MSC 68U07, 86A20 **DOI: 10.14529/mmp200106**

IMPLEMENTATION OF SURFACE-RELATED MULTIPLE PREDICTION PROBLEM ON RECONFIGURABLE COMPUTER SYSTEMS

***K.N. Alekseev*[1], *I.I. Levin*[1], *D.A. Sorokin*[2]**
[1] Southern Federal University, Taganrog, Russian Federation
[2] Supercomputers and Neurocomputers Research Center, Taganrog, Russian Federation
E-mails: alexseev91@mail.ru, iilevin@sfedu.ru, jotun@inbox.ru

The traditional methodology of computer-aided synthesis of parallel-pipeline programs for reconfigurable computer systems based on field programmable gate arrays (FPGAs) is aimed at the highest possible computer system performance, achieved on available hardware resource. Application of such an approach to real-time problems can lead to inefficient use of system hardware resource. Frequently, this fact leads to idle stand of occupied equipment and to higher requirements to power consumption, size and cost of the end product. We suggest a new methodology to synthesize of parallel-pipeline programs for solution of real-time computationally intensive problems. The methodology provides data processing having a specified rate which depends on a specified time interval. With the help of the developed methodology, it is possible to synthesize a problem computing structure, which requires the minimum hardware resource for the specified system performance. In order to illustrate the suggested methodology, we give the solution of the real-time surface-related multiple prediction problem. We evaluate various configurations of reconfigurable computer systems based on Xilinx Kintex UltraScale FPGAs.

Keywords: reconfigurable computer systems; field programmable gate array (FPGA); surface-related multiple prediction (SRMP); real-time problems.

Introduction

At present, there are several main methods that can be used for creation of parallel programs for reconfigurable computer systems (RCS). These methods are based on synthesis of a problem digital circuit described in a graphic editor or in HDLs (Hardware Description Languages). In comparison with programming of traditional computer systems, creation of configuration files for FPGAs with the help of the described methods requires considerably more time. Thereby, an approach of high-level RCS programming is actively developed. The main idea of the approach is transformation of C/C++ code to the RTL-form [1].

The existing software packages of such vendors as Xilinx, Intel, BlueSpec, Cadence Design Systems, NEC, MathWorks, etc. [2] are intended for design of IP-cores (circuit units for specified functions) on the base of high-level programming languages. However, the synthesized IP-core is integrated into the project, which is designed in an HDL language by a circuit engineer. Of course, it speedups the RCS programming process, but this time reduction is modest in comparison with the one for traditional architectures.

The high-level programming languages such as Mitrion-C [3] and OpenCL [4] allow to develop efficient parallel programs for RCS. In the case, when FPGA hardware resource is insufficient for project implementation, the user has to distribute calculations among

several chips himself, splitting one project into several ones and organizing data streams among FPGAs.

There is another approach to multichip RCS programming [5] based on automatic splitting of the project, described in the high-level programming language COLAMO [6] on the available number of FPGAs; data streams among RCS chips, basic modules, racks, etc. are synchronized automatically. The software suite for RCS parallel-pipeline programs development implements the technology of resource-independent programming. According to this technology, a parallel program can be translated into any RCS [7].

The COLAMO language allows to describe parallel algorithms according to the paradigm of structural-procedural organization of calculations [6]. In this approach, the problem is represented as an information graph [5, 7] used for synthesis of a balanced computing structure, which provides the same rate of data transfer through all functional units of the digital circuit.

Frequently the structural implementation of the problem information graph requires hardware resource, which modern RCS cannot provide. Therefore, the COLAMO language contains the minimum structures (the so-caled "cadrs"), which are the result of the problem information graph splitting according to special rules [5]. The main idea of the cadr representation of the problem is creation of structural parts of a special-purpose calculator, which can be sequentially realized on the RCS computational field [5, 6] owing to the configuration capability.

In the case, when the system hardware resource is insufficient for structural implementation of even one cadr, a methodology of hardware costs reduction is applied [8]. It was shown that hardware costs for implementation of the problem information graph can be reduced by several-fold decreasing of the computing structure performance such that the structure remains balanced. Such approach provides achievement of high performance of the computing structure owing to the maximum possible use of available RCS hardware resource.

However, there is a class of problems, which are to be solved during a fixed time interval; for example, computationally intensive real-time problems. If such problems are solved on RCS, it is necessary to provide a fixed performance which depends on the specified time interval.

Frequently, such problems are to be solved on board of vehicles and on remoted industrial locations. In such cases, when designing computer facilities, it is necessary to take into account such technical parameters as power consumption, resistance to mechanical, vibration and other stresses, reliability and fault-tolerance, products size, etc. According to the given requirements, when solving such problems on RCS, it is necessary to define the minimum hardware resource of the system, which is suitable for data processing during the specified time.

The computing structure for the real-time problem synthesized by traditional methods processes data with a certain rate, which does not depend on the specified problem solution time. If the data processing rate is less than the required one, then the problem cannot be solved on the available RCS hardware resource during the specified time. If the data processing rate is higher than the required one, then the RCS equipment will stand idle. Therefore, traditional methods do not allow synthesis of efficient parallel programs for real-time problems.

1. A Methodology of Parallel Programs Synthesis for Real-Time Problems Solution on Reconfigure Computer Systems

In order to achieve higher efficiency of synthesized solutions, we suggest a new methodology for solution of computationally intensive real-time problems on RCS. The main idea of the methodology is synthesis of a balanced computing structure of the problem, which occupies the minimum part of available RCS hardware resource. The new methodology describes special transformations of the problem information graph and provides synthesis of computing structures with a certain performance based on the specified problem time.

