Наука и Образование
МГТУ им. Н.Э. Баумана
Сетевое научное издание
ISSN 1994-0408

Наука и Образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2016. № 06. С. 200–213.

DOI: **10.7463/0616.0842091**

Представлена в редакцию: 09.05.2016
Исправлена: 23.05.2016

УДК 004.056.55

Производительность поточных шифров, основанных на клеточных автоматах, при реализации на графических процессорах

Ключарёв П. Г.[1,*]

*pgkl@yandex.ru

[1]МГТУ им. Н.Э. Баумана, Москва, Россия

Статья посвящена тестированию производительности разработанных автором поточных шифров, основанных на обобщенных клеточных автоматах, при реализации на графических процессорах NVIDIA и AMD, с использованием OpenCL. Такие шифры имеют структуру, основанную на использовании четного числа обобщенных клеточных автоматов, графы которых являются графами Рамануджана. Производительность разработанной реализации составила до 6600 Мбит/с, что является хорошим результатом для шифров, рассчитанных на аппаратную реализацию, и сопоставимо с производительностью других современных поточных шифров.

Ключевые слова: клеточный автомат, поточный шифр, графический процессор

Введение

В настоящее время требования к защищенности информации, передаваемой в телекоммуникационных сетях, непрерывно повышаются вместе с требованиями к пропускной способности этих сетей. Это приводит к необходимости разработки высокопроизводительных алгоритмов шифрования. Разработанные автором способы построения высокопроизводительных поточных симметричных шифров, основанные на использовании обобщенных клеточных автоматов, позволяют получить алгоритмы шифрования, демонстрирующие крайне высокую производительность при аппаратной реализации. Вместе с тем, их реализация на обычных микропроцессорах не обладает высокой производительностью. Сам по себе этот факт вполне обычен – он говорит о сфере применимости этих шифров. Тем не менее, использование графических процессоров позволяет достичь приемлемого уровня производительности для программной реализации.

Данная статья является продолжением серии статей, посвященных исследованию различных аспектов построения и реализации криптографических алгоритмов, основанных на обобщенных клеточных автоматах, в том числе [4; 5; 6; 7; 8; 10] и др. Подобные алгоритмы могут найти применение в большом количестве различных задач, связанных с

информационной безопасностью, в том числе, с задачами, рассмотренными в работах [1; 2; 3]

Целью данной статьи является исследование возможности реализации рассматриваемых криптографических алгоритмов на графических процессорах и тестирование производительности такой реализации.

Графические процессоры

Термин «графический процессор» (англ. Graphic processor unit, GPU) впервые был использован компанией NVIDIA, которая таким образом акцентировала внимание на том, что графические ускорители, используемые до этого лишь для ускорения рендеринга трехмерной графики, стали подходить для решения широкого круга задач, с графикой не связанных. Подробные сведения о высокопроизводительных вычислениях на графических процессорах, в том числе историю развития этой области знаний, можно найти в целом ряде источников, в частности, в книгах [14; 15; 17; 18; 21].

Современные графические процессоры представляют собой высокопроизводительные вычислительные устройства обеспечивающие массовый параллелизм, а также обладающие высоким быстродействием (свыше одного терафлопса) и достаточно большим объемом оперативной памяти.

Для разработки программ для графических процессоров используются специальные API, такие как OpenCL и CUDA, являющиеся общеупотребительными. При их использовании программа состоит из двух частей: ядро (kernel), исполняемое на одном или нескольких устройствах, и хост-программа, исполняемая на основном устройстве (центральном процессоре). После того, как хост программа подает команду на исполнение ядра, определяется индексное пространство. Для каждой точки этого пространства происходит исполнение экземпляра ядра. Такой экземпляр называется рабочим элементом (work-item). Все рабочие элементы исполняют один и тот же код, однако между собой они отличаются значениями локальным и глобальным идентификаторами.

