Досліджується процес управління обчисленнями в реконфігуровних обчислювальних системах. Запропонований метод забезпечення часових вимог якості обслуговування, який забезпечує затребуваний час виконання обчислювальних алгоритмів з врахуванням обмежень реконфігуровної обчислювальної системи. Запропонований метод дозволяє підвищити ефективність процесу управління обчислювальним процесом в реконфігуровних обчислювальних системах за рахунок вибору оптимального способу обслуговування для кожної задачі шляхом визначення обсягу непродуктивних часових витрат

Ключові слова: реконфігурові обчислювальні системи, якість обслуговування, накладні видатки реконфігурації, програмовні логічні інтегральні схеми (ПЛІС)

Исследуется процесс управления вычислениями в реконфигурируемых вычислительных системах. Предложен метод обеспечения временных требований качества обслуживания, который обеспечивает требуемое время выполнения вычислительных алгоритмов с учетом ограничений реконфигурируемой вычислительной системы. Предложенный метод позволяет повысить эффективность процесса управления вычислительным процессом в реконфигурируемых вычислительных системах за счет выбора оптимального способа обслуживания для каждой задачи путем определения объема непроизводительных временных затрат

Ключевые слова: реконфигурируемые вычислительные системы, качество обслуживания, накладные расходы реконфигурации, программируемые логические интегральные схемы (ПЛИС)

UDC 004.27
DOI: 10.15587/1729-4061.2016.81003

THE METHOD FOR PROVIDING QUALITY OF SERVICE TIME REQUIREMENTS IN RECONFIGURABLE COMPUTING SYSTEMS

Y. Kulakov
Doctor of Technical Sciences, Professor*
E-mail: ya.kulakov@gmail.com
I. Klymenko
PhD, Associate Professor*
E-mail: iklymenko@yandex.ua
V. Tkachenko
PhD, Associate Professor*
E-mail: tkvalentina@yandex.ua
O. Storozhuk
Postgraduate student*
E-mail: storozhuk.om@gmail.com
*Department of Computer Engineering
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"
Peremohy ave., 37, Kyiv, Ukraine, 03056

1. Introduction

A paradigm of the solution of wide classes of tasks is the modern trends of development of reconfigurable computing systems, including the tasks that require strict time restrictions of run time, including real-time tasks. Reconfigurable computing systems are characterized by existence of constant structure and variable – as a set of computing nodes, whose structure can be rebuilt. According to this definition, reconfigurable systems are characterized by particular functional and hardware limitations caused by the use of reprogrammed components – FPGAs (field programmable gate arrays).

The processes that occur in dynamically reconfigurable computing systems and that stipulate problems with Quality of Service (QoS) [1], considering hardware constraints of FPGA, are the object of research in this work. Considering the prospect of using reconfigurable high-performance computing systems with increasing the efficiency of parallel computing, the quality of service provision is an actual problem for reconfigurable high-performance computing systems [1–3].

The evolution of techniques of partial dynamic reconfiguration on FPGAs [4–6] opens new opportunities for an increase in efficiency of reconfigurable parallel computing systems due to implementation of dynamic reconfiguration of the computing structure, according to the requirements of executed applications. However, the traditional methods for providing QoS, developed for fixed computing systems aren`t effective for such systems, considering functional restrictions caused by delays in the process of dynamic creation of computing structure and spatial restrictions of FPGAs. Known methods of the accounting of these time delays can't be effectively used within time restrictions of the run applications because of impossibility to evaluate their volume for implementation of the computation control optimization.

That is why there is a need of developing methods and means to ensure the required quality of service in reconfigurable computing systems with functional restrictions, that impact on the computation time.

2. Literature review and problem statement

Ensuring the quality of service in dynamically reconfigurable computing systems gains a certain flexibility by allow-

ing implementation of the most effective computing structure or reducing the computation time by accelerating the reconfiguration. There are many methods and means of acceleration of reconfiguration, which belongs to the type of "Best Effort", and mainly based on techniques of reducing the time overhead. The most known among them are reuse of resources [3, 4], prefetching of configurations [5], clustering of configurations [2] and hardware means for increasing the performance of interfaces [6]. All of them provide the maximum possible computing acceleration without any optimization of usage of the plane of the crystal of FPGA. The problem with excessive usage of computing resources of FPGA is resolved by means of delays of the execution of tasks, defragmentation, and unloading the less critical tasks, HW/SW approach [4]. This, in turn, decreases the level of QoS assurance by additional time expenses and failures in execution of tasks.

To ensure QoS, considering the required amount of used hardware resources, the effective solution was found and it is proposed for a well-known cluster computing system with multicore processors in nodes and reconfigurable structure shared among them [1, 8]. The involved reconfigurable structure consists of a set of similar fine-grained and coarse-grained modules, connected by a common communication network. The principle of reconfiguration is based on an extension of system instruction set to accelerate functional cores – Instruction Set Extensions (ISE) [9]. Herewith, each task depending on its computational complexity can be solved with different performance on various sets of equipment. The idea of the distribution of tasks consists of determination of the minimum required set of equipment, which will provide the required application's execution time without excess equipment usage.

However, existence of predicated computing structure leads to necessity to bring the tasks to a determinant form, which is required by the system's structure, and the reconfiguration is considered only at the level of connections commutation. The overhead reduction issue isn't considered in this work, due to lack of reconfiguration of the computing structure.

The output computing application is a parallel program with mixed type of parallelism that is described by the programming model of M-tasks [10]. M-program is set by Macro Dataflow Graphs (MDG) [10] and macrograph tasks are placed in the vertices of it, and edges specify the relationship between vertices.

In this paper, the abstract concept of the hardware task is used, which in algorithms with the mixed type of parallelism corresponds to macrotasks that have some functional kernel of a computing algorithm synthesized in the digital circuit which in the algorithm mapping process is placed in the field of reconfigurable FPGA crystal. Each hardware task is associated with a work, that is defined by the amount of work, the appropriate certain top of a macrograph of the computing algorithm.

