SUPERCOMPUTING THE ELECTRON TRANSPORT USING CUDA TECHNOLOGY

M. ZHUKOVSKIY[*], M. MARKOV[*], S.PODOLYAKO[*] AND R. USKOV[*]

[*] Keldysh Institute for Applied mathematics of RAS
Moscow, Russia
e-mail: usermath@mail.ru, web page: http://www.keldysh.ru

Summary. Statistical algorithms are presented for modeling the interaction processes between electrons and matter. A software implementation has been developed for hybrid supercomputers making use of NVIDIA© CUDA© technology. Standard Monte Carlo schemes are modified for effectively exploiting the parallel computing capabilities of graphical processors. Modeling inter-action processes of electrons with objects of a complex non-uniformly scaled structure requires developing fine-grained models of the interaction without using well-known approximations. The model of individual collisions (MIC) is worked out to describe the interaction of electrons with atoms. MIC does not include the approximations assumed in multiple collision theory or the continuous slow down approach. The distributions of electron characteristics are obtained from tabulated electron cross-section data. Modeling hundreds of thousands of electron collisions with small energy transfers within the scope of MIC requires a huge amount of calculations. Therefore, modern computers based on hybrid architecture are required for computing electron transport using MIC. A corresponding parallelization technique is developed for using these hybrid computers. The discussed examples demonstrate the applicability of the algorithms to investigating the interaction of electrons with X-ray tube targets producing Bremsstrahlung and to researching the process of electron emission from surfaces of objects being under radiation.

Keywords. Electron Transport, Monte-Carlo Method, Mathematical Modeling, Graphical Processor, CUDA technology

Acknowledgement. The work was partially supported by RFBR grants No **14-01-00350** and **№ 15-01-03027**.

2010 Mathematics Subject Classification: 97M50, 93A30.
Key words and Phrases: Electron Transport, Monte-Carlo Method, Mathematical Modeling, Graphical Processor, CUDA technology

1 INTRODUCTION

Modeling interaction processes of electrons with objects of a complex non-uniformly scaled structure, including microstructure elements, requires developing fine-grained models of the interaction without using well known approximations, such as slow down approximation, embedded trajectories, etc. One such fine-grained model is the model of individual collisions (MIC) described in this paper. The description is based on the model[1]. Modeling hundreds of thousands of electron collisions with small energy transfers within the scope of MIC requires a huge amount of calculations. Therefore, modern computers based on hybrid architecture are required for computing electron transport using MIC. In addition, a corresponding parallelization technique is needed for using these hybrid computers. Algorithms for modeling the X-ray transport using graphical processors have been developed with application of the NVIDIA© CUDA© technology[2,3]. An approach to Monte Carlo modeling the electron transport is considered in this paper.

2 MODELING OF ELECTRON COLLISIONS

2.1. Model of interactions

The complex process of an electron passing through matter can be represented as a sequence of elementary interaction processes. Elastic scattering is usually considered within the bounds of the approximate Goudsmit and Saunderson theory of multiple scattering[4]. Inelastic interactions are described in known works using various modifications of slow-down approximation.

The model of individual collisions (MIC), completely based on elementary cross-section data[5, 6], is described below. Neither multiple scattering theory nor slow-down approximation is used.

The main goal of developing MIC is to obtain probabilistic distributions for changing of electron characteristics (motion direction, energy) when an electron interacts with matter.

2.2. Monte Carlo modeling of electron transport on hybrid computers

Statistically evaluating the mathematical expectation of the required functional, corresponding to the measured value, using the Monte Carlo method implies independent modeling of random trajectories of electrons and determination of the additive contribution of every trajectory to the overall result. The computing scheme is the same for every trajectory.

Such algorithms have a large number of independent calculation branches. Therefore they are easily parallelized and scale well. The parallelization can be done on computers architected for a minimum of interaction between parallel branches. The NVIDIA© CUDA© technology is realized on graphical processors intended for carrying out a large number of similar opera-tions[3].

Algorithms for modeling electron trajectories can show considerable differences, but quite a number of general features can be noted.

First, the parallel algorithm has to consist of a number of computational threads modeling separate trajectories. So far calculations are for the most part independent, and therefore the threads do not require synchronization. Such a computing structure is well suited for parallel

architectures of SIMD (Single Instruction Multiple Data) type, which are typical for modern graphical processors (GPU).