Рабочие элементы объединяются в рабочие группы (work-group). Каждая рабочая группа имеет свой уникальный идентификатор. Локальный идентификатор рабочего элемента назначается внутри рабочей группы таким образом, что сочетание локального идентификатора и идентификатора группы взаимно однозначно определяет рабочий элемент.

Размер рабочей группы ограничен и в стандарте OpenCl это ограничение определяется параметром устройства, определяемым производителем графического процессора. Для всех графических процессоров фирмы AMD этот параметр равен 256. В случае NVIDIA данный параметр зависит от версии CUDA. Для версий 1.1 - 1.3 он равен 512, а для более поздних – 1024. То есть максимально возможный размер графа при реализации на графических процессоров производства NVIDIA равен 1024.

Алгоритм поточного шифрования

Мы не ставим цель подробного описания реализуемого поточного шифра, поскольку он подробно описан в статье [4]. Представляя собой генератор гаммы, он состоит из двух обобщенных клеточных автоматов и линейного регистра сдвига с обратной связью (рис.1). Впервые эта структура предложена в [11]. Начальное заполнение автоматов и регистра зависит от ключа. На выход поступает поразрядная сумма по модулю 2 выходов двух обобщенных клеточных автоматов. Графами клеточных автоматов являются графы Любоцкого-Филипса-Сарнака, являющиеся графами Рамануджана [13; 16; 19].

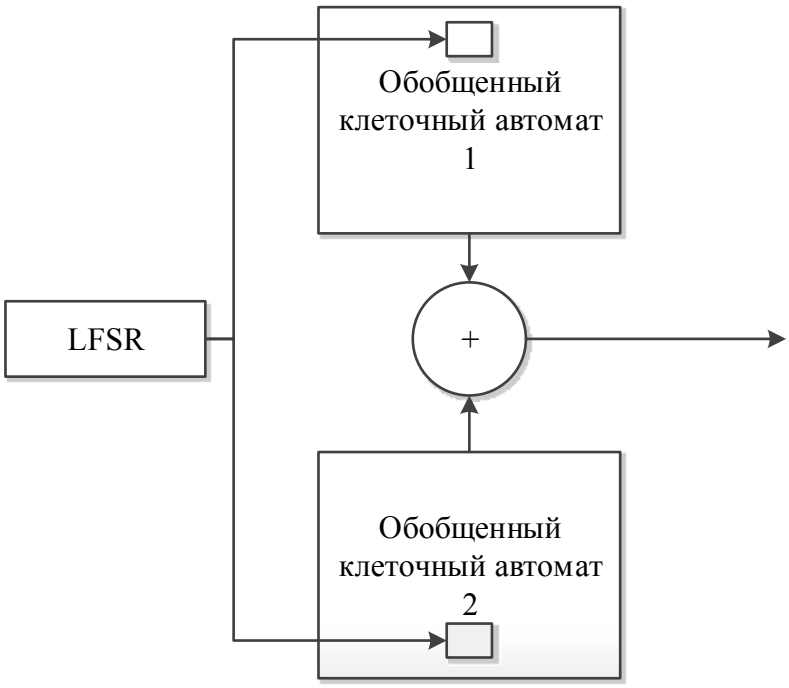

Рис. 1 – Структура поточного шифра

Здесь мы рассмотрим некоторое обобщение такой схемы, состоящее в том, что используется k таких генераторов, каждый из которых вырабатывает свой поток гаммы, а на выходе элементы этих потоков чередуются. Такое обобщение, как мы увидим в дальнейшем, позволяет более полно использовать возможности графического процессора.

Реализация

Реализация криптоалгоритмов была произведена с использованием OpenCL, как наиболее универсального и платформенно-независимого API. Написанная на языке C++ программа была построена таким образом, что у пользователя имелась возможность устанавливать различные параметры, в том числе, ключ шифрования, структуру графа, локальную функцию связи, различные константы и т.д. Часть параметров определены в коде и также могут быть легко изменены. В частности, могут быть изменены следующие параметры:

- размер рабочей группы. Этот параметр должен быть строго больше размера используемых графов, кроме того, он должен быть кратен 64. Рекомендуется использовать следующие значения для него: 256, 512 и 1024.
- Размер выхода клеточного автомата. Параметр должен быть кратен 8.
- Число клеточных автоматов, одновременно используемых. Это число равно удвоенному числу генераторов гаммы.