Let's present the output program as macrograph $G_M=(V_M, E)$, where V_M – a set of vertices corresponding to macro tasks, and E is a set of edges that define the relationship between macro-tasks. Mapping by levels is a common method of mapping tasks to computing structure. Herewith each level of the graph is sequentially mapped on computer system structure [10].

Runtime of the computational program is defined as the sum of execution time of the longest sequence of tasks of each level and described as follows:

$$T_{G_{M\_GRA}} = \sum_{k=1}^{w} \max \{T_{v_k} \left| v_k = \overline{1,H_k} \right.\},$$

where $k = \overline{1,w}$ – the number of the level, w – the amount of levels of the computing algorithm, $v_k = \overline{1,H_k}$ – node number on the level k, H_k – the amount of nodes on the level k. Accordingly, the next ordered set of macrotasks is:

$$B_{G_M} = \{T_{max\_k} \left| k = \overline{1,w} \right.\}, \quad (1)$$

where T_{max_k} – execution time of the longest task of the k-level, is the longest related sequence of tasks, which is the starting point for determining the total execution time of the program.

The runtime constraint is the time limit, set on the basis of certain external factors such as initial requirements of QoS of the application.

In this paper, the problem of time restrictions allocation among the sequences of executed tasks of the computational algorithm is resolved, to ensure execution of the algorithm in the dedicated time, efficient use of the FPGA resources and to reduce the number of tasks execution failures.

The structure of the dynamically reconfigurable computing system was designed, investigated and described in detail by the authors in the previous paper [7, 11, 12].

3. The purpose and research objectives

The purpose of this work is to improve the efficiency of reconfigurable computing systems, during the implementation of applications that put restrictions to the execution time.

To achieve the goal, the following tasks were set:

– to develop a means of improving the computation control process control in reconfigurable systems by assessment of unproductive time with regard to hardware limitations in FPGA while mapping computational algorithms to reconfigurable computing structure;

– to develop a method for providing requested quality of service in reconfigurable computing systems for applications that put restrictions to the execution time;

– to evaluate conditions of effective use of the offered facilities within the functional and hardware limitations of the reconfigurable computer system.

4. Determination and substantiation of major dependencies

The positive effect from hardware acceleration of the computing process in reconfigurable computing systems is measured by the following index of performance acceleration [1, 8]:

$$\rho = \frac{T_{SW}}{T_{HW}}, \quad (2)$$

where T_{SW} – time required for computing the i-task on the processor kernel, T_{HW} – time required for computing the i-task by means of hardware ($i = \overline{1,n}$, where n is the number of tasks in sequence (1), to be performed (n=w)).

To determine the time constraints during solving parallel M-programs, presented in the parallel form graph of

the algorithm, let's substantiate the possibility of using the well-known method [1], designed to determine the time constraints during solving of interrelated tasks in a multitasking mode. The method is based on using acceleration performance index (2) and assumption, that the overall time should be divided proportionally by the expected time of executing each task to determine the required amount of hardware to meet time constraints required by the application. Based on the stated above, we will present the well-known relation [1] in accordance with the problem statement of this paper:

$$T_{QoS\,i}=\left(T_{Count\,i}\times\frac{T_{QoS}}{\sum_{i=1}^{n}T_{Count\,i}}\right)\times\left(\frac{\rho_i}{\sum_{i=1}^{n}\rho_i/n}\right). \quad (3)$$

where T_{QoS} – overall time constraints of the program, $T_{QoS\,i}$ – time constraints, calculated for each i-task to be performed, based on time requirements T_{QoS}, T_{Count} – expected time of the i-task performance, ρ_i – productivity acceleration, calculated by the formula (2).

Define, that the computational time of some i-task using hardware resources is equal to $T_{Count\,i}=T_{Rconf\,i}+T_{HW\,i}$, where $T_{Rconf\,i}$ – reconfiguration time, spent directly on placing the appropriate configuration of appropriate hardware i-task on the fabric area of the reconfigurable computing module. Then, according to the equation (2), the performance acceleration index, which we will define as the factor of acceleration, equals

$$\rho_i=\frac{T_{SW\,i}}{T_{Count\ i}}, \quad (4)$$

To prove the truth of the expression (3) and to receive the proportion between the main parameters, the geometric interpretation is shown (Fig. 1).

The diagram in Fig. 1 shows the total computational time with software $\sum_{i\ 1}\ _{SW\,i}$ and computational time with hardware acceleration $T_{Total}=\sum_{i=1}^{n}T_{Count\,i}$, K_{Av} – averaging factor, $T_{Av\,i}$ and $\sum_{i=1}^{n}T_{SW\,Av\,i}$ – computation time of the i-task and the total computation time, respectively, calculated on the basis of averages. Line (I) shows the distribution of time intervals for the expected computation time parameters of tasks, line (II) – for the averaged parameters, line (III) – depict the process of correcting time intervals taking into account the parameters of quality of service QoS.