With the help of such approach, it is possible to define several the most efficient variants of RCS components. Since all existing FPGA families have qualitatively different characteristics (the number of LUTs, DSPs, blocks RAM, transceivers and other types of embedded blocks), it is possible to detect the most efficient variants only within one FPGA family due to the new methodology. Selection of variants for different families is performed with the help of re-analysis according to other characteristics of the components, such as the frequency and the amount of hardware resource.

The most suitable variant of RCS configuration for solution of a certain problem is selected by the user according requirements to the cost, the power consumption, the size and other characteristics of the end product.

According to all the above, let us present the main algorithm steps and points of the methodology of parallel programs synthesis for computationally intensive real-time problems, solved on RCS.

1) Specify the components of the RCS, which is used for implementation of the problem, and specify such parameter of the components as the planned computer system working frequency v. This parameter is specified on the base of the capabilities of the selected FPGA family.

2) Analyse the problem algorithm, specify its parameters, create the problem information graph, exclude computationally redundant fragments of the graph and exclude fragments of constants calculation. Since several subproblems can use different algorithms providing the same result, it is necessary to choose an algorithm which requires the minimum hardware resource.

3) Specify the time interval T_Z for the problem solution, and calculate the theoretical problem time as $T = \tau/v$, where τ is the latency time of the structure. Compare these time intervals and define the further actions: if $T < T_Z$, then the problem computing structure performance reduction is required; if $T = T_Z$, then reduction is not performed, and we calculate hardware costs straightway; if $T > T_Z$, then the problem cannot be solved on the RCS with the selected components during the specified time T_Z. In order to provide the problem solution, it is necessary to specify the problem time T_Z and/or the planned computing structure frequency v again.

4) If $T < T_Z$, then specify the initial reduction coefficient, and perform the problem computing system performance reduction with the help of existing methods: reduction by number of basic subgraphs; reduction by number of operations; reduction by data capacity; reduction by data ratio and by frequency. In contrast to the existing synthesis methods of RCS parallel-pipeline programs, the initial reduction coefficient for real-time problems depends on the specified problem time and the input data flow rate. Besides, the operation of performance reduction is performed once for all subproblemsof the information graph.

5) Calculate hardware costs for implementation of the problem computing structure taking into account all possible parameters such as the number of input/output channels, the number of embedded transceivers, the number of triggers and LUTs, the number of *DSP* arithmetic units and variants of their use in arithmetic operations, the number of embedded FPGA memory blocks.

6) Depending on the characteristics of the selected FPGA family, find the most efficient variants of implementation for the problem computing structure, proportionally using RCS hardware resource for the considered FPGA series.

7) Synthesize the problem computing structure, to organize data flows and develop a parallel-pipeline program.

2. Description of the Surface-Related Multiple Prediction Problem

In geological-geophysical research dealing with engineering and monitoring of hydrocarbon crude reservoirs, the geophysical information is received as a result of seismic prospecting and used instead of direct observations. Owing to its qualitative interpretation in real time, it is possible to respond efficiently to variations of oil-and-gas reservoirs; as a result it is possible to increase the production level [9]. However, receiving the geophysical information is accompanied with interferences, which complicate further analysis or make it completely impossible. In particular, this is due to the registration of surface-related multiples.

One of widely used methods of surface-related multiple elimination is the SRME two-step method (Surface Related Multiple Elimination) suggested by A.J. Berkhout and D.J. Verschuur [10, 11]. The first step of the method is the multiple prediction by SRMP (Surface Related Multiple Prediction) algorithm, and the second step is to eliminate multiples from the initial data. Surface-related multiples are calculated by summation of convolutions results over all possible source location [12, 13].

Fig. 1 [14] shows generation and recording of multiples that cause interference in the seismogram. The acoustic signal from the source located in the point a propagates along subsurface, has the downward reflection in the point z initiated at the water surface, and then reflects and travels upward to the surface [12]. Then, the wave is recorded by the seismic sensor in the point b [13].

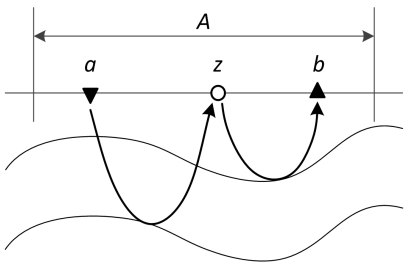

Fig. 1. Process of generation and recording of multiple waves

A multiple wave can be described by the coordinates (x_a, x_z, x_b), where x_a is a coordinate of a signal source located in the point a, x_z is a coordinate of a signal reflection point, x_b is a coordinate of a signal receiver placed in the point b. Such a wave consists of two segments recorded in the initial data: the wave in the source (x_a, x_z) seismic record, and the wave in the receiver (x_z, x_b) seismic record [13].

In order to calculate the multiple waves field, it is necessary to "combine" their traces. The SRMP algorithm performs this operation by convolution of traces, where multiple waves were recorded. Due to the fact that the point x_z, where the ray appears on the surface, is initially unknown, all possible coordinates within a certain interval A are taken, and the results of their convolutions are summed up [13]. The initial data is a sequence of seismic traces represented as a 3D array (see Fig. 2).

