DOI: 10.14529/jsfi230205

Elements of a Digital Photonic Computer

Dmitriy A. Sorokin[1], *Aleksey V. Kasarkin*[1], *Aleksandr V. Podoprigora*[1]

The paper covers a variant of architecture development of digital photonic computers. Along with quantum computers, they are one of the possible ways to overcome the crisis of computing performance. The data processing implementation in digital photonic computers at terahertz frequencies potentially provides the performance exceeding by two or more decimal orders of magnitude the performance of the most modern computing systems. The elements of digital photonic computer architecture described in the paper are focused on solving a wide class of computationally time-consuming problems in the paradigm of structural calculations. The problems of ensuring the performance and accuracy at solving problems on the developed digital photonic computer, as the processing rate should correspond to the data receipt rate, are considered. The synchronization and switching subsystem was designed and analyzed by the authors. At the synthesis (programming) stage, it forms a computational structure and provides both static and dynamic coordination of data flows in calculations.

Keywords: digital photonic computer, architecture of DPC, data flow synchronization, paradigm of structural calculations.

Introduction

The successful scientific researches and technical developments are closely related to the availability of high-performance computing systems, as well as the possibility of timely increasing the speed and quality of solving time-consuming problems. However, the development of high-performance systems element base is slowing down [1]. In modern microelectronics, this is due to the achievement of physical limits for increasing clock frequencies and the integration degree [2], and in actively promoted quantum computing [3] – technological problems of system isolating the from "white noise", poor repeatability and accuracy of experiments [4, 5].

A possible way to overcome the crisis of computer technology performance may be digital photonic computer (DPCs). These are devices, in which calculations are performed using a light flux emitted by a laser. This is similar to the electric current generated by a generator in modern microelectronics. At the same time, to ensure high performance and accuracy of calculations at solving the problems, it is advisable to develop a fully digital photonic computer. In it, the data processing is performed by photonic logic gates, such as NOT, AND, OR [6, 7], as well as triggers and functional devices, based on it. In addition, it is important to choose such architecture of DPC, which, in the absence of immediate prospects for creating photonic memory [8–11], eliminates the traditional "bottleneck" problems, and will ensure the correspondence of data transmission frequencies and the transformation speed.

This paper is devoted to the description of an architecture of DPC proposed by the authors and one of its main components – the switching and synchronization subsystem between functional devices, as well as communication system between DPCs and external data sources and receivers. The first section contains a brief overview of modern achievements in photonic calculators and an assessment of the possibility of their application at solving time-consuming problems. The second section contains a brief description of the DPC proposed architecture with the structural organization of calculations. The development principles of the synchronization and data flows switching subsystem are described. The third section presents a detailed analysis

[1]Supercomputers and Neurocomputers Research Center, Taganrog, Russia

of the main component of the synchronization and data flows switching subsystem – the operand block. Its structure, operation modes are considered. The fourth section gives the estimation of the DPC expected efficiency at solving problems of mathematical physics. The obtained results were analyzed in conclusion.

1. Review of Modern Researches on Photonic Calculators

Interest to the computing system design, in which the information is transmitted by a light stream, originated in the late 50s and early 60s of the last century. The prospect of solving complex problems with near-light speed caused significant progress at the creation an appropriate element base. This led to the appearance of a separate class of devices – optical correlators [12]. The operation principle of these calculators is based on the comparison of signals using the Fourier lens [13]. The data processing is performed in an analog format, which has all the advantages and disadvantages of analog machines. Therefore, since the late 80s and early 90s of the last century, it is necessary to integrate the correlators and digital computers. For example, the Bell Labs presented the first layout of an optical computer in 1990 [14]. The OptiComp introduced a DOC II 32-bit general purpose optical computer in 1991 [15]. In 2015, the ORNL laboratory conducted a number of researches to assess of the speed of solving the Fast Fourier Transform (FFT) problem on the EnLight Alpha computing system. It was based on the EnLight 256 optical processor, compared to the computing system based on 2 Intel Xeon 2 GHZ processors. The conducted researches have shown more than 13,000 times the acceleration in time at problem solution achieved on EnLight Alpha. However, as ORNL researchers note, the calculation speed is inversely dependent on accuracy [16].

Such computers are hybrid systems. In it, an analog converter performs the main calculations, and the data preparation, transmission and storage functions are performed by traditional microelectronic components. Currently, the development of optoelectronic hybrids continues in two directions: by increasing the computational characteristics of optical correlators by growing the resolution and performance of modulation tools, or by using invariant correlators in combination with data mining methods [12]. These technologies make it possible to process information with the bandwidth of tens of gigabits per second and obtain a satisfactory quality solution of image analysis problems. However, the optical correlators and systems based on them are not applicable for many modern time-consuming problems from such fields as gas dynamics and molecular dynamics, plasma physics and inertial fusion, and many others. It is required to perform processing in high-precision data representation formats. Only fully DPCs can provide high-speed computing at solving specified problems with the accuracy of the IEEE 754 standard level or similar.

Currently, the researches have being conducted largely on logic elements that perform operations on light pulses and provide a fully functional basis for a digital computer in Russia [17, 18] and abroad [19–21]. At the same time, few papers have been devoted to the problem of choosing the prospective DPC architecture and methods of organization of calculations [22, 23].

The structure and principles of the DPC implementation are considered in [22], which correspond to the reduction calculation model. The reduction model forms a flow graph of data processing by recursive analysis of the algorithm for problem solution. The sequence of graph operations is dynamically mapped to the system computing resource. Therefore, data is constantly transmitted over the switching network from one functional device (FD) of the system to another. RAM is not required to store intermediate calculations.

In this case, one FD either performs the requested operation, or stores one valid data. The search for exchange paths between FDs, implementing the graph operations, is performed constantly.

However, conflicts and, accordingly, temporary losses are inevitable at exchanging between FDs. This can only be eliminated by a fully connected switching system that requires large equipment costs. If the switching system is not fully connected, then the most of the FD will be occupied not with performing direct calculations, but with storing intermediate results and transferring them to other FD for subsequent calculations at solving the real applied problem. Computing equipment will be used inefficiently.

