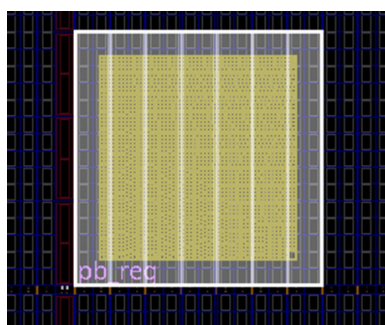


Fig. 3. The process of the P-block allocation in PLD using FPGA architecture

The study confirms the assumed utility of describing topological design constraints only for pipeline registers. This is due to the local relation of the combinational logic between the register groups and corresponding pipeline stages. By defining the pipeline stage placement regions, a compact placement of the combinational logic nodes associated with the registers is achieved while preserving the CAD capability to perform local optimizations. In this case, the manual control of separate pipeline components, including separate triggers and combinational logic nodes, would be excessively labor-intensive.

For the circuit synthesizer developed in RTU MIREA in RTL format, it is recommended to modify it by generating register names with the introduction of fragments that uniquely identify the pipeline stage to which this register belongs. Although this information is available in the internal representation of the synthesizer, it has not yet been used. Analysis of world

analogs has confirmed the impossibility of identifying the pipeline stage is impossible in them; consequently, the exported RTL representation typically uses end-to-end numbering of separate circuit triggers. At the same time, the introduction of information about the pipeline stage allows the allocation of a group of registers corresponding to this stage using a regular expression having the following form:

```
*/pipeline_unit/*reg_Ki\_*
```

EXAMPLES OF PRACTICAL TESTING OF THE METHOD

The practical testing of the method is carried out through the implementation of several types of pipelines. For example, sequential application of the vector rotation operation is used in the CORDIC⁴ algorithm [11]. Similar operations in the form of a combination of addition and shift are used in successive multiplication with accumulation. These operations can be combined in a single configurable pipeline. The schematic fragment generated in *Vivado*⁵ CAD shown in Fig. 4 demonstrates the possibility of obtaining a locally optimal solution by automatic placement of pipeline components followed by their optimization.

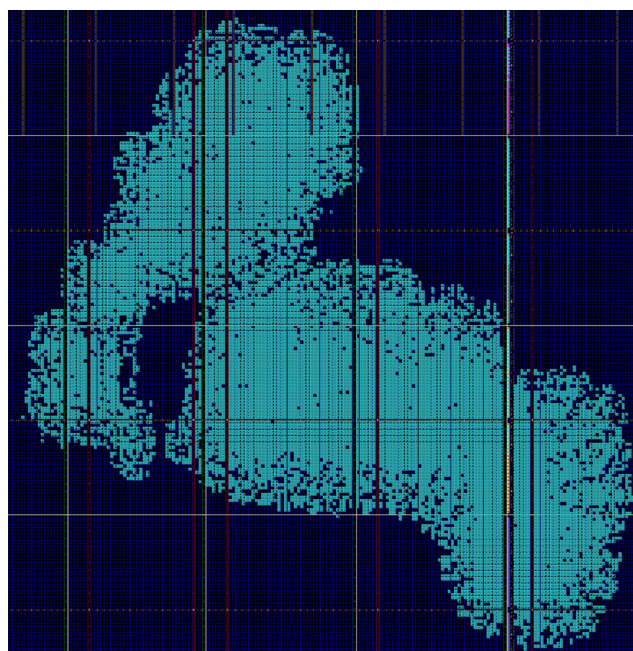


Fig. 4. Pipeline stage placement in automated mode in *Vivado* CAD

From a CAD point of view, the search for an optimal solution does not take into account the stage partitioning inherent in the pipeline architecture. Based on this

⁴ CORDIC is an acronym for COordinate Rotation DIgital Computer; a “digit by digit” method.

⁵ <https://www.amd.com/en/products/software/adaptive-socs-and-fpgas/vivado.html>. Accessed October 10, 2024.

assumption, the placement characteristics can be improved using the approach described above. The experimental results confirm the possibility of improving basic circuit performance when the block boundaries of each stage partially overlap the boundaries of adjacent blocks. The result obtained using this strategy is shown in Fig. 5.