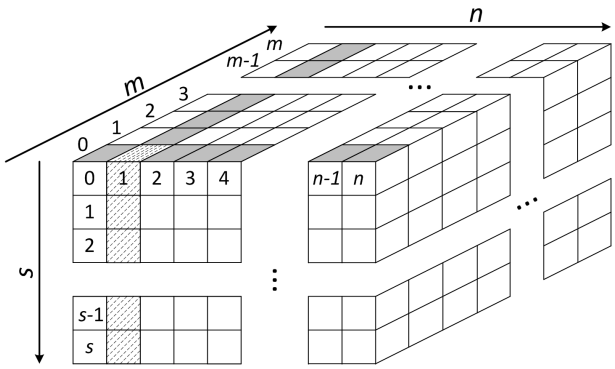

Fig. 2. The structure of a data array for the SRMP problem

The traces, where waves are recorded, are usually [13,15] described as (x_i, x_j, t), where x_i is the signal source coordinate, x_j is the signal receiver coordinate, t is the set of time samples. The traces are united into layers, sorted by sources and receivers in such a way that the layer of the source i is a combination of m traces, and the layer of the receiver j is a combination of n traces. In Fig. 2, the trace recorded by the source coordinate $n = 1$ and by the receiver coordinate $m = 0$ is marked by the dash-dotted line. The layers of the source coordinate $n = 1$ and receiver coordinate $m = 0$ are highlighted in grey color.

According to the SRMP algorithm, the model of multiple waves for one trace $M(x_i, x_j, t)$ can be represented as follows [15]:

$$M(x_i, x_j, t) = f(t) \cdot \sum_{x_z} S_i(x_i, x_z, t) * R_j(x_z, x_j, t), \tag{1}$$

where $S(x_i, x_z, t)$ is the trace recorded for a signal source placed in the point i and a signal receiver placed in the point z; $R(x_z, x_j, t)$ is the trace recorded for a signal source placed in the point z and a signal receiver placed in the point j; $f(t)$ is the matched filter; (*) is the time convolution procedure of traces spectra.

If the problem is computationally complex, memory access is non-linear and processing data is huge, then use of cluster supercomputers is not suitable for implementation of the problem. Due to features of fixed architectures, it is impossible to solve the problem and to provide the required accuracy in the real-time mode in the very moment when geological information is being gathered [14]. Increasing the number of computational nodes, involved in implementation of the problem, we aggravate problems of high-speed interprocessor data exchange, which block the problem solution in the real-time mode [14]. RCS have no architectural limitations of such a kind. For tightly-coupled problems, the RCS performance will grow proportionally to available hardware resource. It will provide linear scalability of the problem computational fragments without increasing time costs [7].

To provide real-time solution of the surface-related multiple prediction problem, let us apply the developed methodology of RCS parallel-pipeline programs creation. Then, let us compare the obtained results with results of traditional RCS programming methods.

3. Synthesis of the Computing Structure for the Surface-Related Multiple Prediction Problem

According to organization of computational process, the SRMP algorithm presumes sequential execution of the following operations: the direct FFT (fast Fourier

transformation), convolution of traces spectra for the source level and for the receiver level, and the reverse FFT. The initial data is single-precision represented according to the IEEE 754 standard [14].

According to the developed synthesis methodology of parallel-pipeline programs for real-time problems, we have to specify, at the preliminary step, RCS components planned for the problem solution. Since the surface-related multiple prediction problem is a computationally intensive one, then, in this case, we have to consider reasonably such high-performance FPGA families as Xilinx Kintex UltraScale [16]. These components are capable to provide the working frequency of computational devices $v_1 = 500MHz$ [17].

Let us define the main parameters of the surface-related multiple prediction problem. To be specific, suppose that after preliminary processing of seismic data the number of sources and receivers is $n = m = 8192$, and the number of time samples is $s = 4096$. In this case, the input data size is equal to 1 TB, and it is close to the extreme values obtained in practice [14].

On the base of the above-mentioned parameters, we form an information graph for the surface-related multiple prediction problem. The information graph G_{SRMP} given in Fig. 3 consists of two direct FFT subproblems P_1 and P_2, a subproblem of traces spectra convolution P_3, and a reverse FFT subproblem P_4.

$$G_{SRMP} = \bigcup_{k=1}^{4} P_k; \quad P_k = \bigcup_{i=1}^{n}\bigcup_{l=1}^{n}\bigcup_{j=1}^{m} g_{ilj}^{k}, \quad k \in \overline{1,2};$$

$$P_3 = \bigcup_{i=1}^{n}\bigcup_{j=1}^{m} G_{ij}^{3}; \quad G_{ij}^{3} = \left(\bigcup_{l=1}^{n} g_{ijl}^{31}\right) \bigcup \left(\bigcup_{t=1}^{s} g_{ijt}^{32}\right); \quad P_4 = \bigcup_{i=1}^{n}\bigcup_{j=1}^{m} g_{ij}^{4},$$

where g_{ilj}^{1} and g_{ilj}^{2} are the basic subgraphs of the direct FFT subproblems; G_{ij}^{3} are the subgraphs of the traces spectra convolution; g_{ijl}^{31} are the basic subgraphs of the operation of complex multiplication; g_{ijt}^{32} are the basic subgraphs of the operation of complex addition; g_{ij}^{4} are the basic subgraphs of reverse FFT subproblems. The structure of the basic subgraphs of direct and reverse FFT consists of a set of FFT basic operations *BO FFT* [18], shown in Fig. 3 as an empty circle. In each basic subgraph the number of basic operations is $N_{BOFFT} = (s/2) \cdot \log_2 s$.