Second, one of the most important conditions for efficiency of the algorithm implementation is accurate data processing in the memory of GPU[7,8]. Usually three types of data are processed when modeling the electron trajectories:

- Parameters of modeled particles;
- Probabilistic distributions of MIC;
- Object parameters.

The parameters of the electrons are coordinates, motion direction, energy, and statistical weight. A sequential layout of the parameters in computer memory is natural for a single-processor (CPU) implementation (Figure. 1).

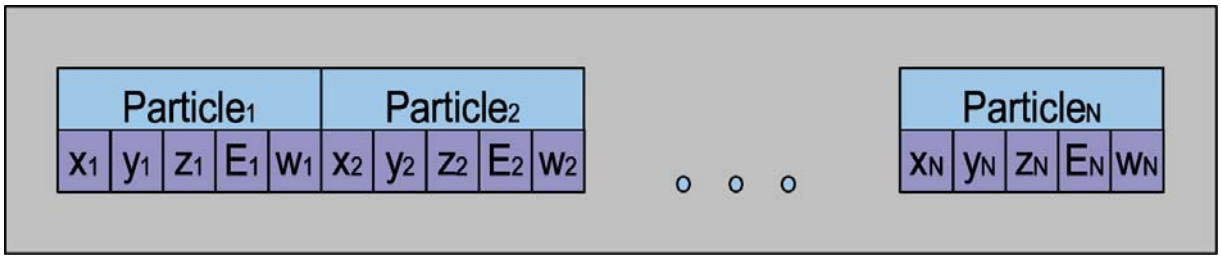

Figure. 1 – optimal data storage layout for CPU.

Such layout is inefficient when used on a GPU, because it does not ensure the fulfillment of the connectivity condition for memory requests. The union of logical memory cell requests from different execution threads in one physical request is called connectivity of memory requests[7]. This union considerably increases the memory productivity and is required for efficient use of the slow "global" memory of the GPU.

A much more efficient data layout is achieved by placing the same parameter of all electrons consecutively in memory for all modeled particles (Figure. 2).

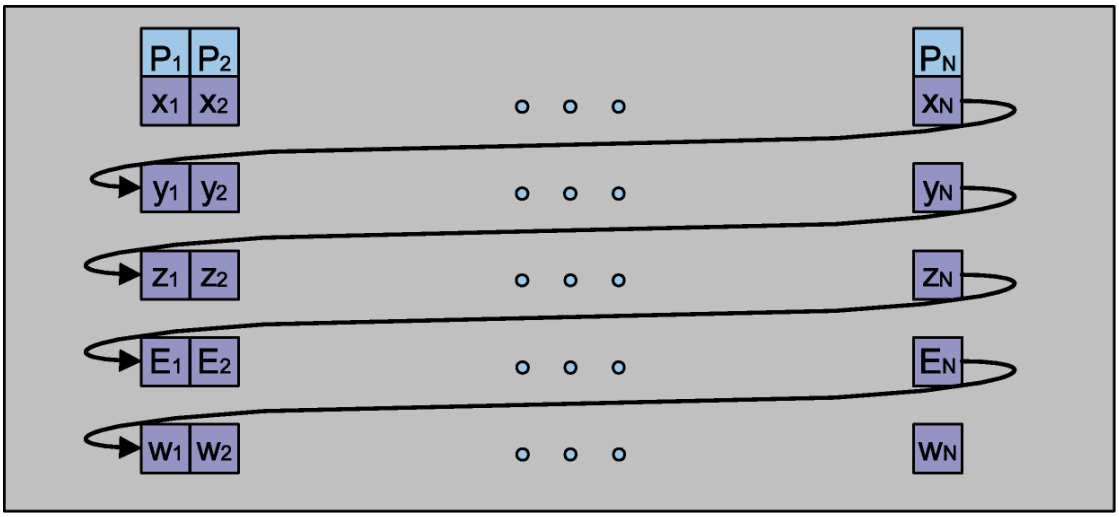

Figure. 2 – optimal data storage layout for GPU.

This approach is suitable for GPU computing, but it can slow down calculations on the CPU in case of hybrid parallelization. The size of the memory fragment containing the set of required electron parameters can exceed the CPU's cache size. This leads to frequent main memory accesses and decreases the calculation efficiency.

The suggested compromise for constructing the data layout is to group particles in blocks. The number of particles per block is determined by the condition of completely placing the data block in CPU cache. The inner structure of each block is chosen taking into account the connectivity requirement for GPU memory access. Such layout allows providing the data proximity for particles in each block and location of all parameters of one particle in CPU cache at the same time (Figure. 3).