The paper [23] describes the architecture of DPC based on the paradigm of structural calculations [24]. In structural calculations, the FDs perform only informationally significant transformations, and pipeline data processing is performed at the rate of their entering at the FD inputs. All informationally insignificant operations, such as the exchange or data source selection, are implemented by spatial switching and synchronization. The processing results are transferred through physical channels to the following FDs, according to the task information graph [25], and are not buffered in memory. This allows minimizing the memory for storing the results of intermediate calculations and reducing the bandwidth requirements for photonic-electronic data exchange interfaces with external devices. The proposed solutions are aimed at efficient use of the available computing resource and preserving the advantage of high data processing frequency in the DPC over microelectronic devices.

2. Architecture of DPC with the Paradigm of Structural Calculations

Within the framework of the paradigm of structural calculations, the high efficiency can be achieved with the equality (consistency) of data exchange rate between all system components at solving time-consuming problems: RAM and system computing nodes, the data processing rate in the nodes and transmission between them.

The DPC data processing is assumed at the frequency about 1 THz [7]. Modern microelectronics technologies provide exchange with RAM at the frequency about 1 GHz. Therefore, the stream must be multiplexed for each channel bit between the RAM and DPC. Figure 1 shows a possible variant of such structure.

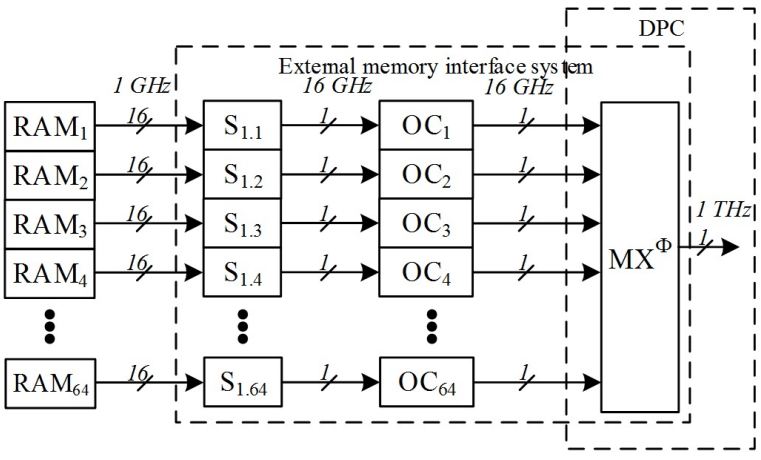

Figure 1. The proposed structure of data flow formation for DPC

RAM can be developed on SDRAM-type chips, for example. Their quantity is determined by the ratio of the required bandwidth of the digital computer channel to the bandwidth of the memory chip channel. Using DDR3 memory with 16-bit channels and the 1 GHz transmission frequency, it is necessary to use 64 chips to provide the stream about 1 Tbit/s (RAM_1–RAM_{64}).

At connection to the DPC, the electronic channels of RAM_1–RAM_{64} chips must be converted into photonic ones using the interface system with external memory. In general, such system is a set of cascades of serializers S_1–S_{64} by the MGT type, optical converters OC_1–OC_{64} by the SFP+ type and MX^{Φ} multiplexer, based on photonic logic. The similar solution can be used to develop data transmission channels of arbitrary digit capacity in DPC. The organization of output channels from the DPC into RAM will require the construction of a similar system that performs transformations in reverse order.

The transformed input data streams are processed in the DPC by a computational structure synthesized on a set of arithmetical and logical FD_1–FD_L (Fig. 2). The switching and synchronization subsystem must ensure a uniform rate of data transfer between the FD involved in processing, as well as between the DPC and external memory. Since it is proposed to solve problems in the paradigm of structural calculations, the switching and synchronization subsystem should have the high connectivity. However, the construction of the most efficient, fully connected switches involves a factorial increase in hardware costs at linear increasing of FB. This will lead to the use of most of the photonic logic for switching instead of calculations, i.e., to the violation of the basic principle of the paradigm of structural calculations: the use of most of the system hardware resource directly for calculations and a smaller part for organization of information-insignificant operations.

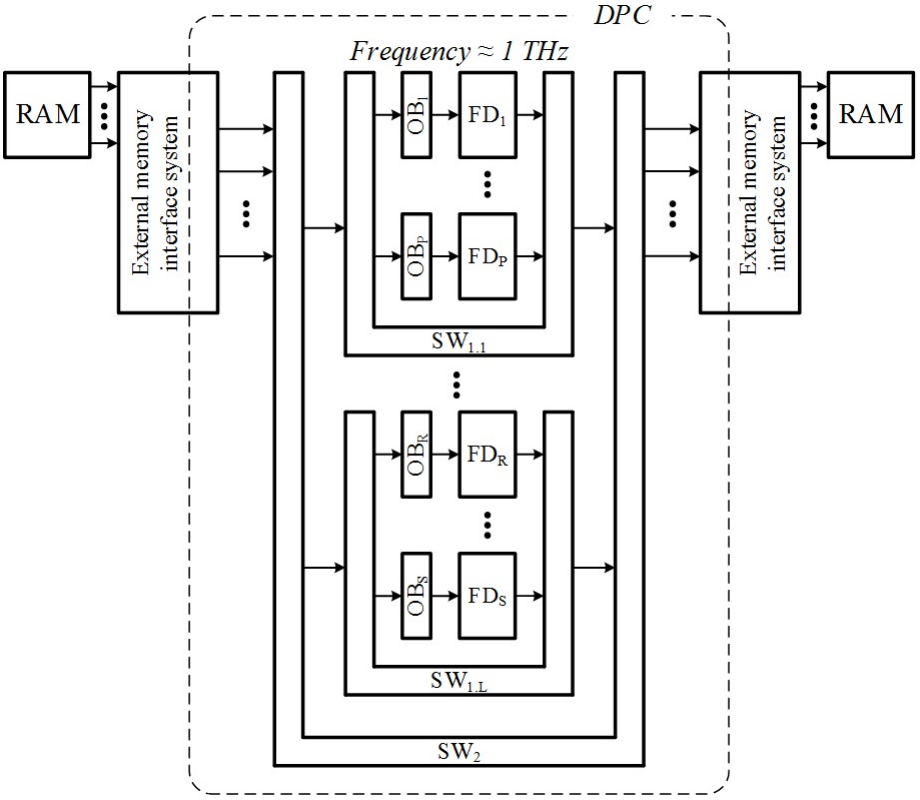

Figure 2. DPC architecture

Therefore, the switching and synchronization subsystem has a hierarchical topology in the proposed version of the DPC construction [23]. It provides more efficient use of hardware resources at solving various problems that differ by the number of involved FD and the connectivity between them. Figure 2 shows the proposed architecture of DPC.