The input data $\{S_1^1 \div S_n^s\}$ and $\{R_1^1 \div R_m^s\}$ are concurrently supplied to the subgraphs G_1 and G_2 according to the execution rules of the convolution operation in the subgraph G^3, where $\{\ \}$ is a data array with parallel access to elements named vector [5]. As a result, after processing $n \cdot m$ vectors of input data, the multiple waves model $\{M_{11}^1 \div M_{nm}^s\}$ is formed.

The analysis of the information graph G_{SRMP} shows that it is reasonable to replace the operation *BO FFT* by the modified *FFT (MBO FFT)*, which in advance performs calculation of the coefficients W and stores them in the FPGA ROM. Such implementation requires less hardware resource for arithmetic operations and data channels, and slightly more memory.

Due to the fact that the initial data is real numbers, it is reasonable to use the algorithm of irredundant FFT given in [18,19] and, as a result, to process two real sequences concurrently. Let us combine two subproblems P_1 and P_2 into one subproblem P_{12}, and add a subproblem *SPLIT* for data flow splitting. The transformed information graph G_{SRMP} is given in Fig. 4.

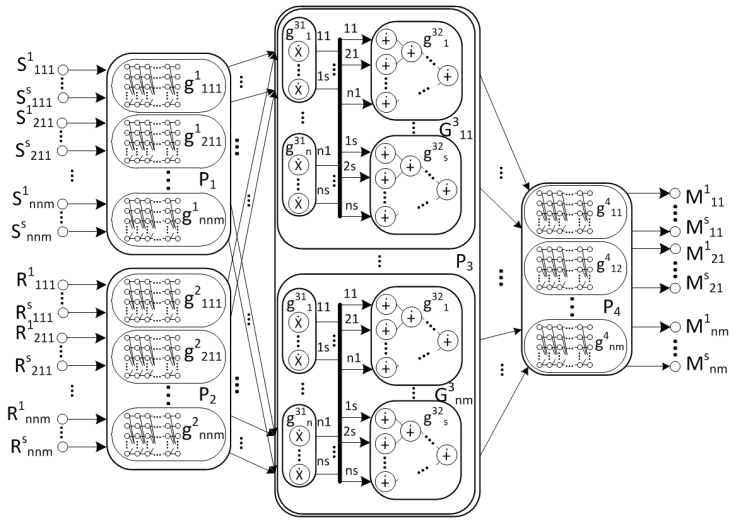

Fig. 3. The information graph G_{SRMP}

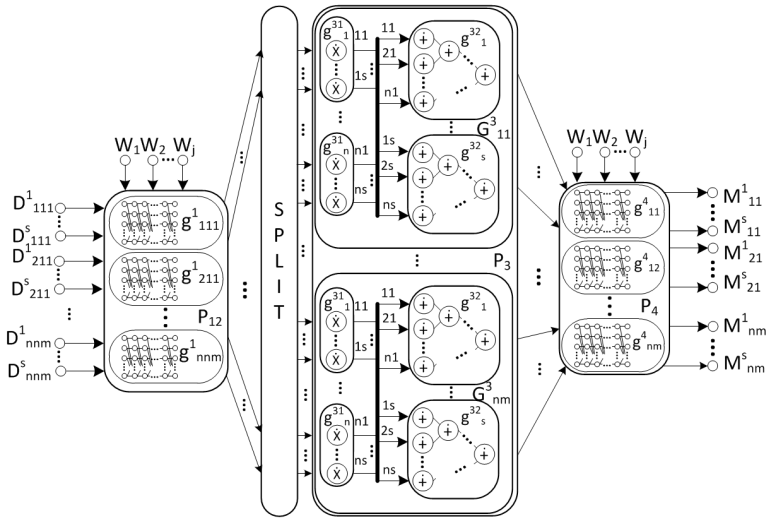

Fig. 4. The transformed information graph G_{SRMP}

After creation and analysis of the problem information graph at the preliminary step of the developed methodology, it is necessary to specify the time T_Z, to calculate the execution time T of the computing structure, synthesized on the base of the problem information graph, and to compare the obtained results. As an example, let us consider the case, when the SRMP problem is to be solved during the time T_Z=1,5 hour. It is possible to calculate the execution time of the computing structure according to the formula $T = \tau/v$. Since the latency of the computing structure synthesized on the base of the information graph G_{SRMP} is τ=1335, then $T < T_Z$. In this case, according to the developed methodology, the operation of performance reduction is required for the computing structure.

The mathematical tool of the computer system performance reduction is described in details in [8]. However, in contrast to the methodology of multi-criteria reduction, real-time problems require this operation to be performed once for all subproblems of the information graph. The reduction coefficient is obtained as [8]

$$r^0 = [T_Z \cdot v + \tau], \tag{2}$$

where [ ] is the rounding downward operation. In our case, the initial reduction coefficient for the FPGA Kintex UltraScale family is $r^0 = 2,7 \cdot 10^{12}$.

After the operation of performance reduction with the coefficient r^0, and primary evaluation of the required hardware resource, we conclude that the critical resource is the number of data channels. Therefore, we have developed a computational pipeline, which calculates one trace of multiple waves, and the required performance of the computing structure was achieved with the help of the developed macropipeline circuit. In contrast to structural implementation, all p of computational pipelines do not have data exchange among each other. Owing to this, it is possible to reduce the number of required data channels between FPGAs. The computing structure $G^R_{SRMP} = P^R_{12} \bigcup P^R_3 \bigcup P^R_4$ is shown in Fig. 5.