In comparison to the standard placement (Fig. 6), a clock frequency of 1 GHz can be achieved by placement in an AMD FPGA Virtex™ UltraScale™+xcvu440_CIV-flga2892-3-e⁶ (Fig. 7).

When the number of stages of the CORDIC algorithm is increased, the result of the approach becomes even

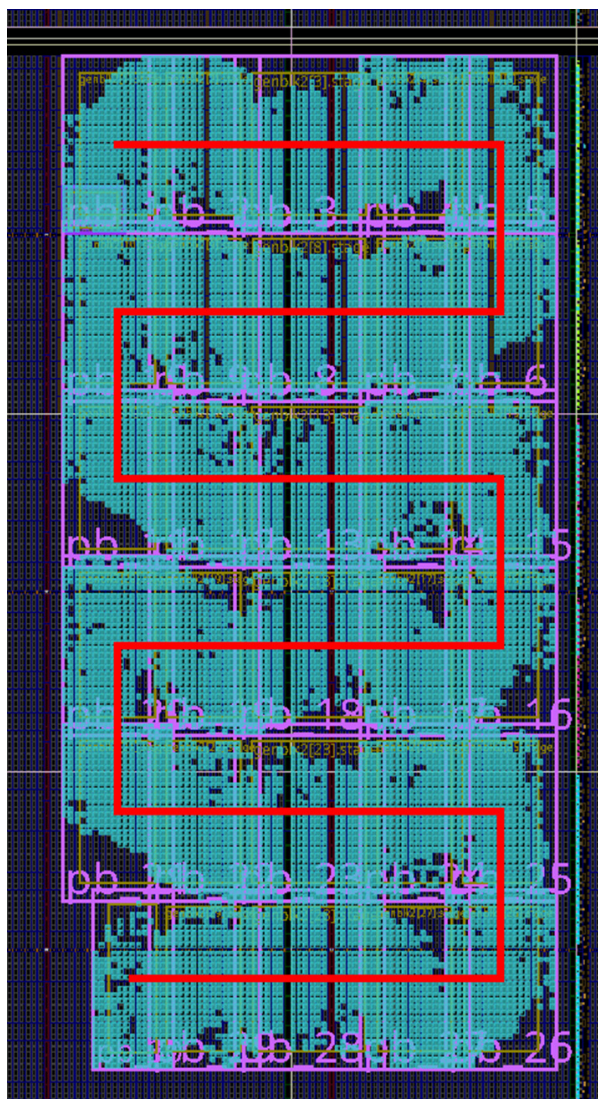


Fig. 5. Pipeline stage placement of using topological design constraints

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): -0.179 ns	Worst Hold Slack (WHS): 0.016 ns	Worst Pulse Width Slack (WPWS): -0.176 ns
Total Negative Slack (TNS): -45.105 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): -0.176 ns
Number of Failing Endpoints: 970	Number of Failing Endpoints: 0	Number of Failing Endpoints: 1
Total Number of Endpoints: 96075	Total Number of Endpoints: 96075	Total Number of Endpoints: 94415

Fig. 6. CAD report snippet with project time characteristics for the circuit solution obtained in auto mode

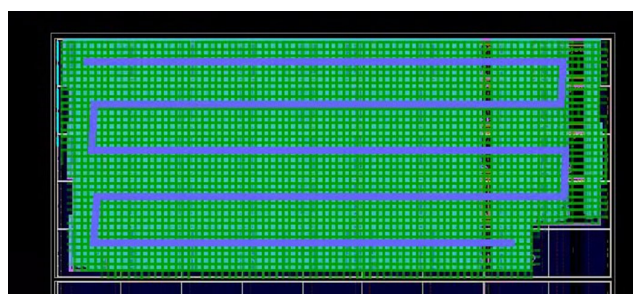
⁶ <https://www.xilinx.com/products/boards-and-kits/1-66ql3z.html>. Accessed October 10, 2024.