The switching and synchronization subsystem consists of the SW static switches and blocks of dynamic switching and synchronization of operand streams (hereinafter – operand blocks, OB).

The SW static switches are devices of the "multiplexer/demultiplexer" type. Owning to it, the computational structure is synthesized on the set of FD in accordance with the task information graph at the DPC programming stage [26]. For this, the configuration parameters formed at the stage of DPC program transmission must be submitted on the control inputs of static switches. Configuration parameters cannot be changed at solving the problem.

Let us consider a variant of a computing structure, synthesized on a certain DPC fragment (Fig. 3). It implements the calculation function of a random variable dispersion

$$D = \frac{\sum_{i=1}^{n} x_i^2}{n} - \left(\frac{\sum_{i=1}^{n} x_i^2}{n} \right)^2.$$

This function is used at statistical analysis of big data, training and execution of artificial neural networks, forecasting the financial markets situation and others. Three adders, two multipliers and two dividers are necessary for construction of the pipeline computing structure that fully corresponds to the information graph of calculation D.

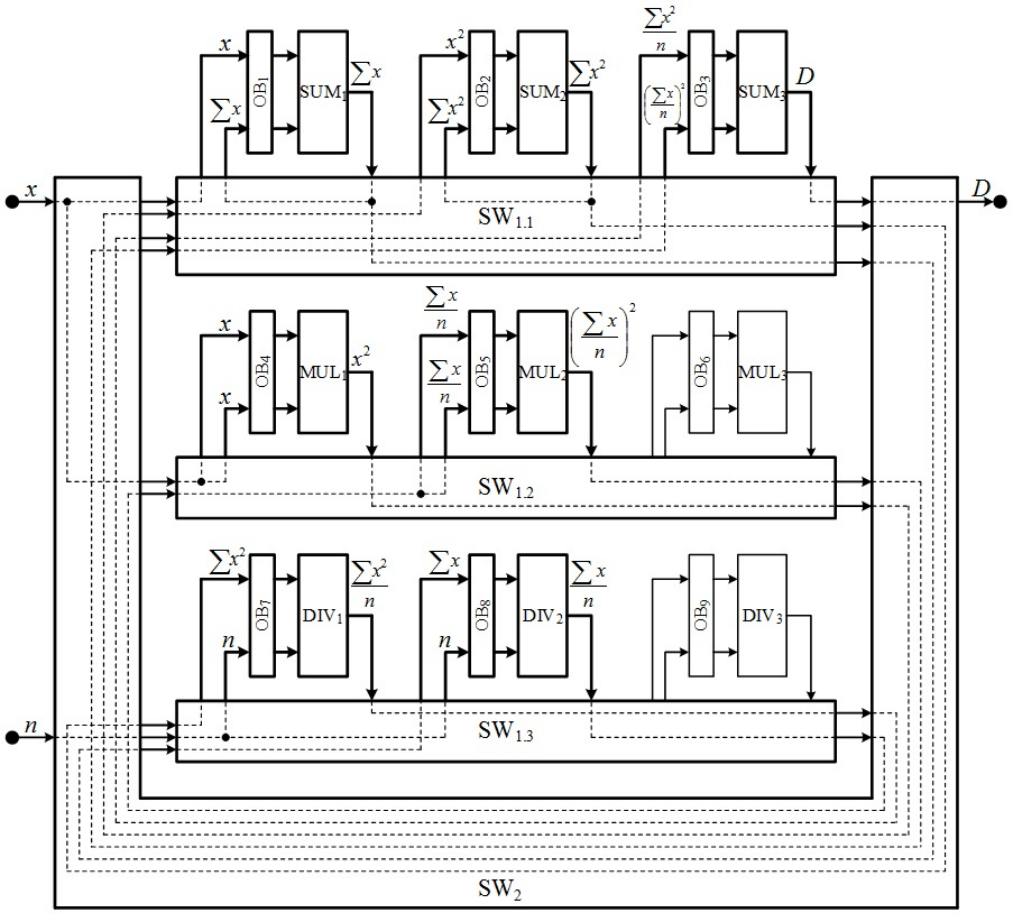

Figure 3. Computational structure of the function D

According to the Fig. 3, the selected DPC fragment contains three static switches $SW_{1.1}$–$SW_{1.3}$ of the hierarchy first level, the static switch SW_2 of the hierarchy second level, operand blocks OB_1–OB_9 and an FD array divided into three groups: SUM_1–SUM_3 adders, MUL_1–MUL_3 multipliers and DIV_1–DIV_3 dividers.

The information channels formed during the DPC programming in the static switches $SW_{1.1}$–$SW_{1.3}$ and SW_2 are shown by dotted lines.

The blocks OB_1–OB_3 are configured to operate in synchronization mode and operand streams' switching at inputs SUM_1–SUM_3. OB_4 and OB_5 are configured to operate in switching mode of operand streams at inputs MUL_1 and MUL_2. The blocks OB_7 and OB_8 are configured to operate in switching mode operand streams and the register variable n, which does not change during calculations of the current dispersion value at the inputs DIV_1 and DIV_2.

Note that the latency of various arithmetic logic FD, as well as the out of sync between different fragments of the computational structure may differ many times, in general. Therefore, it is necessary to be able to synchronize threads through external memory, both at the synthesis (programming) stage and at the DPC operation in operand blocks.

3. Data Flow Synchronization Management. Operand Block

Without violating the generality, we assume that all operations in the DPC computing structures are double. They are divided into two groups: operations with constants and threads. The necessary synchronizing component configuration is performed during the synthesis of computing structures on DPC or during data processing in OBs. The operand stream can be synchronized either with a register variable (constant), with an array of constants, or with another stream. The operand block structure OB_i is given in Fig. 4.