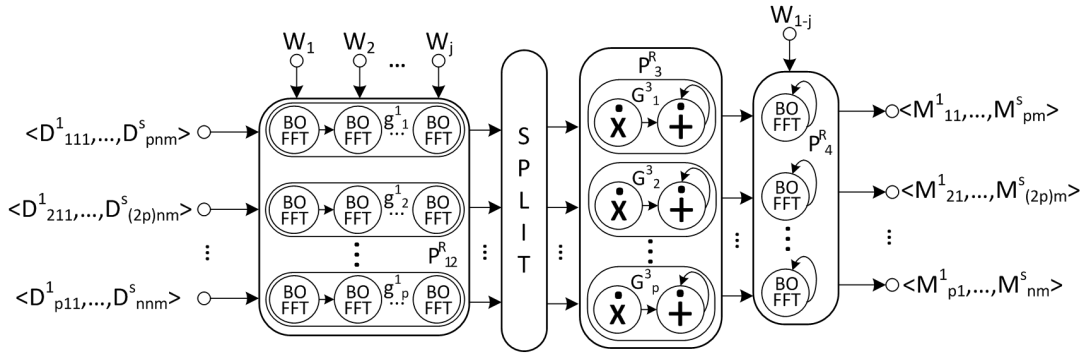

Fig. 5. The computing structure G^R_{SRMP} after the operation of performance reduction

The input data for the reduced information graph G^R_{SRMP} of the SRMP problem (see Fig. 5) are represented as $[D^1_{111} \div D^s_{nnm}]$, where $[\cdot]$ is a data array with parallel-sequential access to elements [5]. The coefficient p is the number of data channels, which is calculated according to the number of operations in the first layer of the FFT graph of the subproblem P^R_{12}. In our case, p=417.

The result of processing of all sequences of input data arrays by the computing structure G^R_{SRMP} is the model of surface-related multiples $[M^1_{11} \div M^s_{nm}]$. Therefore, the problem information graph is transformed into the form, which provides data processing during the specified time.

4. Evaluation of the Minimum Hardware Resource for the Specified Solution Time of the Surface-Related Multiple Prediction Problem

RCS hardware resource for implementation of a problem is to be evaluated according to the following parameters: the number of input/output channels, the number of embedded transceivers, the number of used *FF* triggers, the number of *LUTs*, the number of embedded *DSP* arithmetic blocks, and the number of embedded memory blocks.

The computing structure with the reduced performance contains N_{mul}=25 020 multipliers and N_{add}=35 862 adders. RCS hardware resource depends on implementation of arithmetic operations with or without use of embedded *DSP*-units. Table 1 contains

hardware resource used for implementation of arithmetic operations on RCS and given for different variants [17].

Table 1

Description of hardware resource for implementation of arithmetic operations on RCS

Required hardware resource	Multiplication		Addition	
	no *DSP*	with *DSP*	no *DSP*	with *DSP*
FlipFlops, *FF*	672	166	562	309
Look Up Tables, *LUT*	572	92	353	206
Arifmetic blocks, *DSP*	–	2	–	2

In order to synthesize a computing structure that occupies the minimum hardware resource, it is necessary to choose a proportion of components, which provides even filling of FPGAs according to the considered parameters. Since proportions of various FPGA hardware resource parameters are frequently constants within a family, it is possible to detect the optimal proportion of used *DSP* arithmetic blocks to the number of used *LUT* tables. Owing to this proportion, we can calculate the part of arithmetic operations using embedded *DSP* blocks for the designed computing structure.

For all FPGAs of the Kintex UltraScale family except the KU095 FPGA, the proportion of the number of *LUTs* to the number of *DSP* blocks is $K_{DSP}^{LUT} = N_{LUT}/N_{DSP} \approx 120$. According to the total number of operations and variants of their implementation, all operations of multiplication and 58,4% of addition are to be realized on the base of *DSP* blocks. In this case, N_{FF}^{op}=19 008 932, N_{LUT}^{op}=11 882 358, and N_{DSP}^{op}=91 928.

For the KU095 FPGA, the proportion of the number of *LUTs* to the number of *DSP* blocks is $K_{DSP}^{LUT} \approx 700$. In this case, all operations of addition are to be realized without *DSP* blocks, and 57,3% of operations of multiplication are to be realized with *DSP* blocks. Therefore, N_{FF}^{op}=29 793 816, N_{LUT}^{op}=20 165 286, and N_{DSP}^{op}=28 356.

Let us represent calculations of hardware costs for the surface-related multiple prediction problem for several variants of the RCS components. The RCS is designed on the base of the Kintex UltraScale family. Table 2 contains the required number of FPGAs, which are suitable for realization of p computational pipelines of the surface-related multiple prediction problem.