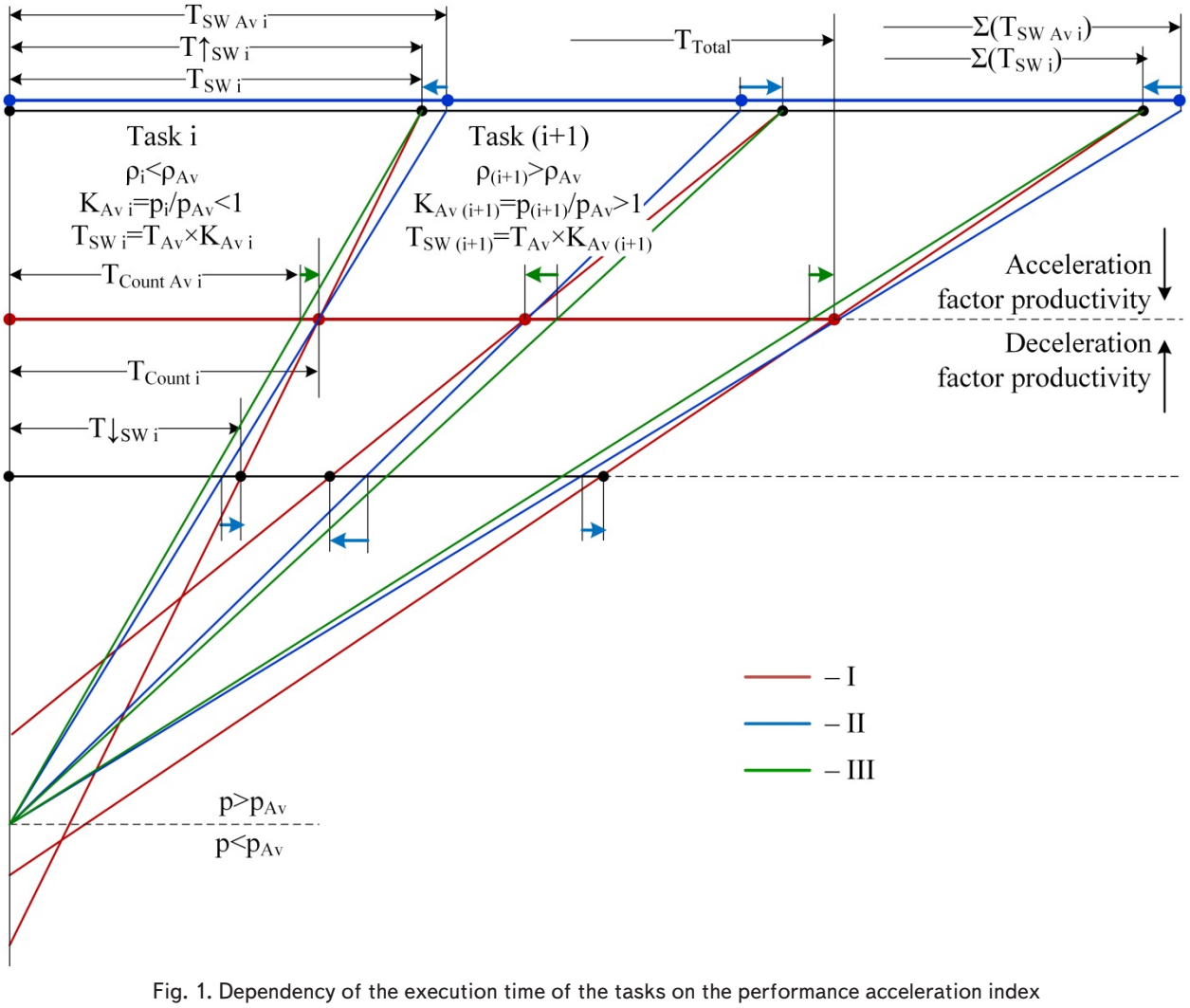

Fig. 1. Dependency of the execution time of the tasks on the performance acceleration index

The diagram in Fig. 1 shows that the total computation time with software and the total computation time with hardware acceleration, as well as all appropriate intervals that define each task in sequence of computations are in a proportional relation to the average acceleration factor ρ_{Av}:

$$\frac{\sum_{i=1}^{n} T_{SW\,i}}{\sum_{i=1}^{n} T_{Count\,i}} = \frac{T_{SW\,i}}{T_{Count\,i}}\Bigg| i = \overline{(1,n)} = \rho_{Av},$$

then

$$T_{SW\,i} = T_{Count\,i} \times \frac{\sum_{i=1}^{n} T_{SW\,i}}{\sum_{i=1}^{n} T_{Count\,i}} = T_{Count\,i} \times \rho_{Av}.$$

Time intervals, obtained using the average acceleration factor, require correction over the deviation from the average value, which we will define as the averaging factor. The averaging factor is calculated by the following formula:

$$K_{Av\,i} = \frac{\rho_i}{\left(\sum_{i=1}^{n} \rho_i\right)/n},$$

where

$$\left(\sum_{i=1}^{n} \rho_i\right)/n = \rho_{Av}$$

is average acceleration factor. The diagram (Fig. 1) shows; that if the acceleration factor of the i-task of the sequence ρ_i is higher than the average acceleration factor ρ_{Av} ($K_{Av}>1$), this time needs to be corrected towards increase, and vice versa, if the acceleration factor of the i-task of the sequence ρ_i is less than the average acceleration factor ρ_{Av} ($K_{Av}<1$), this time requires a correction towards decrease.

Thus, after carrying out correction, we get initial values of computational time without acceleration: the total –

$$T_{Av\ Total} = \sum_{i=1}^{n} T_{SW\ Av\,i},$$

for each task – $T_{SW\,i} = T_{Av\,i} \times K_{Av\,i}$.

Based on foregoing, we define the following dependence of the execution time of the task on its acceleration factor:

$$T_{SW\,i} = T_{Count\,i} \times \frac{\sum_{i=1}^{n} T_{SW\,i}}{\sum_{i=1}^{n} T_{Count\,i}} \times \frac{\rho_i}{\left(\sum_{i=1}^{n} \rho_i\right)/n} =$$

$$= T_{Count\,i} \times \rho_{Av} \times \frac{\rho_i}{\rho_{Cp}} = T_{Count\,i} \times \rho_i, \qquad (5)$$

according to the equation (4), it proves the validity of the equation (3) for the initial settings described above.

The following dependencies between the main computation parameters can be obtained from the diagram (Fig. 1):

$$T_{Count\,i} = \frac{T^{\uparrow}_{SW\,i}}{\rho^{\uparrow}_i}, \quad T_{Count\,i} = T^{\downarrow}_{SW\,i} \times \rho^{\uparrow}_i = \frac{T^{\downarrow}_{SW\,i}}{\rho^{\downarrow}_i},$$

if $\rho^{\uparrow}_i = \frac{1}{\rho^{\downarrow}_i}$.