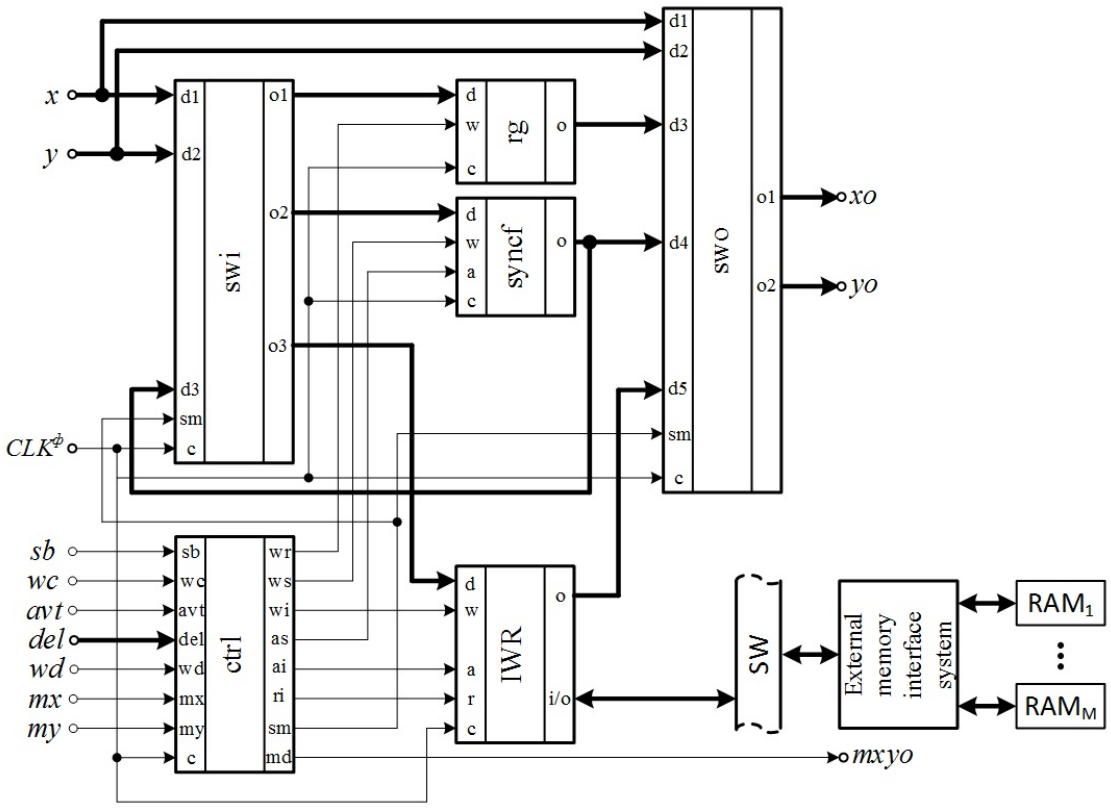

Figure 4. OB structure

The operand block consists of *swi* and *swo* dynamic switches, the *rg* variable storage register, the *syncf* operand flow synchronization block, the *IWR* external memory access interface, and the *ctrl* control block. The *ctrl* control block is designed to calculate the out of sync value, automatic or forced control of operating modes and the formation of appropriate control signals.

The *swi* and *swo* dynamic switches are also devices of the "multiplexer/demultiplexer" type. It are intended both for implementation of conditional transitions at solving problems and for synthesis of a computational structure. The parameters on their control inputs can be determined by the FD operation results or formed at the DPC program transmission.

The *rg* register is used in arithmetic and logical operations on a data stream and a variable that changes its value in no less time than the stream processing time. In this case, the variable digit capacity should not exceed the *rg* digit capacity. For example, the FP64 format of the IEEE 754 standard is used at solving mathematical physics problems. Therefore, the *rg* digit capacity must be at least 64 bits. A constant is written to this register at the DPC programming stage, or a variable is written and stored until a new entry is made at solving problem.

The *syncf* and *IWR* delay synchronizers in OBs operate according to the "shift register" principle and perform the data flows coordinating functions between the FDs. The latencies of different FD cannot coincide, and streams from different parts of the computational structure can arrive asynchronously. The *syncf* is used at operand streams matching, if the out of sync does not exceed 10^3 clock cycles. However, the out of sync value can be more than 10^3 clock cycles between fragments of computational structures. In this case, it is often necessary to use arrays of constants or intermediate data with the volume more than 10^3 bits at solving complex problems. It is necessary to use the *IWR* that provides access to external memory through the SW static switch and external memory interface system, described above. It is important to note that the number of simultaneously possible interfaces between different OBs and external memory is determined by the bandwidth of the memory channels of the entire DPC.

Taking into account the possible variety of synchronization scenarios, four main operating modes should be provided in the OB. In this case, the synchronization mode definition in the *ctrl* block can be performed at the DPC programming stage. For this, the *del* synchronization value is written in the *ctrl* block under the control of the *wd* signal, the value of which determines the operation mode. At the same time, the forced synchronization signal is set as $avt = 1$. If $avt = 0$, the determination of the operation mode will be performed automatically in accordance with the out of sync value between the mx and my markers of the x and y operand streams at solving the problem.

Operation modes 1, 2 or 3 are set according to the out of sync value between the operand streams. The mode 1 is set if the operand streams come synchronously or it is set forcibly at solving the problem. In this case, the input operand streams of the x and y are transferred directly to the outputs *o1* and *o2* of the *swu* switch. The mode 2 is set if the out of sync value is up to 10^3 clock cycles or forcibly at operating with arrays of constants with a volume up to 10^3 bits. The leading operand stream of x (or y) through the swi switch is transferred to $syncf$, and then with the lagging operands stream of y (or x) flows synchronously through the swi switch to the outputs. The mode 3 is set at the out of sync value and is more than 10^3 clock cycles or forcibly at operating with arrays of constants of more than 10^3 bits, stored in external memory. The mode 0 is set in operations with a constant. At the same time, during operation, the *rg* can be connected to the required output of the *swo* switch, or be disconnected, and the operand block can operate in modes 1–3.

The description of input signals and OB operation modes, corresponding to it, are given in Tab. 1.