Table 2

Variants of RCS configuration according to used components

Resource RCS	FPGAs of the Xilinx Kintex UltraScale family						
	KU025	KU035	KU040	KU060	KU085	KU095	KU115
Pipes, p	5	7	8	11	17	11	19
N_{FPGA}	84	60	53	38	25	38	19

We suggest a structure of data flows in the FPGA (see Fig. 6). The external storing device (ES) stores the whole input data array. The RAM blocks ($RAM_1 \div RAM_p$) are loaded with the trace layers sorted by receivers. Calculations are organized to provide concurrent data transfer (layers sorted by the source i) from the ES into the all pipelines $Pipe_1 \div Pipe_p$; p layers sorted by the receivers are concurrently read from the RAMs. In this case, at the output, we obtain p traces of multiples $M_{i,p}$, then the source layer $i+1$ is transferred from the ES. After convolution of all n source coordinate layers with the layers

from p receivers coordinate, we obtain p layers of multiples. When all trace layers loaded into the RAM are processed, it is necessary to load new layers and continue processing.

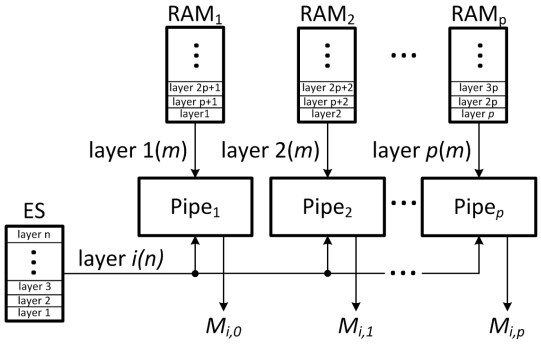

Fig. 6. A structure of data flows for the RCS solution of the surface-related multiple prediction problem

In order to specify the required number of input/output channels and the number of embedded transceivers in the FPGA, let us calculate the data supply rate $V_{G_{SRMP}}^{FPGA} = v \cdot q \cdot (S_{in} + S_{out}) \cdot c$, where $S_{in} = p + 1$ is the number of input data channels with the capacity q, $S_{out} = 2p$ is the number of output data channels. Since the output data flow rate is less than the input one by ten folds, we take into account this difference using the correcting coefficient $c = 0,4992$. For the considered variant of implementation, the capacity q is equal to 64.

The family contains embedded transceivers with the data transfer rate V_{tr}=16,3 Gb/s. The number of transceivers, required for implementation of the surface-related multiple prediction problem is calculated as follows: $N_{tr} =]V_{G_{SRMP}}^{FPGA}/V_{tr}[$, where] [rounding upward. The obtained data is given in Table 3.

Table 3

The number of transceivers required for implementation of the surface-related multiple prediction problem

Resource RCS	FPGAs of the Xilinx Kintex UltraScale family						
	KU025	KU035	KU040	KU060	KU085	KU095	KU115
Transceivers, N_{tr}	16	29	25	34	51	34	57

In our case, to provide the required data transfer rate, it is necessary to increase N_{FPGA} for KU025, KU035, KU040 and KU060 FPGAs. The number of embedded transceivers in all other reviewed FPGAs of the Xilinx Kintex UltraScale family completely fulfills data transfer rate requirements. The final number of FPGAs required for the reviewed set of RCS components is given in Table 4.

Table 4

The number of FPGAs required for various RCS configurations

Resource RCS	FPGAs of the Xilinx Kintex UltraScale family						
	KU025	KU035	KU040	KU060	KU085	KU095	KU115
N_{FPGA}	112	109	67	41	25	38	19

The performance of the synthesized computing structure is $P_{SRMP} = N_{op} \cdot v$; where $N_{op} = N_{mul} + N_{add}$ is the total number of IEEE 754 single precision arithmetic operations. In our case, P_{SRMP}=30,441 TFLOPS.

Traditional methods of RCS parallel-pipeline programs synthesis [5, 7, 8] are aimed at creation of a parallel-pipeline program for an existing RCS. Let us compare the obtained results with a computational block "Nekkar", which consists of 12 basic modules with 96 placed KU095 FPGAs [16]. This RCS is designed to solve computationally complex problems such as surface-related multiple prediction problem. In this case, it is impossible to compare results of traditional methods for all reviewed FPGAs of the UltraScale family.

The synthesized computing structure of the surface-related multiple prediction problem occupies the whole available RCS hardware resource. Evaluation shows that it is possible to place the problem computing structure on the considered RCS after performance reduction with the coefficient $r = 1,65 \cdot 10^{12}$. In this case, the total number of computational pipelines is p=682, and P_{SRMP}=49,786 TFLOPS. Since, in this case, the computing structure performance exceeds the desired value more than 1,5 times, the occupied hardware stays idle from time to time. Therefore, the traditional method is not efficient in this case.

Conclusion

We create a new methodology of parallel applications development for real-time problems implementation on RCS. Owing to this methodology, it is possible to reduce the performance of the synthesized problem computing structure to the value, which provides results during the specified time interval. Using the methodology, we can detect the minimum RCS hardware resource for the specified rate of data flows processing. We suggest a method which forms variants of RCS configuration according to FPGA family and parameters.

The methodology is applied to the surface-related multiple prediction problem. We design the computing structure which provides real-time dataflow processing.

We compare the results of the developed methodology with traditional synthesis of parallel-pipeline programs. Since traditional methods ensure the problem implementation only on existing RCS, we compare our results with the computational block "Nekkar", which main computational elements are 96 high-performance Xilinx Kintex UltraScale KU095 FPGAs. Owing to use of the developed methodology, it is possible to reduce in more than 1,5 time the required hardware resource of the surface-related multiple prediction problem.

References

1. Chu P.P. *RTL Hardware Design Using VHDL: Coding for Eflciency, Portability, and Scalability*. N.Y., John Wiley and Sons, 2006.