Accordingly,

$$\rho^{\uparrow}_i = \frac{T^{\uparrow}_{SW\,i}}{T_{Count\,i}}, \quad \rho^{\downarrow}_i = \frac{T^{\downarrow}_{SW\,i}}{T_{Count\,i}}.$$

If $T^{\uparrow}_{SW\,i}$ and $T^{\downarrow}_{SW\,i}$ are initial time parameters, corresponding to the input time value $T_{SW\,i}$, where the index ($\uparrow$) represents the computational acceleration and ($\downarrow$) – the deceleration, its relation to the desired task execution time on the hardware will determine the acceleration factor value, according to the equation (4), and the acceleration factor value will determine the acceleration (if $\rho_i>1$) or the deceleration (if $\rho_i<1$) of the computational process. Then the equation (5) is appropriate for cases of computation acceleration so as for cases of deceleration. Based on the equation (5), we will obtain the dependence of the task runtime on the performance acceleration factor:

$$T_{Count\,i} = T_{SW\,i} \times \frac{\sum_{i=1}^{n} T_{Count\,i}}{\sum_{i=1}^{n} T_{SW\,i}} \times \frac{\rho_i}{\left(\sum_{i=1}^{n} \rho_i\right)/n},$$

$$\text{if } \rho_i = \frac{1}{\rho^{\uparrow}_i} = \rho^{\downarrow}_i = \frac{T_{Count\,i}}{T_{SW\,i}}. \qquad (6)$$

4. 1. Formalization of the modified method of determining the time intervals of tasks under time constraints required by QoS

The initial parameter for solving tasks of providing required time limits is an overall time-limit T_{QoS} of the calculation algorithm, which is imposed by certain external factors. The initial parameters of the system are time parameters of each task $T_{SW\,i}$, $T_{Count\,i}$ and values of performance acceleration factors calculated in the equation (4).

The time constraints imposing requires providing certain acceleration from executed tasks $\rho_{QoS\,i}$, i. e. the execution time of each task from the tasks sequence is imposed by the following restrictions:

$$T_{Count\,i} \le T_{QoS\,i} = \frac{T_{SW\,i}}{\rho_{QoS\,i}},$$

in case of total computation acceleration, i. e.

$$\rho_i = \frac{T_{SW\,i}}{T_{Count\,i}} > 1.$$

To solve the task considering restrictions, we will calculate the value $T_{QoS\,i}$ according to the equation (6). From the previous studies it comes out that

$$T_{QoSi} = T_{SW\,i} \times \frac{\sum_{i=1}^{n} T_{QoSi}}{\sum_{i=1}^{n} T_{SW\,i}} \times \frac{\rho_{QoS\,i}}{\left(\sum_{i=1}^{n} \rho_{QoS\,i}\right)/n}.$$

Thus, the expression is as follows:

$$T_{QoS\,i} = T_i \times \left[\frac{T_{QoS}}{\sum_{i=1}^{n} T_{Count\,i}} \times \frac{\rho_{QoS\,i}}{\left(\sum_{i=1}^{n} \rho_{QoS\,i} \right) / n} \right]_{Av}^{Correct} \times \frac{\rho_i}{\rho_{Av}}.$$

Comparing with the well-known method [4], the obtained expression considers a deviation, which is brought by the assumption that the given time limit T_{QoS} is the average amount of time, and the acceleration factor, required by constraints, is the average acceleration factor. In this case, the following decision would be right:

$$\rho_{QoS} = \frac{\sum_{i=1}^{n} T^{\uparrow}_{SW\,i}}{T_{QoS}} = \frac{\sum_{i=1}^{n} T_i \times \rho_{Av}}{T_{QoS}}, \quad T^{\uparrow}_{SW\,i} = T_i \times \rho_i,$$

hence

$$T_{QoS\,i} = \frac{T^{\uparrow}_{SW\,i}}{\rho_{QoS\,i}} = T_i \times \left[\frac{T_{QoS}}{\sum_{i=1}^{n} T_{Count\,i}} \right]_{Av}^{Error} \times \frac{\rho_i}{\rho_{Av}}.$$

In case of the ordered sequence of tasks and to address goals of this article, computations should be deprived of deviations. Geometric interpretation (Fig. 1) shows that the total computational time correction to the average value is equal to the total time value correction in the opposite direction. In case of preserving factors of deviation from the average before computations, it would be advisable to adjust the output time T_{QoS} to the following magnitude, in order to bring the time to the average value:

$$K_{QoS\,Correct} = \frac{\sum_{i=1}^{n} T^{\uparrow}_{SW\,i}}{\sum_{i=1}^{n} T_{Count} \times \rho_{Av}},$$

then, the corrected value of the average acceleration, required by the time constraints equals

$$\rho_{QoS} = \frac{T_{QoS}}{\sum_{i=1}^{n} T_{Count\,i}} \times \frac{\sum_{i=1}^{n} T^{\uparrow}_{SW\,i}}{\sum_{i=1}^{n} T_{Count} \times \rho_{Av}}. \quad (7)$$

The final equation for the calculations is as follows:

$$T_{QoSi} = T_i \times \left[\frac{T_{QoS}}{\sum_{i=1}^{n} T_{Count\,i}} \times \frac{\sum_{i=1}^{n} T^{\uparrow}_{SW\,i}}{\sum_{i=1}^{n} T_{Count} \times \rho_{Av}} \right] \times \frac{\rho_i}{\rho_{Av}}. \quad (8)$$

Execution of time constraints in general, is given by the equation:

$$T_{Count\,i} \le T_{QoS\,i}. \quad (9)$$

Geometric interpretation of performed calculations is shown in Fig. 2.

4. 2. Determination of the acceptable boundaries of time constraints

Based on the above evidence, calculations and final equations (8) and (9) we can determine the following conditions for constraints execution:

1. Cases, when the acceleration of the computational process is less than 1 ($\rho_{Cp}<1$, $\sum_{k=1}^{n} T_{SW\,i} < \sum_{i=1}^{n} T_i$) that is when the overall computational process slows down (Fig. 1, 2) are not considered. Processor cores can execute such algorithms.