Table 1. Error-free data transfer rates

Ctrl input signals						Mode number	Note
sb	*wc*	*avt*	*wd*	*mx*	*my*		
–	0	0	0	1	1	1	if Δ=0
–	0	0	0	1	1	2 or 3	mode 2, if $\Delta \leq 10^3$ mode 3, if $\Delta > 10^3$
0	1	0	0	1	0	–	rg=x
1	1	0	0	0	1	–	rg=y
0	0	0	0	1	0	0	yo=rg
1	0	0	0	0	1	0	xo=rg
–	1	0	1	0	0	–	write *del* to *ctrl*
–	0	1	0	1	1	2 or 3	mode 2, if *del*$\leq 10^3$ mode 3, if *del*$> 10^3$
0	1	1	0	1	0	–	write x to *syncf*, if *del*$\leq 10^3$ write x to *IWR*, if *del*$> 10^3$
1	1	1	0	0	1	–	write y to *syncf*, if *del*$\leq 10^3$ write y to *IWR*, if *del*$> 10^3$

At calculation the out of sync value, one clock cycle of machine time at the DPC frequency is taken as a measure unit. The calculation process of out of sync value is started and stopped by the mx or my marker of the leading operand x or y according to the formula:

$$Q_i = Q_{i-1} + 1, \quad if \quad mx_i \oplus my_i = 1,$$

where Q is a counter value; i is the clock cycle number of the counter at some time.

Fixing the desynchronization value Δ is performed according to the formula:

$$\Delta = Q_i, \quad if \quad (\overline{mx_{i-1}} \vee \overline{my_{i-1}})\&mx_i\&my_i = 1.$$

If only one of the x or y channels starts receiving the data into the OB, the *ctrl* is in the mode 0. At the same time, the leading data from x (or y) channel is transferred to the input $x0$ (or $y0$), and a constant from *rg* is transferred to the output $y0$ (or $x0$). In general, it is not known in advance whether the second data stream will come, so the determination process of Δ value starts automatically. Until the Q value achieves the value 10^3, the input operand stream is written by *syncf* under the signal control $ws = 1$. If the Q value becomes greater than 10^3, then the signal $wi = 1$ is generated and the input operand stream from *syncf* is rewritten to external memory via *IWR*. In the case, after i clock cycles after the arrival of the operand leading stream, the lagging stream of operands begins to arrive ($mxi = myi = 1$), then the *ctrl* sets the operating mode 2 or 3 based on the value of the out of sync value $\Delta > 0$. The *ctrl* block allows forcing the mode 2 or 3. For this, the value of the forced delay is written over the *del* bus at $wc = 1$ and $wd = 1$. Then the signals $avt = 1$, $wc = 0$ and $wd = 0$ are set.

Figure 5 shows the mode and transition diagram of the *ctrl* control unit. In accordance with this diagram, the configuration and installation algorithms of synchronization modes of

the input operand streams x and y are defined. According to the Fig. 5, the "Mode 0" is the base for control block. Therefore, transitions from it to any operation modes are possible.

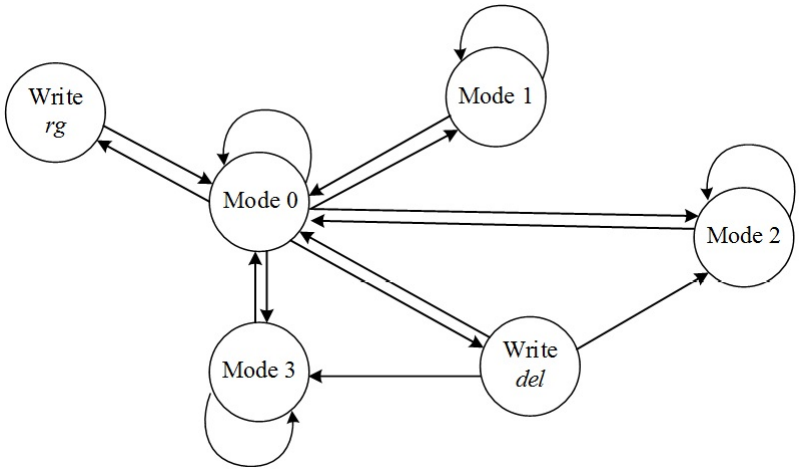

Figure 5. Diagram of transitions and the *ctrl* signals modes

Table 2 shows the basic conditions for switching the *ctrl* block to each operating mode and installation of the corresponding signals and data: *sm* mode numbers; *md* data markers; *wi*, *ws*, *wr* writting enable signals; *as* and *ai* delay depths.

Table 2. Basic conditions for transitions and the *ctrl* signals modes

The mode	Transition condition	Action
"Mode 0"	$(mx \oplus my)\&\overline{wc}\&\overline{avt}\&\overline{wd} = 1$	$sm = 0; ws = 1; md = 1$
"Write rg"	$(mx \oplus my)\&wc\&\overline{avt}\&\overline{wd} = 1$	$sm = 0; wr = 1; ws = 0; md = 0;$ rg=x, if sb=0; rg=y, if sb=1; switch to "Mode 0"
"Mode 1"	$mx\&my\&(\Delta = 0)\&\overline{ws}\&\overline{avt}\&\overline{wd} = 1$	$sm = 1; ws = 0; md = 1$
"Mode 2"	$mx\&my\&(0 < \Delta \leq 10^3)\&\overline{ws}\&\overline{avt}\&\overline{wd} = 1$ or $mx\&my\&(0 < del \leq 10^3)\&\overline{ws}\&\overline{wd} = 1$	$sm = 2; ws = 1; wi = 0; md = 1;$ as=Δ, if avt=0; as=del, if avt=1
"Mode 3"	$mx\&my\&(\Delta > 10^3)\&\overline{ws}\&\overline{avt}\&\overline{wd} = 1$ or $mx\&my\&(del > 10^3)\&\overline{ws}\&\overline{wd} = 1$	$sm = 3; wi = 1; ws = 0; md = 1;$ $ai = \Delta$, if avt=0; ai=del, if avt=1
"Write del"	$wc\&wd\&\&\overline{avt}\&\overline{mx}\&\overline{my} = 1$	write *del* in *ctrl*

The considered modes of the *ctrl* block provide the necessary OB functional set for synthesis of computational structures or data processing on synthesized structures at solving the problems.