Заметим, что обобщенный клеточный автомат состоит из некоторого множества ячеек, для каждой из которых на каждом шаге требуется произвести однотипные вычисления. Этот процесс, очевидно, хорошо распараллеливается и может быть эффективно реализован на графическом процессоре. При этом необходимо обеспечить синхронизацию этих вычислений.

Как известно, в языке OpenCL существует два типа синхронизации:

- синхронизация рабочих элементов внутри рабочей группы;
- синхронизация команд, находящихся в очереди команд.

Синхронизация между рабочими элементами одной рабочей группы производится с помощью так называемых барьеров. Все рабочие элементы должны достичь барьера прежде, чем какой-либо из них продолжит выполнение программы за барьером.

Из определения клеточного автомата следует, что синхронизация должна выполняться на каждом шаге вычислений. При этом вызывать на каждый шаг автомата новую команду не эффективно, так как это относительно долгий процесс, поэтому весь граф целиком должен помещаться в одной рабочей группе, а каждой ячейки автомата должен соответствовать свой рабочий элемент.

Теперь заметим, что размер рабочей группы должен быть кратен размеру волнового фронта. Волновым фронтом называется группа рабочих элементов, исполнение которых происходит одновременно и выполняющих один и тот же код. Размер волнового фронта может отличаться для различных графических процессоров, но составляет, как правило, 32 или 64. Исходя из этого, имеет смысл использовать группы размера кратного 64. В данной работе мы будем использовать рабочие группы размера 256, 512 и 1024.

Для каждого обобщенного клеточного автомата, входящего в структуру поточного шифра используется своя рабочая группа. При этом комбинирование их выходных последовательностей тоже выполняется на графическом процессоре. А вот вычисление выходных последовательностей линейных регистров сдвига с обратными связями, как показали проведенные вычислительные эксперименты, более эффективно производить на CPU. Это по-видимому объясняется более высокой тактовой частотой центрального процессора и плохой способностью к распараллеливанию линейных регистров сдвига.

Тестирование производительности

Тестирование производительности проводилось на различных графических процессорах, при этом варьировались значения ряда параметров, которые приведены в таблице 1.

Таблица 1 – параметры тестирования шифрования

Размер рабочей группы	Размер выхода	Размер графа	Число клеточных автоматов
256	160	230; 242	2,4,8,16,32,64,128,256
512	320	462; 510	2,4,8,16,32,64,128
1024	640	1010; 1022	2,4,8,16,32,64,128

При проведении тестирования использовалось видеоадаптеры, основанные на следующих графических процессорах:

- NVIDIA GTX 650;
- NVIDIA GTX 770;
- AMD R9 280X.

Основные параметры этих видеоадаптеров приведены в таблице 2.

Таблица 2. Параметры графических процессоров

Параметр	Видеокарта		
	NVIDIA GTX 650	NVIDIA GTX 770	AMD R9 280X
Количество вычислительных элементов(compute units)	4	8	32
Тактовая частота, МГц	1033	1137	1000
Максимальный размер рабочей группы	1024	1024	256
Размер глобальной памяти, МБ	1024	2048	2048
Размер локальной памяти, КБ	48	48	32
Тип памяти	GDDR5	GDDR5	GDDR5
Год появления на рынке	2012	2013	2014

Приведем теперь непосредственные результаты тестирования. В таблице 3 и на рис. 2 приведены результаты для шифра, использующего обобщенные клеточные автоматы размера 230; 242.