Design Timing Summary

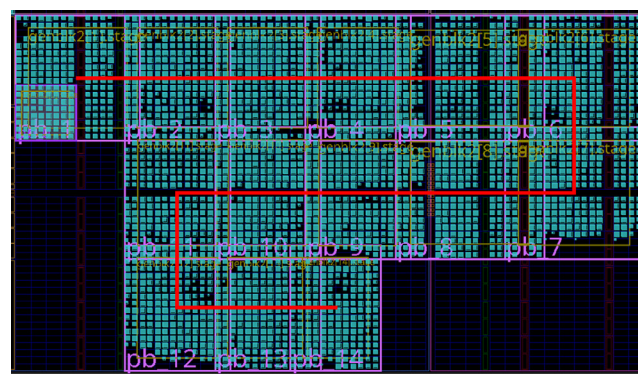
Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 0.005 ns	Worst Hold Slack (WHS): 0.018 ns	Worst Pulse Width Slack (WPWS): -0.176 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): -0.176 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 1
Total Number of Endpoints: 142287	Total Number of Endpoints: 142287	Total Number of Endpoints: 140609

Fig. 7. CAD report snippet with project time characteristics for a circuit solution using topological design constraints

more pronounced. The results for 64 stages at a clock frequency of 600 MHz are shown in Figs. 8 and 9.

Fig. 8. Pipeline stage placement
(64 stages, 600 MHz clock frequency)

A similar approach is applied to the placement of the considered circuit on the AMD FPGA Artix-7 xc7a100tcs324-1⁷ (16 stages, 400 MHz clock frequency) and AMD FPGA Kintex™ UltraScale™ xcku115_CIV-flvf1924-3-e⁸ (32 stages, 850 MHz clock frequency) chips. The viability of the approach is demonstrated by the comparison of timing analysis results for the circuits under consideration (Figs. 10–13).

Fig. 10. Pipeline stage placement using topological
design constraints (xc7a100tcs324-1 PLD)

Design Timing Summary

Setup	Hold
Worst Negative Slack (WNS): 0.024 ns	Worst Hold Slack (WHS): 0.010 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 16911	Total Number of Endpoints: 16911

All user specified timing constraints are met.

Fig. 11. CAD report snippet with project
time characteristics for the obtained
placement (xc7a100tcs324-1 PLD)

Design Timing Summary

Setup	Hold	Pulse Width
Worst Negative Slack (WNS): 0.005 ns	Worst Hold Slack (WHS): 0.016 ns	Worst Pulse Width Slack (WPWS): 0.100 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns	Total Pulse Width Negative Slack (TPWS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 780243	Total Number of Endpoints: 780243	Total Number of Endpoints: 775890

All user specified timing constraints are met.

Fig. 9. CAD report snippet with time characteristics of the CORDIC pipeline computer project
(64 stages, 600 MHz clock frequency)

⁷ <https://www.amd.com/en/products/adaptive-socs-and-fpgas/fpga/artix-7.html>. Accessed October 10, 2024.

⁸ <https://www.amd.com/en/products/adaptive-socs-and-fpgas/fpga/kintex-ultrascale-plus.html>. Accessed October 10, 2024.

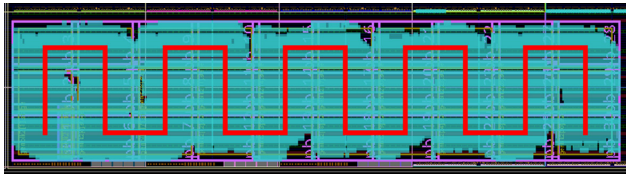


Fig. 12. Pipeline stage placement using topological design constraints (xcku115_CIV-flvf1924-3-e PLD)

Design Timing Summary

Setup	Hold
Worst Negative Slack (WNS): 0.040 ns	Worst Hold Slack (WHS): 0.030 ns
Total Negative Slack (TNS): 0.000 ns	Total Hold Slack (THS): 0.000 ns
Number of Failing Endpoints: 0	Number of Failing Endpoints: 0
Total Number of Endpoints: 142195	Total Number of Endpoints: 142195

Timing constraints are not met.