The construction of *rg* and *syncf* elements is of particular importance for OB implementation. These elements perform memory functions at storing constants and synchronizing threads. As noted earlier, the prospects for the photonic memory creation are very indefinite. It is possible to use only the DPC trigger elements for their implementation. At the same time, DPC triggers do not have an information storage mode. Therefore, it is necessary to use a "ring buffer" scheme

to develop the delay and store register of the *rg* constant. The *rg* block diagram, implemented in the DPC basis, has the form as shown in Fig. 6.

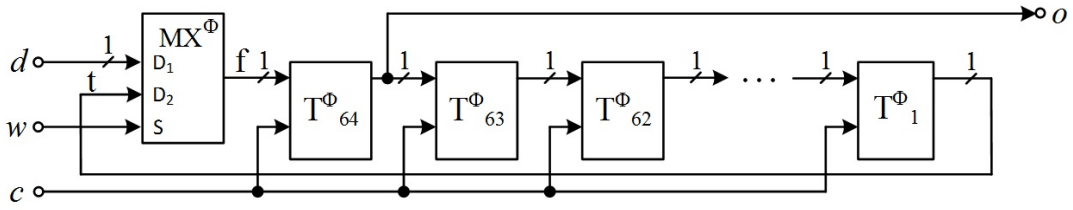

Figure 6. The *rg* block diagram

The *rg* block includes:

– T_{64}^{Φ}–T_1^{Φ} are DPC triggers;

– MX^{Φ} is the DPC multiplexer.

The time diagram of the *rg* operation is given in Fig. 7.

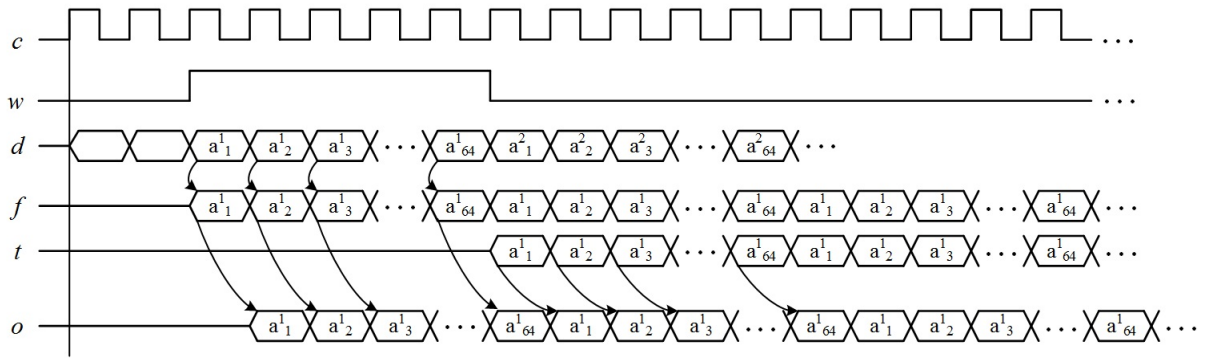

Figure 7. Time diagram of *rg* operation

When the logical "1" is applied to input w, the MX^{Φ} multiplexer output switches to bus d, and the input data is written to the T_{64}^{Φ}–T_1^{Φ} triggers. As soon as the logical "0" is received at the input w, the MX^{Φ} output will switch to the bus t, closing the ring connection. According to the time diagram, the output data $< a_1^1, a_2^1, a_3^1, \ldots, a_{64}^1 >$ appears at the output o after 1 clock cycle. If necessary, the delay can be increased without increasing hardware costs up to 64 clock cycles by switching the output bus o to the appropriate trigger.

The described approach to the construction of the n-bit data storage register assumes sequential one-bit processing. At the same time, one MX^{Φ} multiplexer and n T^{Φ} triggers are required to implement the *rg*.

Such hardware costs for the T^{Φ} resource completely coincide with the costs of $n$bit data parallel processing with any bit reduction coefficient r by digits [27]. The hardware costs of the MX^{Φ} resource are determined by the ratio $\frac{n}{r}$.

The computational structure of the *syncf* block included in the OB is shown in Fig. 8.

The *syncf* block allows the delay of operands by Δ clock cycles. It consists of N serially connected DPC triggers T_1^{Φ}–T_N^{Φ} and MX multiplexer. The delay value is set by transfer of the value Δ to the input a. The time diagram of the *rg* operation is shown in Fig. 9.

Three options of operand stream synchronization on the *syncf* block are shown in the diagram: 1 is the synchronization of operand streams with the difference $a = 3$ clock cycles; 2 is the synchronization of operand streams with the difference $a = 5$ clock cycles; 3 is the synchronization of operand streams with the difference $a = 6$ clock cycles.