2. Nane R., Sima V.M., Pilato C., Choi J., Fort B., Canis A., Chen Y.T., Hsiao H., Brown S. A Survey and Evaluation of FPGA High-Level Synthesis Tools. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2016, vol. 35, no. 10, pp. 1591–1604.

3. *Mitrion-c*. Available at: http://mitc-openbio.sourceforge.net/ (accessed 20 December 2018).

4. *OpenCL Overview Development.* Available at: https://www.khronos.org/opencl/ (accessed: 20 December 2018).

5. Kalyaev A.V., Levin I.I. *Modul'no-narashhivaemye mnogoprocessornye sistemy so strukturno-procedurnoy organizaciey vychisleniy* [Modular-Stackable Multiprocessor Systems with Structural-procedural Organization of Calculations]. Moscow, Janus-K, 2003. (in Russian)

6. Levin I.I., Dordopulo A.I., Gudkov V.A. *Programmirovanie rekonfiguriruemyh vychislitel'nyh uzlov na yazyke COLAMO* [Programming Reconfigurable Compute Nodes in COLAMO]. Rostov-on-Don, SFEDU, 2016. (in Russian)

7. Kalyaev I.A., Levin I.I., Semernikov E.A., Shmojlov V.I. *Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury* [Reconfigurable Multi-pipeline Computing Structures]. Rostov-on-Don, UNC RAN, 2008. (in Russian)

8. Dordopulo A.I., Sorokin D.A. A Methodology of Hardware Overhead Decrease in Complex Systems While Solving Problems With Considerably Variable Data Flow Density. *Izvestiya SFedU. Engineering Sciences*, 2012, no. 4, pp. 213–219. (in Russian)

9. Hmelevskoj V.K. *Geofizicheskie metody issledovaniya zemnoy kory* [Geophysical Methods of Studying the Earth's Crust. Book 1]. Dubna, Dubna State University, 1997. (in Russian)

10. Verschuur D.J., Berkhout A.J., Wapenaar C.P. Adaptive Surface-Related Multiple Elimination. *Geophysics*, 1992, vol. 9, pp. 1166–1177.

11. Berkhout, A.J., Verschuur D.J. Estimation of Multiple Scattering by Iterative Inversion, Part I: Theoretical Considerations. *Geophysics*, 1997, vol. 5, pp. 1586–1595.

12. Huang Xin-Wu, Sun Chun-Yan, Niu Bin-Hua, Wang Huan-Di, Zeng Min-Shan. Surface-Related Multiple Prediction and Suppression Based on Data-Consistence: a Theoretical Study and Test. *Chinese Journal of Geophysics*, 2005, vol. 1, pp. 188–196.

13. Denisov M.S., Finikov D.B. [Multiple Wave Suppression Methods in Seismic Exploration. Part 1]. *Seismic Technologies*, 2007, no. 1, pp. 5–16. (in Russian)

14. Kurin E.A., Denisov M.S. [Application of High Performance Computing Systems System in the Suppression of Multiple Reflected Wave Interference]. *Seismic Technologies*, 2011, vol. 4, pp. 35–40. (in Russian)

15. Verschuur E., Dragoset B., Moore I., Bisley R. A Perspective on 3D Surface-Related Multiple Elimination. *Geophysics*, 2010, vol. 5, pp. 245–261.

16. *UltraScale Architecture and Product Data Sheet: Overview.* Available at: https://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf (accessed December 5, 2018).

17. *Performance and Resource Utilization for Floating-point v7.1.* Available at: https://www.xilinx.com/support/documentation/ip_documentation/ru/floating-point.html (accessed May 6, 2019).

18. Rabiner L., Gold B. *Theory and Application of Digital Signal Processing.* Delhi, PHI Learning, 1975.

19. Semernikov E.A., Doronchenko Yu.I. [Conveyor Macroprocessor Digital Signal Processing with Structural and Procedural Organization of Computing]. *Herald of Computer and Information Technologies*, 2005, no. 8, pp. 49–55. (in Russian)

Received July 11, 2019

УДК 004.382.2 **DOI: 10.14529/mmp200106**

РЕАЛИЗАЦИЯ ЗАДАЧИ ПРОГНОЗИРОВАНИЯ КРАТНЫХ ВОЛН РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

***К.Н. Алексеев*[1], *И.И. Левин*[1], *Д.А. Сорокин*[2]**
[1]Южный федеральный университет, г. Таганрог, Российская Федерация
[2]ООО «Научно-исследовательский центр супер-ЭВМ и нейрокомпьютеров»,
г. Таганрог, Российская Федерация

Традиционная методика автоматизированного синтеза параллельно-конвейерных программ для реконфигурируемых вычислительных систем, основным вычислительнымсхемы, нацелена на достижение максимально возможной производительности вычислительной системы на доступном аппаратном ресурсе. Применение такого подхода при решении задач реального времени может приводить к неэффективному использованию аппаратного ресурса систем. Зачастую это приводит как к простою используемого оборудования, так и к повышенным требованиям к энергопотреблению, габаритам и стоимости конечного изделия. Предложена новая методика синтеза параллельно-конвейерных программ для решения вычислительно-трудоемких задач реального времени, позволяющая вести обработку данных с конкретной скоростью, зависящей от заданных временных рамок. С помощью разработанной методики выполняется синтез вычислительной структуры задачи, требующей минимум аппаратного ресурса для требуемой производительности системы. В качестве иллюстрации работы предлагаемой методики приведено решение задачи прогнозирования кратных волн в режиме реального времени. Были приведены оценки различных конфигураций реконфигурируемых вычислительных систем, основным вычислительным элементом которых являются программируемые логические интегральные схемы фирмы Xilinx семейства Kintex UltraScale.