Fig. 13. CAD report snippet with project time characteristics for the obtained placement (xcku115_CIV-flvf1924-3-e PLD)

The results confirm the possibility of setting design constraints on registers of a separate module describing the pipeline to systematically improve the design properties of computers having a pipelined structure. The continuing interest in pipelined nodes [12] confirms the relevance of this research direction. In addition, pipelined devices can function as subsystems of computational complexes to increase their efficiency both in widespread tasks [13] and

when used as accelerators of a narrow computational subclass [14, 15].

CONCLUSIONS

The presented materials describe results obtained by the Specialized Computer Systems Laboratory at RTU MIREA in the development of a design methodology for specialized pipelined computing accelerators. By focusing on a simple hardware architecture with localized connections between nodes, it is possible to develop a set of algorithms and design measures for systematically improving the topological representation properties of a computing device from its initial high-level language description. The obtained results can be adapted to other types of architectural templates to extend the nomenclature of specialized electronic component bases.

ACKNOWLEDGMENTS

The research is carried out within the framework of the State Assignment of the Ministry of Science and Higher Education of the Russian Federation (No. FSFZ-2022-0004 “Architectures of special-purpose computing units and procedures, algorithms, and tools for the design of digital computing units”).

Authors' contribution

All authors equally contributed to the research work.

REFERENCES

1. Saidov B.B., Telezhkin V.F., Gudaev N.N., et al. Development of Equipment for Experimental Study of Digital Algorithms in Nonstationary Signal Processing Problems. *Ural Radio Engineering Journal*. 2022;6(2):186–204. <https://doi.org/10.15826/urej.2022.6.2.004>
2. Jasek R. SHA-1 and MD5 Cryptographic Hash Functions: Security Overview. *Communications (Komunikacie)*. 2015;17(1):73–80.
3. Carrión D.S., Prohaska V., Diez O. Exploration of TPUs for AI Applications. In: Daimi K., Al Sadoon A. (Eds.). *Proceedings of the Second International Conference on Advances in Computing Research (ACR'24)*. ACR 2024. Lecture Notes in Networks and Systems. Springer; 2024. V. 956. P. 559. https://doi.org/10.1007/978-3-031-56950-0_47
4. Tarasov I.E., Sovietov P.N., Lulyava D.V., Mirzoyan D.I. Method for designing specialized computing systems based on hardware and software co-optimization. *Russian Technological Journal*. 2024;12(3):37–45. <https://doi.org/10.32362/2500-316X-2024-12-3-37-45>
5. Alekhin V.A. Designing Electronic Systems Using SystemC and SystemC–AMS. *Russian Technological Journal*. 2020;8(4):79–95 (in Russ.). <https://doi.org/10.32362/2500-316X-2020-8-4-79-95>
6. Pham-Quoc C., Dinh-Duc A.-V. Automatic generation of area constraints for FPGA implementation. In: *2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*. 2011. P. 469–472. <https://doi.org/10.1109/ICCSN.2011.6014937>
7. Li K., Lei L., Guang Q., Shi J.-Y., Hao Y. Improving the performance of an SOC design for network processing based on FPGA with PlanAhead. In: *2011 International Conference on Electronics, Communications and Control (ICECC)*. 2011. P. 297–300. <https://doi.org/10.1109/ICECC.2011.6066640>
8. Sarker A.L.Md., Lee M.H. Synthesis of VHDL code for FPGA design flow using Xilinx PlanAhead tool. In: *2012 International Conference on Education and e-Learning Innovations (ICEELI)*. 2012. <https://doi.org/10.1109/ICEELI.2012.6360614>