2. For the cases, when time constraints require more or less acceleration from the sequence of tasks than their overall average acceleration, conditions of fulfillment of constraints are shown in (Table 1).

3. The maximum factor of deviation from the average value is used during definition of the permissible boundaries of constraints execution (Table 1):

– maximum index of deviation towards a computation acceleration requirement:

$$\vartheta^{\uparrow} = \max(K^{\uparrow}_{Av\,i} \,|\, \forall (i = \overline{1,n},\ \rho_i > \rho_{Av}));$$

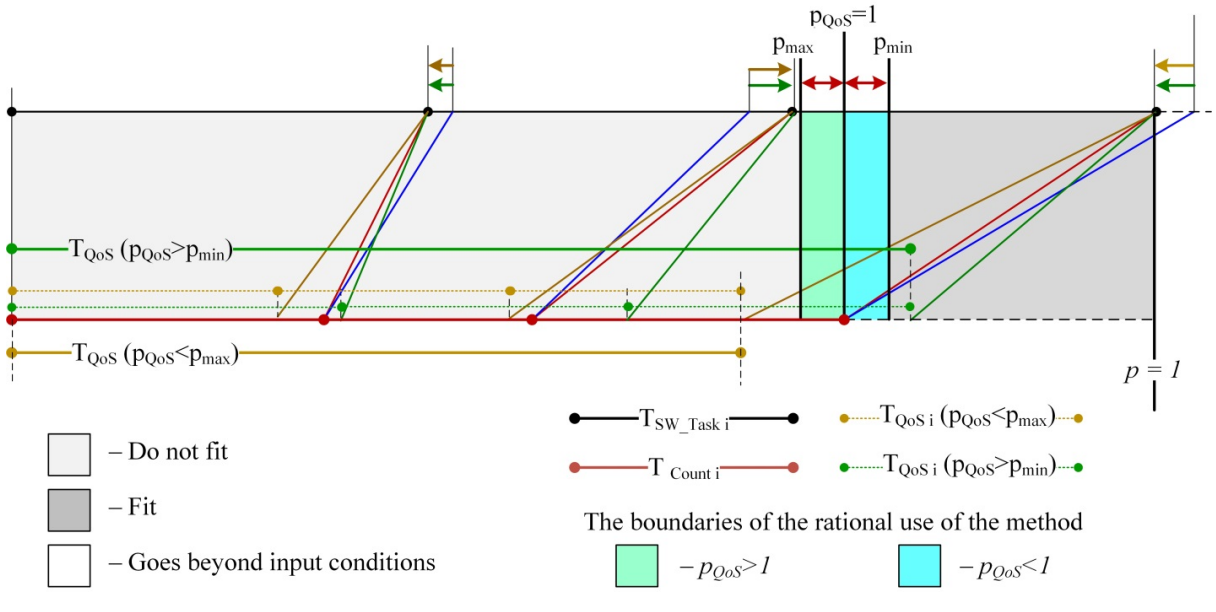

Fig. 2. Determination of the boundaries of the rational use of the proposed method

– maximum index of deviation towards a computation deceleration requirement:

$$\vartheta^{\downarrow} = \max(K^{\downarrow}_{Av\,i} \,|\, \forall(i = \overline{1,n},\ \rho_i < \rho_{Av})).$$

In general, the maximum factor of deviation from the average value is defined as follows:

$$\vartheta = \max\left(\frac{\rho_i}{\rho_{Cp}} \,|\, \forall(i = \overline{1,n})\right),$$

then

$$\rho_{max} = \rho_{QoS} \times \vartheta \,|\, (\vartheta > 1),\ \ \rho_{min} = \rho_{QoS} \times \vartheta \,|\, (\vartheta < 1),$$

where ρ_{QoS} is calculated according to the equation (8).

Table 1

Time constraints execution conditions

Application settings	Task settings	Comment
$\rho_{max} > \rho_{QoS} > \rho_{Av}$ $\rho_{max} \to \rho_{Av}$	$K_{Av\,i} > 1,\ \ \rho_i > \rho_{Av}$	Fit (acceleration of tasks is more than the average)
	$K_{Av\,i} < 1,\ \ \rho_i < \rho_{Av}$	Do not fit (acceleration of tasks is less than the average)
$\rho_{max} < \rho_{QoS} > \rho_{\Sigma\,Av}$	For all tasks	None of the tasks fit into constraints
$\rho_{min} < \rho_{QoS} < \rho_{Av}$ $\rho_{QoS} > 1$ $\rho_{min} \to \rho_{Av}$	$K_{Av\,i} > 1,\ \ \rho_i > \rho_{Av}$	Fit (acceleration of tasks is more than the average)
	$K_{Av\,i} < 1,\ \ \rho_i < \rho_{Av}$	Do not fit (acceleration of tasks is less than the average)
$\rho_{min} > \rho_{QoS} > 1$	For all tasks	All tasks fit into the constraints

It is advisable to introduce flexible limits for the maximum factor of deviation from the average value, which can be defined as follows:

$$\rho_{max} \to \rho_{Av},\ \ \rho_{min} \to \rho_{Av}.$$

For this purpose, it is possible to use the standard deviation index δ, which is a measure of spreading the values of their average rates. The tasks, which acceleration of productivity approaches the values 3δ ($\rho_i \to 3\delta$), significantly affect the uniformity of time constraints distribution and it will lead, from the practical side of implementation, to excessive actions to minimize the time of task execution, and, as a consequence, to the excessive use of hardware and software resources of the system.

It is possible to form the following permissible boundaries of the specified time constraint, which are based on defined conditions and determine the case for rational usage of the proposed method:

$$\begin{cases} \rho_{Av} > 1, \\ \rho_{max} > \rho_{QoS} > 1, \\ \rho_{max} \to \rho_{Cp},\ \rho_{min} \to \rho_{Av}. \end{cases} \qquad (10)$$

Therefore, based on the demanded QoS time constraints, the proposed modified method of determination of the tasks time is as follows:

1. The specified time constraint is divided into parts inversely proportional to the execution time of each task, according to the equation (7) in the following order:

– time constraint correction to the average value is performed;

– average time constraint distribution is performed proportionally to the average performance acceleration, getting the average intervals restrictions;

– correction of the averaged intervals towards an opposite direction of the averaging factor of each task is performed.