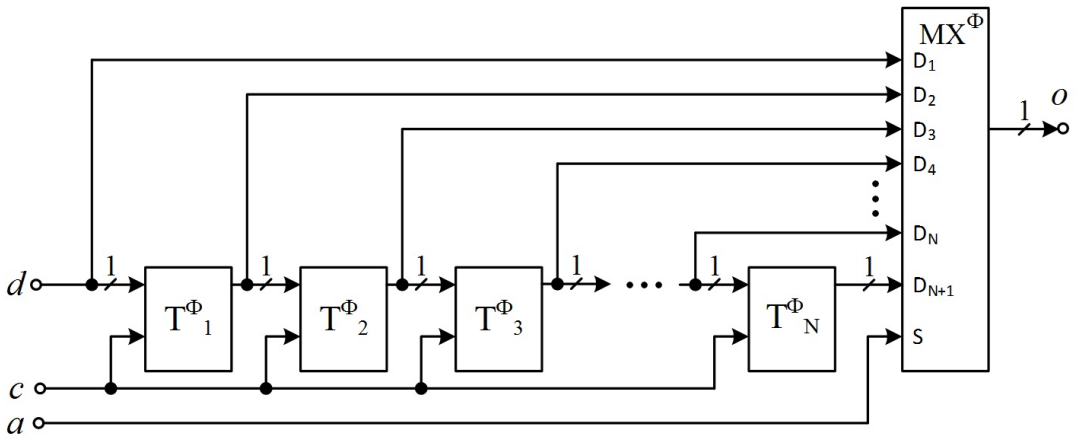

Figure 8. *Syncf* block diagram

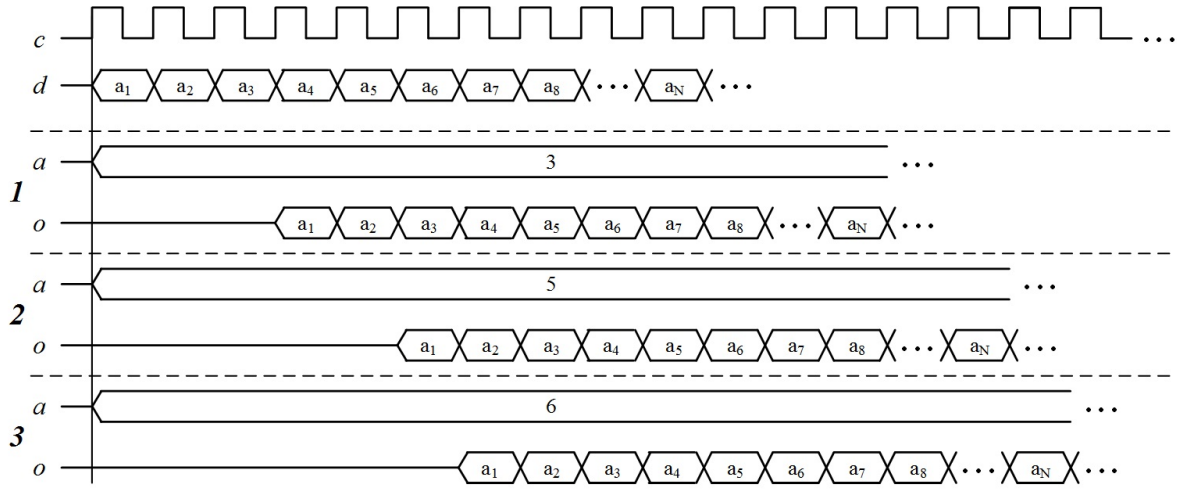

Figure 9. Time diagram of *syncf* operation

To organize data delays synchronization by Δ clock cycles during sequential one-bit processing, the Δ of T^{Φ} triggers and one MX^{Φ} multiplexer will be required. This is r times less compared to the costs at parallel data processing. The increasing of the processed data digit capacity will lead to corresponding increase in the hardware costs for *syncf* implementation.

Theoretically, the DPC data processing can be performed in a format of any digit capacity. However, taking into account the necessity to coordinate data flows between the DPC and external memory, the increase of digit capacity of data processing at solving problems on the DPC in the paradigm of structural calculations leads to the same increase in the cost of implementation of external microelectronic memory. Therefore, the DPC data processing will be present in the minimum possible single-digit format in the near future.

4. Performance Evaluation of the Proposed Solutions

The performance P of computational structure D described in the third section was evaluated according to the following formula $P = (f \cdot N)/S$, where f is the DPC operation frequency; N is the number of FD, performing arithmetic operations in the 64FP IEEE754 standard; S is the duty cycle of input data.

At $f = 1THz$, $N = 3$, $S = 64$ the theoretical performance of the computational structure will be equal to $P = 46.875$ Gflops. This corresponds to the performance of modern processors.

Theoretical researches of the DPC performance with the structural organization of calculations were performed on the DPC functional prototype made on the RCS "Tertius-2T" [28]. The computational structure is synthesized for solving five-diagonal SLAE by dimension 10^4 and the grid step of 0.01 by the Gauss-Seidel method [29] of 60 sequentially connected pipes, each of which performs one algorithm iteration. The one pipe contains five one-bit OB and five one-bit FD: four adders and one multiplier performing calculations in the 64FP standard. In addition, three one-bit static switches of the first level, one one-bit static switch of the second level and six one-bit RAM access channels are used to develop the computational structure.

Researches have shown that such computing structure on DPC, provided it operates at the 1 THz frequency, will solve the problem in about 430 μs. The computational structure of 60 pipes on one FPGA XCKU095 RCD "Tertius-2T" solves a five-diagonal matrix of about 0.15 s, and the Intel Core i5-12600K 3.4 GHz processor is about 2 s. Therefore, the DPC can provide acceleration by 340 times compared to the modern FPGAs [30], and compared to modern processors – by 4500 times, subject to an equivalent amount of hardware costs.

Conclusion

The architecture of DPC and approaches to the construction of one of its main elements – the data flow switching and synchronization subsystem are described in the paper. In the future, they will make it possible to effectively use the available DPC computing resource and maintain the performance gain over microelectronic devices due to the higher frequency of data processing.

The proposed scheme for block construction of dynamic switching and synchronization of operand flows provides basic functionality both for synthesis of computing structures on DPC and matching the rate of operand flows during solving problems of mathematical physics. Further reseaches of the proposed architecture of DPC and development of the functionality of its elements will potentially expand the class of problems efficiently solved on the DPC that require high computational accuracy.

The estimates performed by the authors show that the DPC with the data flow architecture and the structural organization of calculations at solving time-consuming problems of mathematical physics currently have the potential to provide performance exceeding by two or more decimal orders of magnitude the performance of modern computing systems.

Acknowledgements

The work was carried out within the framework of the scientific program of the National Center for Physics and Mathematics (the project "National Center for Supercomputer Research").

References

1. Chernyak, L.: Amdahl's Law and the future of multicore processors. Open systems. DBMS 04 (2009). https://www.osp.ru/os/2009/04/9288815/, accessed: 2023-05-02

2. Moore, G.: No Exponential is Forever: But "Forever" Can Be Delayed! In: International SolidState Circuits Conference (ISSCC), 2003. Session 1. Plenary 1.1. Vol. 91, no. 11, pp. 1934–1939. IEEE (2003). https://doi.org/10.1109/ISSCC.2003.1234194.