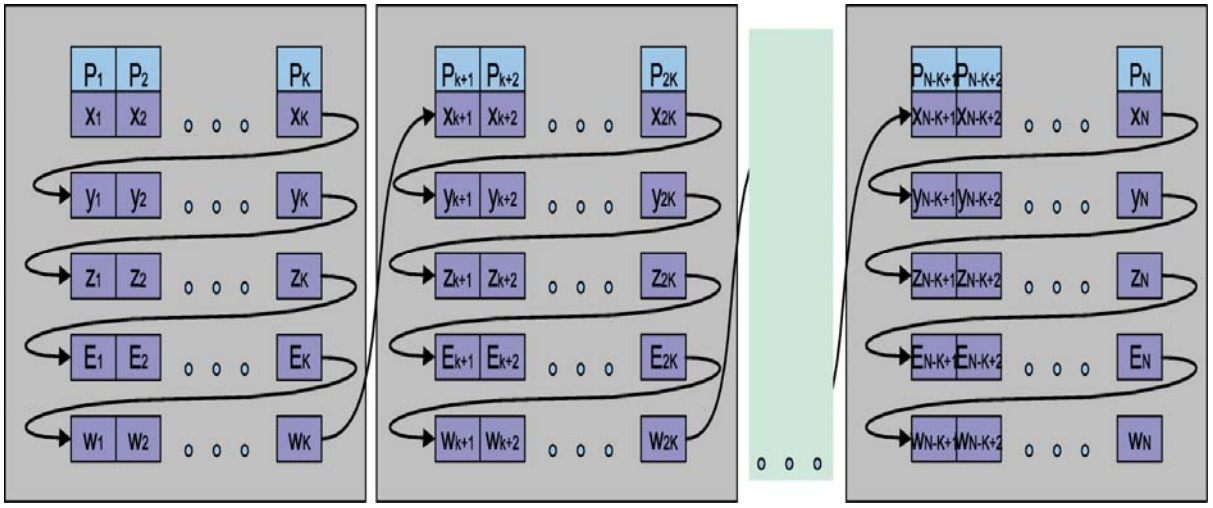

Figure. 3 – compromise data storage layout for hybrid computing.

Particle parameter data is accessed sequentially. The distributions of electron interaction characteristics (see 2nd part of the paper) are random access data. Therefore it is necessary, if possible, to place them in "fast" shared memory when computing on GPUs.

Third, conditional jumps are the most problematic operation for a GPU. In fact GPUs normally do not execute conditional jumps. Both branches of the algorithm are always executed and the result of one of them is ignored. This results in the necessity to develop an algorithm, that is as linear as possible, or to "straighten" the existing one.

For instance, the standard method of particle transport modeling implies checking for the particle reaching a detector for its registration. Due to the above mentioned reasons this detection technique can be inefficient, when one needs to check the particle hit on every segment of a particle trajectory.

Weight Monte Carlo algorithms[9] are much more efficient. In this case it is assumed that some "part" of a particle reaches the detector and the statistical weight of this part is equal to the probability of the event. After particle registration its trajectory is continued. In general the algorithm for hybrid computing is developed under the principle of the maximum of information value of particle trajectories. This means, that every event which can occur with some probability, but is not actually modeled, is assumed to really happen. The statistical weight of the event is equal to the specified probability. This method permits decreasing the

dispersion of results and reducing the number of conditional jumps of the algorithm. It is therefore especially important for computing on hybrid computers.

3. EXAMPLE OF MODELING USING HYBRID COMPUTERS

Let us consider the task of acquiring the spectral distribution of electrons in an x-ray tube target irradiated by an electron flux. A set of parallel planes are considered as detectors. The first of them is the target surface. The wanted functions are the energy distributions of electrons crossing each plane in the "forward" and "backward" directions.

The method for constructing the parallel algorithm for electron transport modeling is discussed below. The method takes into account the above mentioned peculiarities of hybrid parallelization.

The particle state is described by a set of phase space coordinates $\{E, \mathbf{\Omega}, \boldsymbol{r}, w\}$ - energy, motion direction, coordinates, and statistical weight of the particle. This data is placed according to the mentioned connectivity condition in global memory of the GPU for all K electrons of a block (see Figure. 3) before launching the computing kernels[7].

The distributions of electron interaction characteristics are tabulated and laid out as data arrays. These arrays are allocated in "fast" shared memory of the GPU.