2. Obtained intervals are analyzed according to the general attributes of performance constraints (8).

5. Method for providing QoS time requirements in reconfigurable computing systems

The proposed modified method for determination of tasks runtime and previously defined attributes of providing time requirements (Table 1) are used to implement the method for providing time requirements of QoS, consisting of the following stages:

Step 1. Determinate the rational boundary of providing timing conditions (10):

$$\begin{cases} \rho_{Av} > 1, \\ \rho_{max} > \rho_{QoS} > 1. \end{cases}$$

Step 2. Delete the sets of tasks D ($T_i \in D$, if $T_{Rconf\,i} = \max$) – the tasks which time cannot be reduced, in particular, the tasks of the first level of the parallel form graph from the sequence of tasks B (1):

$$T_{v_1} \in D,\ \text{if}\ T_{v_1} = \max\{T_{v_1} \,|\, v_1 = \overline{1,H_1}\},$$

then

$$T_{QoS} = T_{QoS} - \sum_{i=1}^{n} (T_i \,|\, \forall T_i \in D).$$

Step 3. Ensure conditions fulfillment $\rho_{max} \to \rho_{QoS}$.

Step 4. Calculate time constraints for tasks that remain in the sequence (8):

$$T_{QoSi} = T_i \times \left[\frac{T_{QoS}}{\sum_{i=1}^{n} T_{Count\,i}} \times \frac{\sum_{i=1}^{n} T^{\uparrow}_{SW\,i}}{\sum_{i=1}^{n} T_{Count\,i} \times \rho_{Av}} \right] \times \frac{\rho_i}{\rho_{Av}} \,|\, \forall(T_i \notin D).$$

Step 5. Determinate the concept of reducing the execution time for tasks that do not fit into the time limits, based on the maximum or partial reduction of the reconfiguration time parameter:

$$\rho_i \,|\, \forall(T_{Count\,i} < T_{QoSi}) = \frac{T_{SW\,i}}{T_{Count\,i}} = \frac{T_{SW\,i}}{T_{Rconf\,i} + T_{HW\,i}} \Rightarrow$$

$$\Rightarrow \begin{cases} T_{Rconf\,i} \to \min, \\ T_{Count\,i} \to T_{HW\,i}. \end{cases}$$

6. Experimental researches of effectiveness of the method of providing service quality time requirements in reconfigurable computing systems

Theoretical research of the effect of application of the proposed method of time distribution is completed. Randomly synthesized sequences of tasks were researched. The characteristics of hardware functional cores, which were synthesized on FPGA Cyclone II Altera, were taken as their timing parameters. Diagrams (Fig. 3) show that inversely proportional correction (T_{QoS_Mod}) of averaged time intervals (T_{QoS_Av}) provides additional time resources to the critical tasks, which have high acceleration factor (higher than average). The graph shows that Tasks (2, 5–6) are no longer critical, because they got extra runtime. It can be seen that the Task 5, whose acceleration factor is slightly lower than average, also got enough runtime.

This distribution is effective on condition of uniform deviation of the acceleration factor from the average value. Otherwise, we can see excessive allocation of time for the tasks, which have acceleration factor higher than the average value. Further, in the sequence of tasks there may be the tasks, the runtime of which cannot be reduced, i.e. the tasks of the first level of the parallel form of the algorithm. Therefore, during the allocation, these tasks should be defined as those, which cannot be effectively executed with the usage of the proposed method, as was described above. These tasks should be deleted from a general sequence of proposed method applying and required execution time should be given to them.

The software emulator of reconfigurable computing systems and software model of the implementation of the proposed method [12] were designed for simulation of the proposed method for providing time requirements of QoS for reconfigurable computing systems. Efficiency of the software model is proved by modeling the functional elements of the system with timing characteristics, maximally approached to the real. Timing characteristics of functional blocks of tasks, module components and functional processes of a computing system are obtained from the developed models, which were synthesized by the usage of Verilog language and implemented on FPGA Cyclone II Altera.

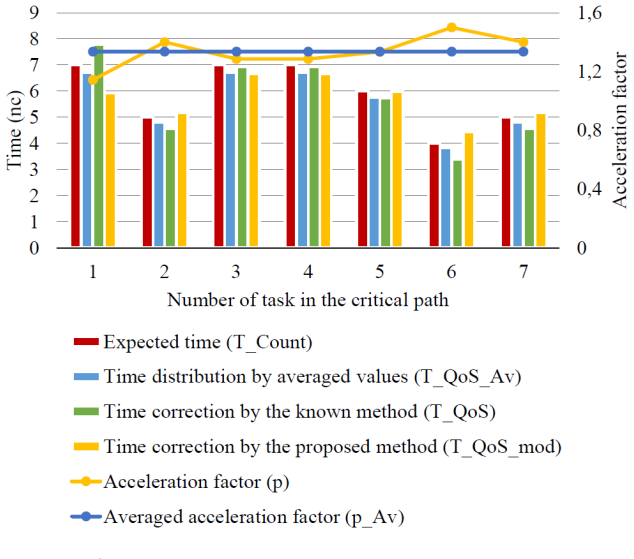

Fig. 3. Providing additional runtime for critical tasks

The research is conducted for a set of applications presented as parallel form graphs of the algorithms. Based on conducted experiments, comparative dependences of execution time for different mechanisms of accelerating the computational process for algorithms with different number of similar tasks are obtained (Fig. 4).