3. Benioff, P.: Quantum mechanical hamiltonian models of turing machines. Journal of Statistical Physics 29(3), 515–546 (1982). https://doi.org/10.1007/BF01342185

4. D-Wave Announces General Availability of First Quantum Computer Built for Business. https://www.dwavesys.com/company/newsroom/press-release/d-wave-announces-general-availability-of-first-quantum-computer-built-for-\business/, accessed: 2023-05-02

5. Dalzell, A.M., Harrow, A.W., Koh, D.E., Placa, R.L.L.: How many qubits are needed for quantum computational supremacy? Quantum 4, 264 (2020). https://doi.org/10.22331/q-2020-05-11-264

6. Shubin, V.V., Balashov, K.I.: Fully optical logical basis based on a micro-ring resonator. Patent No. 2677119. The Federal State Unitary Enterprise "Russian Federal Nuclear Center - All-Russian Research Institute of Experimental Physics" (FSUE RFNC-VNIIEF), Rosatom VNIEF

7. Tamer, A.: Moniem All-optical XNOR gate based on 2D photonic-crystal ring resonators. Quantum Electronics 47(2), 169 (2017). https://doi.org/10.1070/QEL16279

8. Next generation photonic memory devices are "light-written", ultrafast and energy efficient (2019). https://www.tue.nl/en/news/news-overview/10-01-2019-next-generation-photonic-memory-devices-are-light-written-ultra\fast-and-energy-efficient/, accessed: 2023-05-02

9. Using light for next-generation data storage (2018). https://phys.org/news/2018-06-next-generation-storage.html, accessed: 2023-05-02

10. Zhang, Q., Xia, Z., Cheng, Y.B., *et al.*: High-capacity optical long data memory based on enhanced Young's modulus in nanoplasmonic hybrid glass composites. Nat Commun 9, 1183 (2018). https://doi.org/10.1038/s41467-018-03589-y

11. Gordeev, A., Voitovich, V., Svyatets, G.: Promising photonic and phonon domestic technologies for terahertz microprocessors, RAM and interface with ultra-low power consumption. Modern Electronics 2(22). https://www.soel.ru/online/perspektivnye-fotonnye-i-fononnye-otechestvennye-tekhnologii-dlya-teragert\sovykh-mikroprotsessorov-o/, accessed: 2023-05-02

12. Starikov, R.S.: Optical image correlators: History and current state. In: XVI International Conference on Holography and Applied Optical Technologies, HOLOEXPO 2019. Abstracts. pp. 82–90. Bauman Moscow State Technical University, Moscow (2019)

13. Lugt, A.V.: Signal detection by complex spatial filtering. IEEE Transactions on Information Theory 10(2), 139–145 (1964). https://doi.org/10.1109/TIT.1964.1053650

14. Arsenault, H.H., Sheng, Y.: An Introduction to Optics in Computers. Volume 8 of Tutorial texts in optical engineering. SPIE Press, Washington (1992). 126 p.

15. Stone, R.V., Zeise, F.F., Guilfoylev, P.S.: DOC II 32-bit digital optical computer: optoelectronic hardware and software. In: Optical Enhancements to Computing Technology. Proceedings, vol. 1563. SPIE (1991). https://doi.org/10.1117/12.49689

16. Barhen, J., Kotas, C., Humble, T.S., *et al.*: High performance FFT on multicore processors. In: 2010 Proceedings of the Fifth International Conference on Cognitive Radio Oriented Wireless Networks and Communications, (CROWNCOM). pp. 1–6. IEEE (2010). https://doi.org/10.4108/ICST.CROWNCOM2010.9283

17. Stepanenko, S.A.: Interference logic elements. Reports of the Russian Academy of Sciences. Mathematics, Computer Science, Management Processes 493, 68–73 (2020)

18. Kuznetsova, O.V., Speransky, V.S.: Solving optical signal processing problems without optoelectronic conversion. Telecommunications and Transport. T-Comm. 8, 35–39 (2012)

19. Wu, X., Tian, J., Yang, R.: A Type of All-Optical Logic Gate Base on Graphene Surface Plasmon Polaritons. Optics Communications 403, 185–192 (2017). http://doi.org/10.1016/j.optcom.2017.07.041

20. Papaioannou, M., Plum, E., Valente, J., *et al.*: All-Optical Multichannel Logic Based on Coherent Perfect Absorption in a Plasmonic Metamaterial. APL PHOTONICS 1, 090801 (2016). https://doi.org/10.1063/1.4966269

21. Hussein, M.E., Ali, T.A., Rafab, N.H.: New Design of a Complete Set of Photonic Crystals Logic Gates. Optics Communications 411, 175–181 (2018). https://doi.org/10.1016/j.optcom.2017.11.043

22. Stepanenko, S.A.: Photonic computing machine. Principles of implementation. Parameter estimates. Reports of the Academy of Sciences 476(4), 389–394 (2017). https://doi.org/10.1134/S1064562417050234

23. Levin, I.I., Sorokin, D.A., Kasatkin, A.V.: Perspective architecture of a digital photonic computer. Izvestiya of the SFeDu. Technical sciences 6(230), 61–71 (2022). https://doi.org/10.18522/2311-3103-2022-6-61-71

24. Besedin, I.V., Dmitrenko, N.N., Kalyaev, I.A., *et al.*: A family of basic modules for building reconfigurable computing systems with a structural and procedural organization of computing. In: Scientific service on the Internet. Proceedings of the All-Russian Conference. pp. 47–49. MSU, RSU, IVT RAS (2006)

25. Kalyaev, I.A., Levin, I.I.: Reconfigurable multiconveyor computing systems for solving streaming problems. Information technologies and computing systems 2, 12–22 (2011)

26. Kalyaev, I.A., Levin, I.I., Semernikov, E.A., Shmoilov, V.I.: Reconfigurable Multipipeline Computing Structures. Nova Science Publishers, Inc., New York, USA (2012). 345 p.

27. Dordopulo, A.I., Sorokin, D.A.: Methodology for reducing hardware costs in complex systems at solving problems with significantly variable intensity of data flows. Izvestiya SFeDu. Technical sciences 4(129), 194–199 (2012)

28. Supercomputers and Neurocomputers Research Center. Tertsius-2. http://superevm.ru/index.php?page=tertsius-2, accessed: 2023-05-02

29. Shpakovsky, G.I., Verkhoturov, A.E.: Algorithm of parallel SLOUGH solution by Gauss-Seidel method. Bulletin of BSU 1(1), 44–48 (2007)

30. Kalyaev, I.A., Levin, I.I.: Reconfigurable computing systems with high real performance. In: International Scientific Conference, Parallel Computational Technologies, PaCT-2009. Proceedings. SUSU Publishing House, Chelyabinsk (2009)