The computing algorithm consists of the following basic steps:

- Sampling the path length between the current and next point of electron interaction according to the distribution density $f_s = \frac{1}{\mu}\exp\left(-\int_0^s \mu(x)\,dx\right)$ and calculating the coordinates of the next point;
- Sampling the type of interaction (elastic scattering, excitation, ionization, or bremsstrahlung);
- Sampling the electron parameters after the corresponding interaction process in accordance with MIC;
- Computing the contribution of the current electron trajectory segment to the registration functions;
- Continuation of the electron trajectory while its energy is higher than a given threshold.

The contribution of the current electron trajectory segment is calculated analytically. Namely, the energy cell of the detector is defined by the current energy of the electron, and the value of the contribution is calculated by multiplying the current electron weight by the probability of reaching of the detector un-changed.

Let $F(E)$ be the required value. We introduce the energy grid

$\{E_k\}_{k=0}^{K} = \{E_0 = E_{\min}, E_1, ..., E_K = E_{\max}\}$.

Let $\{P_n\}_{n=0}^{N}$ be a set of detector planes. Plane P_0 is the target surface.

The contribution of the current electron trajectory segment in the kth energy cell on the nth detector plane is calculated as $F_n^+(E_k) = w_e \omega_n \eta(E - E_{k-1}) \times \eta(E_k - E); k = 1,...,K$.

In this formula w_n is the current weight of the electron; ω_n is the probability of the electron reaching the nth plane: $\omega_n = \exp(-\mu(E) s_n)$, where μ, s_n are the macroscopic cross-section of electron interaction with the target material and the distance between the current point and the nth plane.

The contribution F_n^- for the electron penetrating deep into the target is identified as F_n^+. The contribution of the electron leaving the target is determined similarly.

The required spectral distribution $f_n^\pm$ is calculated by normalization of $F_n^\pm$:

$f_n^\pm = F_n^\pm \Big/ \int F_n^\pm dE$.

The algorithm in question is realized as a parallel code using MPI (Message Passing Interface) and CUDA libraries. MPI is used for transferring messages between CPUs. CUDA is used for computation on the GPUs. Calculations were carried out on the KIAM hybrid cluster K-100 (http://www.kiam.ru/MVS/resourses/).

The calculation scheme is depicted in figure 4.