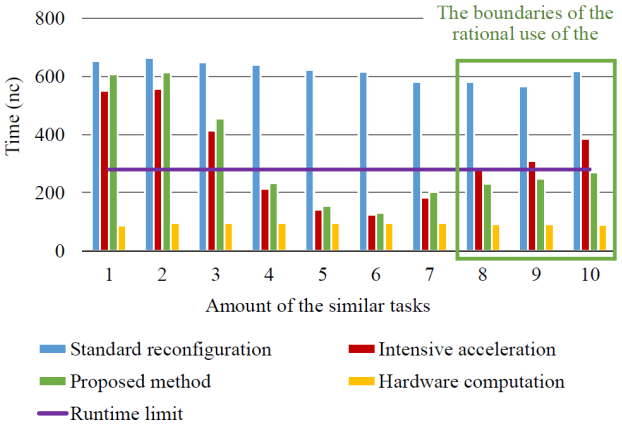

Fig. 4. Analysis of the reconfiguration time

The diagram Fig. 4 shows that the reconfigurable computations performed by means of standard reconfiguration sequence require significant time of execution in case with no acceleration due to a significant advantage of reconfiguration time over execution time of the task. Herewith, the computation time does not depend on the type of tasks performed. Removing similar tasks' reloading leads to intensive acceleration. Efficient method of intensive acceleration was proposed and described by the authors in the previous paper [12]. The graph shows that the minimum computation time is achieved provided that the width of the MDG graph corresponds to spatial parameters of the reconfigurable FPGA environment [11]. With the increasing number of similar tasks, it is necessary to introduce additional tools for loading of the active tasks. It requires extra time as shown in Fig. 4. The proposed method allows optimizing the proposed technology of intensive acceleration [12]. The diagram demonstrates that the proposed method allows to stay within time constraints, required by the application for cases with a large number of similar tasks.

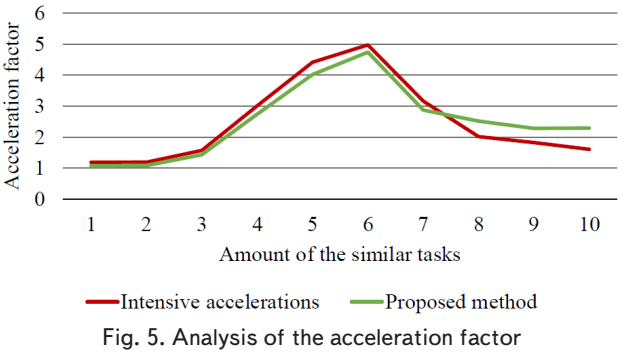

Fig. 5. Analysis of the acceleration factor

According to the analysis of the acceleration factor, the mechanism of resources reuse provides an intensive acceleration of reconfiguration with increasing the number of tasks types to the amount that is commensurate with the width of the parallel form graph. Increase in intensity of acceleration of reconfiguration on average by 63 % is achieved, if the

reconfigured area is commensurate with the width of the parallel form graph and there are many similar tasks. Herewith, there is a sharp decrease in the intensity of acceleration of reconfiguration, on average by 85 % during the process of overcoming the spatial constraints of FPGA.

The proposed method for ensuring time constraints reduces the intensity of the impact of spatial constraints on the computation speed, in this case approximately by 10 %. However, this measure depends on the defined amount of reconfiguration overhead with criteria of providing time and hardware restrictions of the reconfigurable computing systems.

7. Discussion of the results of research of the efficiency of the method of providing quality of service time requirements in reconfigurable computing systems

From a practical point of view, we can interpret the proposed method of determining time-limit intervals as follows. The correction of the averaged periods is performed by allocation of an additional time of execution to the tasks, which have high acceleration factor (higher than average), using tasks with low acceleration factor (lower than average). Considering the acceleration factor as a measure of effective hardware task implementation, tasks with high acceleration factor can be effectively solved by the means of the standard sequence of the reconfiguration process. Otherwise, it is necessary to introduce additional means of tasks implementation acceleration, particularly, by reducing reconfiguration time.

The use of the service quality support method allows to define such sequence of tasks for which the target architecture of the computing system is effective and the use of additional mechanisms of an acceleration of reconfiguration and reduction of overhead costs [2–5, 12] doesn't lead to an acceleration of computation within the exposed temporal requirements. Resolving of these tasks by standard utilities of reconfigurable computation reduces excess use of the hardware and hardware-software resources of the reconfigurable computing system. The use of the offered method allows to reduce the influence of space constraints for the period of reconfiguration and to reduce the quantity of deviations of execution of tasks in case of dynamic mapping of the flow of tasks.

Modification of the quality provision method allows to determine a sequence of tasks execution within implementation of the service quality support method in the reconfigurable computing systems. The paper mathematically proved the increase in accuracy of calculations, in comparison with the known method. Mathematical reasons for the offered means prove the reliability of the use of the offered method and the method of its implementation for computing algorithms with the mixed type of parallelism provided by graphs of parallel form graphs of the algorithm.

The offered method allows to provide a wide class of tasks with effective target computing structure for achievement of the necessary quality level of service, and also to reduce the quantity of failures in case of distribution of a flow of tasks, arriving dynamically.

The offered method can be used for an increase in efficiency of high-performance reconfigurable computing systems in case of the solution of tasks of control of different technical and technological processes and implementation of multivariate computation in complex information systems.

8. Conclusions

1. The method of determining the time intervals of tasks execution by taking into account their performance acceleration was modified and mathematically substantiated. The offered modification, based on the analysis of tasks acceleration index, allows to evaluate unproductive time expenditure taking into account FPGA hardware constraints in case of computing algorithms display to a reconfigurable computing structure. It allows to increase efficiency of the computation control process in the reconfigurable systems due to determining the tasks that meet time requirements and won't demand involvement of intensive reconfiguration acceleration mechanisms.

2. The method for providing time requirements of quality of service (QoS) in reconfigurable computing systems, as opposed to well-known methods, based on the maximum possible (intensive) reduction of unproductive time costs, provides a given application runtime by determining the unproductive time-consuming reconfiguration amount and the choice, based on this, of optimal service discipline for each task in terms of current time and hardware limitations was proposed and substantiated.

3. The formalization of the method for determining the time intervals of tasks was developed, for which the optimal boundaries of the effective use of the proposed method to ensure quality of service and its implementation, in terms of the adequacy of the offered application time limits to the limits of the reconfigurable computer system were defined and proved.