Ключевые слова: реконфигурируемые вычислительные системы; программируемые логические интегральные схемы (ПЛИС); поверхностное множественное предсказание; задачи реального времени.

Литература

1. Chu, P.P. RTL Hardware Design Using VHDL: Coding for Efficiency, Portability, and Scalability / P.P. Chu. – New York: John Wiley and Sons, 2006.
2. Nane, R. A Survey and Evaluation of FPGA High-Level Synthesis Tools / R. Nane, V.M. Sima, C. Pilato, J. Choi, B. Fort, A. Canis, Y.T. Chen, H. Hsiao, S. Brown // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – 2016. – V. 35, № 10. – P. 1591–1604.
3. Mitrion-c. – URL: http://mitc-openbio.sourceforge.net/ (дата обращения: 20.12.2018).
4. OpenCL Overview Development. – URL: https://www.khronos.org/opencl/ (дата обращения: 20.12.2018).
5. Каляев, А.В. Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений / А.В. Каляев, И.И. Левин. – М.: Янус-К, 2003.
6. Левин, И.И. Программирование реконфигурируемых вычислительных узлов на языке colamo / И.И. Левин, А.И. Дордопуло, В.А. Гудков. – Ростов-на-Дону: ЮФУ, 2016.
7. Каляев, И.А. Реконфигурируемые мультиконвейерные вычислительные структуры / И.А. Каляев, И.И. Левин, Е.А. Семерников, В.И. Шмойлов. – Ростов-на-Дону: ЮНЦ РАН, 2008.

8. Дордопуло, А.И. Методика сокращения аппаратных затрат в сложных системах при решении задач с существенно-переменной интенсивностью потоков данных / А.И. Дордопуло, Д.А. Сорокин // Известия ЮФУ. – 2012. – № 4. – С. 213–219.
9. Хмелевской, В.К. Геофизические методы исследования земной коры / В.К. Хмелевской. – Дубна: Международный университет природы, общества и человека, 1997.
10. Verschuur, D.J. Adaptive Surface-Related Multiple Elimination / D.J. Verschuur, A.J. Berkhout, C.P. Wapenaar // Geophysics. – 1992. – V. 9. – P. 1166–1177.
11. Berkhout, A.J. Estimation of Multiple Scattering by Iterative Inversion, Part I: Theoretical Considerations / A.J. Berkhout, D.J. Verschuur // Geophysics. – 1997. – V. 5. – P. 1586–1595.
12. Huang Xin-Wu. Surface-Related Multiple Prediction and Suppression Based on Data-Consistence: a Theoretical Study and Test / Huang Xin-Wu, Sun Chun-Yan, Niu Bin-Hua, Wang Huan-Di, Zeng Min-Shan.// Chinese Journal of Geophysics. – 2005. – V. 1. – P. 188–196.
13. Денисов, М.С. Методы подавления кратных волн в сейсморазведке. Часть 1 / М.С. Денисов, Д.Б. Фиников // Технологии сейсморазведки. – 2007. – № 1. – С. 5–16.
14. Курин, Е.А. Применение высокопроизводительных вычислительных систем в задаче подавления многократно отраженных волн-помех / Е.А. Курин, М.С. Денисов // Технологии сейсморазведки. – 2011. – № 4. – С. 35–40.
15. Verschuur, E. A Perspective on 3D Surface-Related Multiple Elimination / E. Verschuur, B. Dragoset, I. Moore, R. Bisley // Geophysics. – 2010. – V. 5. – P. 245–261.
16. UltraScale Architecture and Product Data Sheet: Overview. – URL: https://www.xilinx.com/support/documentation/data_sheets/ds890-ultrascale-overview.pdf (дата обращения: 05.12.2018).
17. Performance and Resource Utilization for Floating-point v 7.1. – URL: https://www.xilinx.com/support/documentation/ip_documentation/ru/floating-point.html (дата обращения: 06.05.2019).
18. Рабинер, Л. Теория и применение цифровой обработки сигналов / Л. Рабинер, Б. Голд. – М.: Мир, 1978.
19. Семерников, Е.А. Конвейерный макропроцессор цифровой обработки сигналов со структурно-процедурной организацией вычислений / Е.А. Семерников, Ю.И. Доронченко // Вестник компьютерных и информационных технологий. – 2005. – №8. – С. 49–55.

Кирилл Николаевич Алексеев, аспирант, кафедра интеллектуальных многопроцессорных систем, институт компьютерных технологий и информационной безопасности инженерно-технологической академии, Южный федеральный университет (г. Таганрог, Российская Федерация), alexseev91@mail.ru.

Илья Израилевич Левин, доктор технических наук, профессор, кафедра интеллектуальных многопроцессорных систем, институт компьютерных технологий и информационной безопасности инженерно-технологической академии, Южный федеральный университет (г. Таганрог, Российская Федерация), iilevin@sfedu.ru.

Дмитрий Анатольевич Сорокин, кандидат технических наук, ООО «Научно-исследовательский центр супер-ЭВМ и нейрокомпьютеров» (г. Таганрог, Российская Федерация), jotun@inbox.ru.

Поступила в редакцию 11 июля 2019 г.