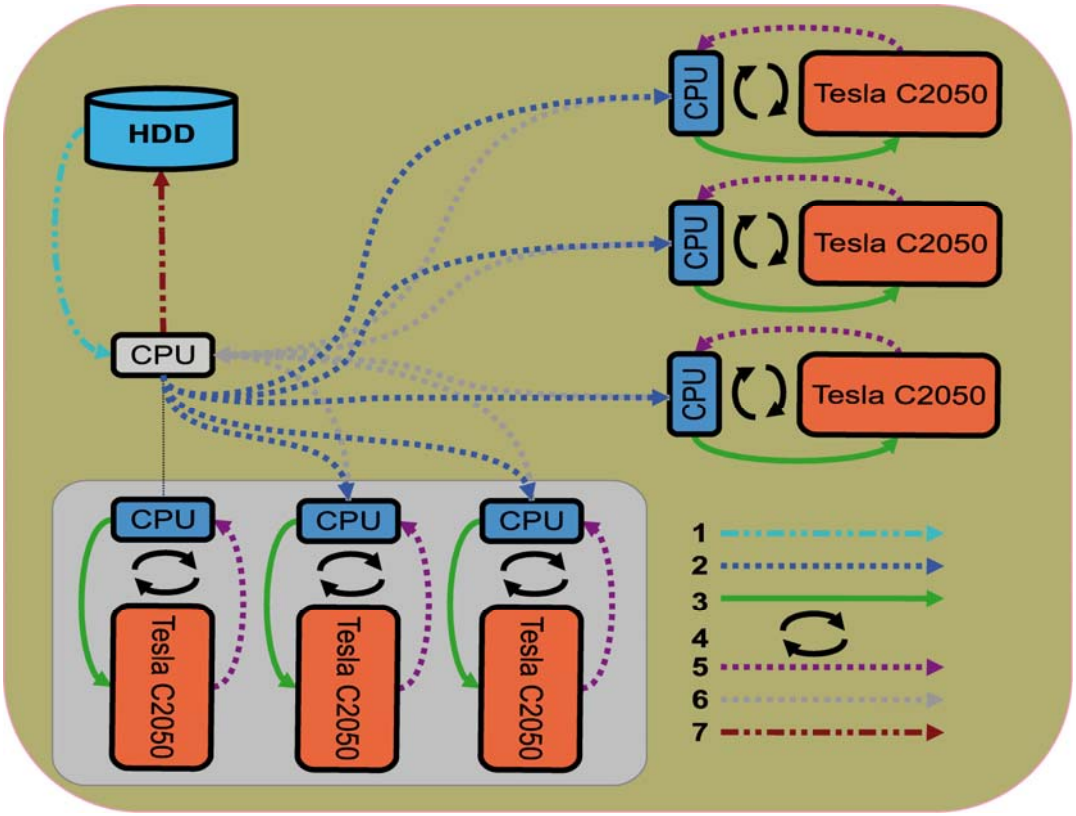

Figure. 4 –Scheme of carrying out the computation on K-100. 1 – loading and handling the parameters by host CPU; 2 – broadcasting the initial data to all CPUs; 3 – sending the data to GPU; 4 - iterative launching of computing kernels for modeling electron trajectories; 5 – returning the data to CPUs; 6 – collecting the results by host CPU; 7 – storing the results.

The developed code consists of three main components:

- The first component carries out initial data loading and handling, and storing the results;
- The second one organizes interactions between central processors of the cluster;
- The third one launches the computing kernels on the GPUs.

An investigation of the spectral distribution of electrons in the target was carried out utilizing the code in question. A 100 keV electron pencil beam normally incident on an aluminum target was used in the computational experiment.

Spectra of electrons crossing the detector planes in the original beam direction are presented in figure 5. The results for different depths in the target are marked by different colors (see Figure. 5). Spectra of electrons crossing the planes backwards are presented in Figure. 5.

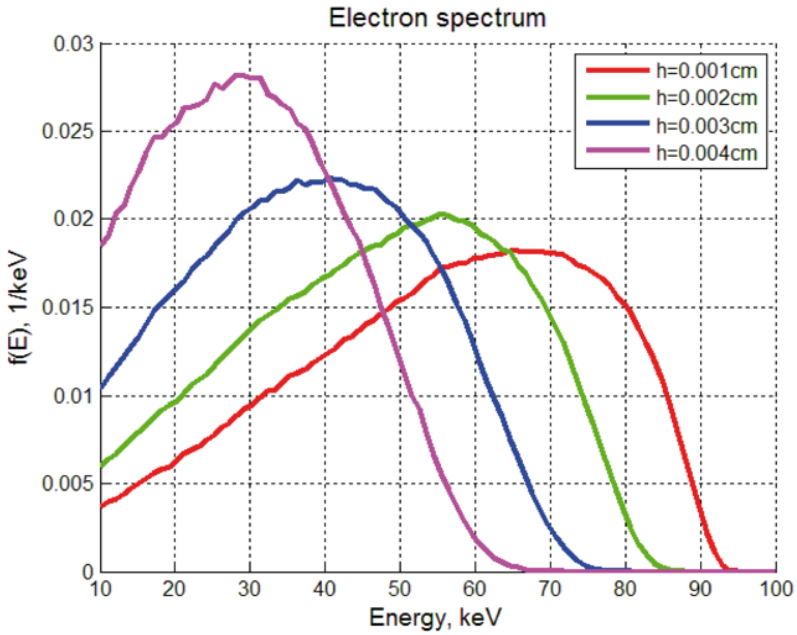

Figure. 5 Spectra of electrons moving backwards

Degradation of the spectra with increasing depth in the target is visible in the figures.

Second example is presented on the figure 6. Spectrum of electrons emitted from surface of an aluminum plate irradiated by X-ray radiation of 100 keV energy is pictured on the figure. In addition the comparison with MCNP computing is carried out (see fig. 6).