References

1. Ahmed, W. Adaptive Resource Management for Simultaneous Multitasking in Mixed-Grained Reconfigurable Multi-core Processors [Text] / W. Ahmed, M. Shafique, L. Bauer, J. Henkel // Proceedings of the seventh IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis - CODES+ISSS '11, 2011. – P. 365–374. doi: 10.1145/2039370.2039426
2. Huang, M. Reconfiguration and Communication-Aware Task Scheduling for High-Performance Reconfigurable Computing [Text] / M. Huang, V. K. Narayana, H. Simmler, O. Serres, T. El-Ghazawi // ACM Transactions on Reconfigurable Technology and Systems. – 2010. – Vol. 3, Issue 4. – P. 1–25. doi: 10.1145/1862648.1862650
3. Bassiri, M. M. Mitigating Reconfiguration Overhead In On-Line Task Scheduling For Reconfigurable Computing Systems [Text] / M. M. Bassiri, S. H. Shahriar // 2010 2nd International Conference on Computer Engineering and Technology, 2010. – P. 397–402. doi: 10.1109/iccet.2010.5485509

4. Al-Wattar, A. Efficient On-line Hardware/Software Task Scheduling for Dynamic Run-time Reconfigurable Systems [Text] / A. Al-Wattar, S. Areibi, F. Saffih // 2012 IEEE 26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum, 2012. – P. 401–406. doi: 10.1109/ipdpsw.2012.50
5. Taher, M. Virtual Configuration Management: A Technique for Partial Runtime Reconfiguration [Text] / M. Taher, T. El-Ghazawi // IEEE Transactions on Computers. – 2009. – Vol. 58, Issue 10. – P. 1398–1410. doi: 10.1109/tc.2009.81
6. Liu, S. Achieving Energy Efficiency through Runtime Partial Reconfiguration on Reconfigurable Systems [Text] / S. Liu, R. N. Pittman, A. Forin, J.-L. Gaudiot // ACM Transactions on Embedded Computing Systems. – 2013. – Vol. 12, Issue 3. – P. 1–21. doi: 10.1145/2442116.2442122
7. Kulakov, Y. O. The multilevel memory organisation in the reconfigurable computing systems [Text] / Y. O. Kulakov, I. A. Klymenko // Visnyk NTUU "KPI". Informatyka, upravlinnia ta obtchislyuvalna technika. – 2015. – Vol. 61. – P. 18–26.
8. Ahmed, W. mRTS:Run-Time System for Reconfigurable Processors with Multi-Grained Instruction-Set Extensions [Text] / W. Ahmed, M. Shafique, L. Bauer, J. Henkel // 2011 Design, Automation & Test in Europe, 2011. – P. 1–6. doi: 10.1109/date.2011.5763246
9. Koenig, R. KAHRISMA: A Novel Hypermorphic Reconfigurable-Instruction-Set Multi-grained-Array Architecture [Text] / R. Koenig, L. Bauer, T. Stripf, M. Shafique, W. Ahmed, J. Becker, J. Henkel // 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), 2010. – P. 819–824. doi: 10.1109/date.2010.5456939
10. Dümmler, J. Scalable computing with parallel tasks [Text] / J. Dümmler, T. Rauber, G. Rünger // Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers – MTAGS '09, 2009. – P. 1–10. doi: 10.1145/1646468.1646477
11. Klymenko, I. A. The effectiveness analysis of resources management in reconfigurable computer systems [Text] / I. A. Klymenko // Visnyk NTUU "KPI". Informatyka, upravlinnia ta obtchislyuvalna technika. – 2015. – Vol. 62. – P. 11–21.
12. Kulakov, Y. O. Development of the reconfiguration acceleration method in the dynamically reconfigurable computing systems [Text] / Y. O. Kulakov, I. A. Klymenko, M. V. Rudnytskyi // Eastern-European Journal of Enterprise Technologies. – 2015. – Vol. 4, Issue 4 (76). – P. 25–29. doi: 10.15587/1729-4061.2015.47227

Описано методику та результати дослідження впливу форми напруги живлення рентгенівського випромінювача на загальну дозу опромінення пацієнтів під час виконання медичних обстежень. Надано результати експериментальних вимірів, описано алгоритм розрахунку дози опромінення, проведено порівняльний аналіз дозового навантаження на пацієнтів при використанні різних видів джерел живлення та аналіз ефективності модернізації існуючого рентгенівського обладнання

Ключові слова: чисельне моделювання, еквівалентна доза опромінення, медичні рентгенівські обстеження, джерела живлення

Описаны методика и результаты исследования влияния формы напряжения питания рентгеновского излучателя на общую дозу облучения пациентов во время выполнения медицинских обследований. Приведены результаты экспериментальных измерений, описан алгоритм расчета дозы облучения, выполнен сравнительный анализ дозовой нагрузки на пациентов при использовании различных видов источников питания и анализ эффективности модернизации существующего рентгеновского оборудования

Ключевые слова: численное моделирование, эквивалентная доза облучения, медицинские рентгеновские исследования, источники питания

UDC 004.94 : 621.386.12
DOI: 10.15587/1729-4061.2016.81305

EXPLORING DEPENDENCE OF THE PATINTS' RADIATION DOSE ON THE FORM OF POWER SUPPLY VOLTAGE OF X-RAY TUBE

S. Reva
Senior Lecturer*
E-mail: iec-lab@karazin.ua
M. Malakhova
Postgraduate student*
E-mail: maryna.malakhova88@gmail.com
*Department of Electronics and Control Systems
V. N. Karazin Kharkiv National University
Svobody sq., 4, Kharkiv, Ukraine, 61022

1. Introduction

In modern medicine, one of the most common and important methods of diagnosis is the X-ray study. According to experts' estimations, establishing of more than 80 % of the diagnoses that require serious medical intervention is performed using X-ray images, results of roentgenoscopy and X-ray tomography. X-ray images are the main tool for the