9. Song X., Lu R., Guo Z. High-Performance Reconfigurable Pipeline Implementation for FPGA-Based SmartNIC. *Micromachines*. 2024;15(4):449. <https://doi.org/10.3390/mi15040449>
10. Anderson T., Wheeler T.J. An FPGA-based hardware accelerator supporting sensitive sequence homology filtering with profile hidden Markov models. *BMC Bioinformatics*. 2024;25:247. <https://doi.org/10.1186/s12859-024-05879-3>
11. Tarasov I.E., Sovetov P.N. *Device for Calculating Transcendental Functions and Multiplying Binary Numbers*: Pat. 222880 U1 RF. Publ. 22.01.2024 (in Russ.).
12. Oishi R., Kadomoto J., Irie H., Sakai S. FPGA-based Garbling Accelerator with Parallel Pipeline Processing. *IEICE Trans. Inform. Syst.* 2023;E106.D(12):1988–1996. <https://doi.org/10.1587/transinf.2023PAP0002>
13. Nurvitadhi E., Sheffield D., Sim J., et al. Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC. In: *2016 International Conference on Field-Programmable Technology (FPT)*. 2016. P. 77–84. <https://doi.org/10.1109/FPT.2016.7929192>
14. Hennessy J.L., Patterson D.A. A new golden age for computer architecture: Domain-specific hardware/software co-design, enhanced security, open instruction sets, and agile chip development. In: *Proceedings of the 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. 2018. P. 27–29. <https://doi.org/10.1109/ISCA.2018.00011>
15. Hennessy J.L., Patterson D.A. *Computer Architecture: A Quantitative Approach*: 6th ed. The Morgan Kaufmann Series in Computer Architecture and Design. 2017. 936 p.

СПИСОК ЛИТЕРАТУРЫ

1. Saidov B.B., Telezhkin V.F., Gudaev N.N., et al. Development of Equipment for Experimental Study of Digital Algorithms in Nonstationary Signal Processing Problems. *Ural Radio Engineering Journal*. 2022;6(2):186–204. <https://doi.org/10.15826/urej.2022.6.2.004>
2. Jasek R. SHA-1 and MD5 Cryptographic Hash Functions: Security Overview. *Communications (Komunikacie)*. 2015;17(1):73–80.
3. Carrión D.S., Prohaska V., Diez O. Exploration of TPUs for AI Applications. In: Daimi K., Al Sadoon A. (Eds.). *Proceedings of the Second International Conference on Advances in Computing Research (ACR'24)*. ACR 2024. Lecture Notes in Networks and Systems. Springer; 2024. V. 956. P. 559. https://doi.org/10.1007/978-3-031-56950-0_47
4. Тарасов И.Е., Советов П.Н., Люлява Д.В., Мирзоян Д.И. Методика проектирования специализированных вычислительных систем на основе совместной оптимизации аппаратного и программного обеспечения. *Russian Technological Journal*. 2024;12(3):37–45 <https://doi.org/10.32362/2500-316X-2024-12-3-37-45>
5. Алехин В.А. Проектирование электронных систем с использованием SystemC и SystemC–AMS. *Russian Technological Journal*. 2020;8(4):79–95. <https://doi.org/10.32362/2500-316X-2020-8-4-79-95>
6. Pham-Quoc C., Dinh-Duc A.-V. Automatic generation of area constraints for FPGA implementation. In: *2011 IEEE 3rd International Conference on Communication Software and Networks (ICCSN)*. 2011. P. 469–472. <https://doi.org/10.1109/ICCSN.2011.6014937>
7. Li K., Lei L., Guang Q., Shi J.-Y., Hao Y. Improving the performance of an SOC design for network processing based on FPGA with PlanAhead. In: *2011 International Conference on Electronics, Communications and Control (ICECC)*. 2011. P. 297–300. <https://doi.org/10.1109/ICECC.2011.6066640>
8. Sarker A.L. Md, Lee M.H. Synthesis of VHDL code for FPGA design flow using Xilinx PlanAhead tool. In: *2012 International Conference on Education and e-Learning Innovations (ICEELI)*. 2012. <https://doi.org/10.1109/ICEELI.2012.6360614>
9. Song X., Lu R., Guo Z. High-Performance Reconfigurable Pipeline Implementation for FPGA-Based SmartNIC. *Micromachines*. 2024;15(4):449. <https://doi.org/10.3390/mi15040449>
10. Anderson T., Wheeler T.J. An FPGA-based hardware accelerator supporting sensitive sequence homology filtering with profile hidden Markov models. *BMC Bioinformatics*. 2024;25:247. <https://doi.org/10.1186/s12859-024-05879-3>
11. Тарасов И.Е., Советов П.Н. *Устройство для вычисления трансцендентных функций и умножения двоичных чисел*: пат. 222880 U1 РФ. Заявка № 2023131099; заявл. 28.11.2023; опубл. 22.01.2024. Бюл. № 3.
12. Oishi R., Kadomoto J., Irie H., Sakai S. FPGA-based Garbling Accelerator with Parallel Pipeline Processing. *IEICE Trans. Inform. Syst.* 2023;E106.D(12):1988–1996. <https://doi.org/10.1587/transinf.2023PAP0002>
13. Nurvitadhi E., Sheffield D., Sim J., et al. Accelerating Binarized Neural Networks: Comparison of FPGA, CPU, GPU, and ASIC. In: *2016 International Conference on Field-Programmable Technology (FPT)*. 2016. P. 77–84. <https://doi.org/10.1109/FPT.2016.7929192>
14. Hennessy J.L., Patterson D.A. A new golden age for computer architecture: Domain-specific hardware/software co-design, enhanced security, open instruction sets, and agile chip development. In: *Proceedings of the 2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. 2018. P. 27–29. <https://doi.org/10.1109/ISCA.2018.00011>
15. Hennessy J.L., Patterson D.A. *Computer Architecture: A Quantitative Approach*: 6th ed. The Morgan Kaufmann Series in Computer Architecture and Design. 2017. 936 p.