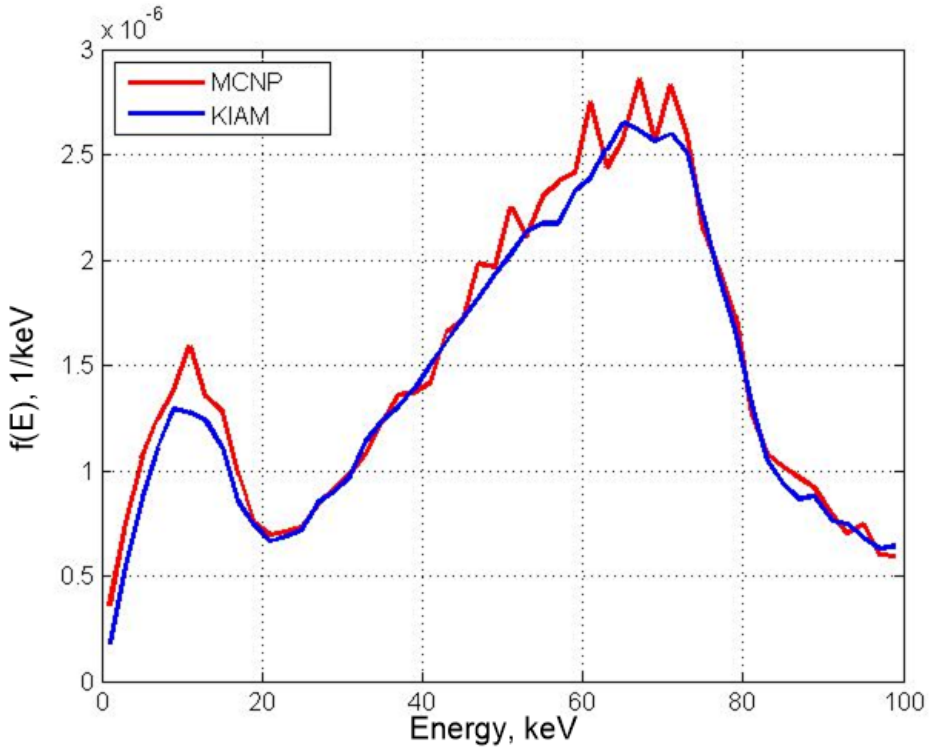

Figure. 6 Spectrum of electrons emitted from irradiated surface

In particular, the obtained results permit evaluating the gradient of spectrum degradation. This information is useful when estimating the effectiveness of the investigated target. The dis-cussed results are compared with the corresponding results obtained using MCNP[10]. The comparison has shown agreement of the results within the range of statistical error. The efficiency of modeling on K-100 is almost a hundred times higher than MCNP calculations. MCNP computing was carried out using a multiprocessor cluster with traditional linear architecture.

4 CONCLUSION

Analysis of the results of modeling electron transport on the hybrid cluster K-100 has shown the high efficiency of the developed algorithms on computers with hybrid architecture. Computing acceleration can reach a hundred times compared to computers based on a traditional linear architecture.

It should be noted that the used mathematical model of electron interaction with matter (MIC) is convenient for implementation on hybrid clusters. MIC does not use general approximations (multi-scattering theory, slow-down approximation, etc.) and therefore is preferable compared to widely used models. At the same time MIC is very hard to implement on linear multi-processor computing systems.

Keeping the above in mind, it can be asserted that the use of high performance hybrid supercomputers allows developing and using more detailed mathematical models for modeling electron transport in matter.

REFERENCES

[1] M. Zhukovskiy, S. Podoliako, R. Uskov. ”Model individualnykh soudareniy dlya opisaniya perenosa elektronov v veshestve”, *Matematicheskoye modelirovanie,* **23** (6), 147-160 (2011) (in Russian).

[2] *Monte Carlo simulation of x-ray transport in a GPU with CUDA.* (http://code.google.com/p/mcgpu/)

[3] M. Zhukovskiy, R. Uskov. *Ob ispolzovanii graficheskikh processorov v prilozheniyakh*. Preprint IPM RAN. 20 (2010) (in Russian).

[4] Goudsmit S. and J.L. Saunderson, “Multiple scattering of electrons”, *Phys. Rev.*, 57, 24-29 (1940).

[5] *National Nuclear Data Center*, (http://www.nndc.bnl.gov/)

[6] *Evaluated Nuclear Data File* (ENDF, http://www.nndc.bnl.gov/exfor/endf00.jsp).

[7] *NVIDIA CUDA Programming Guide, Version 2.3.1*, (2009).

[8] A. Boreskov, A. Kharlamov. *Osnovy raboty s CUDA tekhnologiey*. Moskva, DMK Press, (2010) (in Russian).

[9] G. Mikhailov. *Vesovye metody Monte-Karlo*. Novosibirsk. Siberians depart. of RAS, (2000) (in Russian).

[10] Briesmeister J.F. (ed.). *MCNP - A General Monte Car-lo N-Particle Transport Code*. LANL Report LA-13709-M, Los Alamos, (2000).

The results were presented at the thirteenth international seminar "Mathematical models & modeling in laser-plasma processes & advanced science technologies" (May 30 - June 6, 2015, Petrovac, Montenegro).