About the Authors

Ilya E. Tarasov, Dr. Sci. (Eng.), Associated Professor, Head of the Laboratory of Specialized Computing Systems, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: tarasov_i@mirea.ru. Scopus Author ID 57213354150, RSCI SPIN-code 4628-7514, <http://orcid.org/0000-0001-6456-4794>

Peter N. Sovietov, Cand. Sci. (Eng.), Senior Researcher, Laboratory of Specialized Computing Systems, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: peter.sovietov@gmail.com. Scopus Author ID 57221375427, RSCI SPIN-code 9999-1460, <http://orcid.org/0000-0002-1039-2429>

Daniil V. Lulyava, Junior Researcher, Laboratory of Specialized Computing Systems, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: lyulyava@mirea.ru. Scopus Author ID 58811698000, RSCI SPIN-code 1882-0989, <http://orcid.org/0009-0009-9623-7777>

Nikita A. Duksin, Engineer, Laboratory of Specialized Computing Systems, MIREA – Russian Technological University (78, Vernadskogo pr., Moscow, 119454 Russia). E-mail: duksin@mirea.ru. RSCI SPIN-code 1082-8956, Scopus Author ID 58811361100, <https://orcid.org/0009-0009-0014-7065>

Об авторах

Тарасов Илья Евгеньевич, д.т.н., доцент, заведующий лабораторией специализированных вычислительных систем, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: tarasov_i@mirea.ru. Scopus Author ID 57213354150, SPIN-код РИНЦ 4628-7514, <http://orcid.org/0000-0001-6456-4794>

Советов Петр Николаевич, к.т.н., старший научный сотрудник, лаборатория специализированных вычислительных систем, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: peter.sovietov@gmail.com. Scopus Author ID 57221375427, SPIN-код РИНЦ 9999-1460, <http://orcid.org/0000-0002-1039-2429>

Люлява Даниил Вячеславович, младший научный сотрудник, лаборатория специализированных вычислительных систем, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: lyulyava@mirea.ru. Scopus Author ID 58811698000, SPIN-код РИНЦ 1882-0989, <http://orcid.org/0009-0009-9623-7777>

Дуксин Никита Александрович, инженер, лаборатория специализированных вычислительных систем, ФГБОУ ВО «МИРЭА – Российский технологический университет» (119454, Россия, Москва, пр-т Вернадского, д. 78). E-mail: duksin@mirea.ru. SPIN-код РИНЦ 1082-8956, Scopus Author ID 58811361100, <https://orcid.org/0009-0009-0014-7065>

Translated from Russian into English by K. Nazarov

Edited for English language and spelling by Thomas A. Beavitt