Таблица 3. Результаты теста производительности для шифра, использующего обобщенные клеточные автоматы размера 230; 242

Количество потоков гаммы	Скорость, Мбит/сек		
	NVIDIA GTX650	NVIDIA GTX770	AMD R9 280X
1	271	298	121
2	535	634	277
4	941	1243	542
8	1474	2006	1038
16	1672	3274	2026
32	1787	3690	3954
64	1860	3911	6225
128	1898	4098	6607

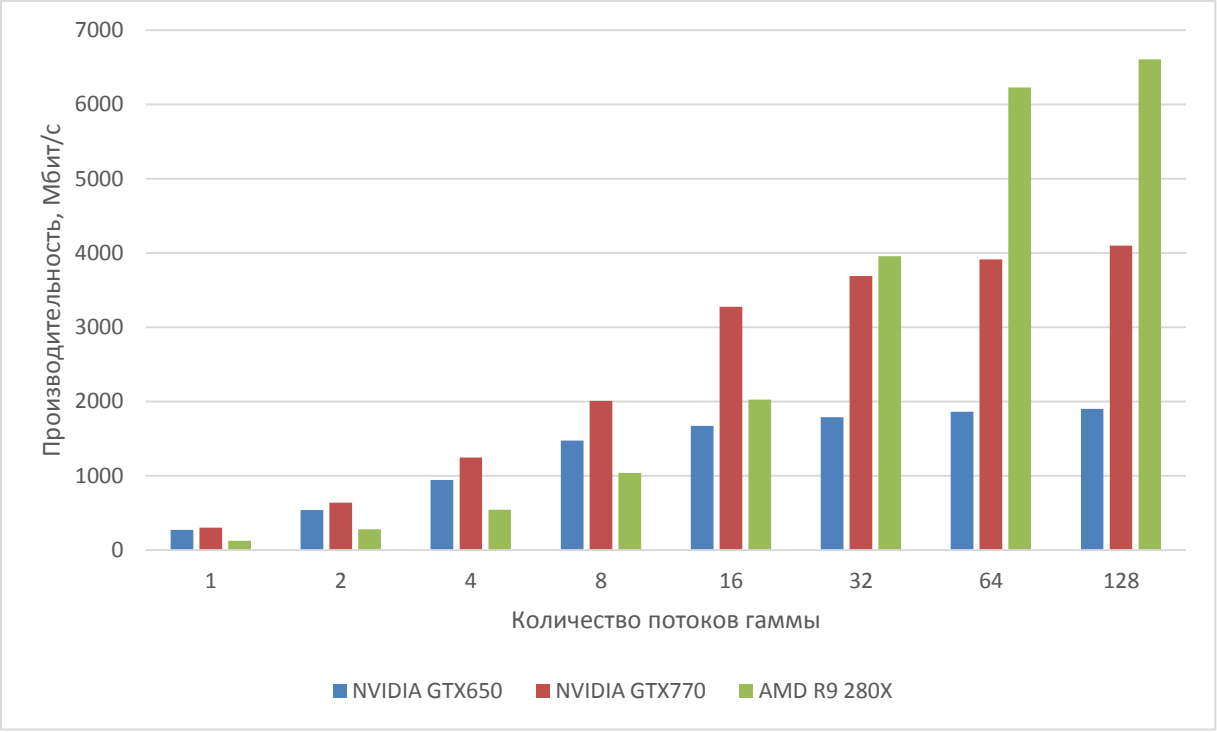

Рис. 2 – Результаты теста производительности для шифра, использующего обобщенные клеточные автоматы размера 230; 242

В таблице 4 и на рис. 3 приведены результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 462; 510. В этом тесте приняли участие только видеокарты от NVIDIA, что связано с тем, что максимальный размер рабочей группы для графических процессоров от AMD составляет не более 256.

Таблица 4. Результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 462; 510

Количество потоков гаммы	Скорость, Мбит/сек	
	NVIDIA GTX650	NVIDIA GTX770
1	392	427
2	764	842
4	1142	1656
8	1348	2434
16	1391	2816
32	1402	3079
64	1426	3154

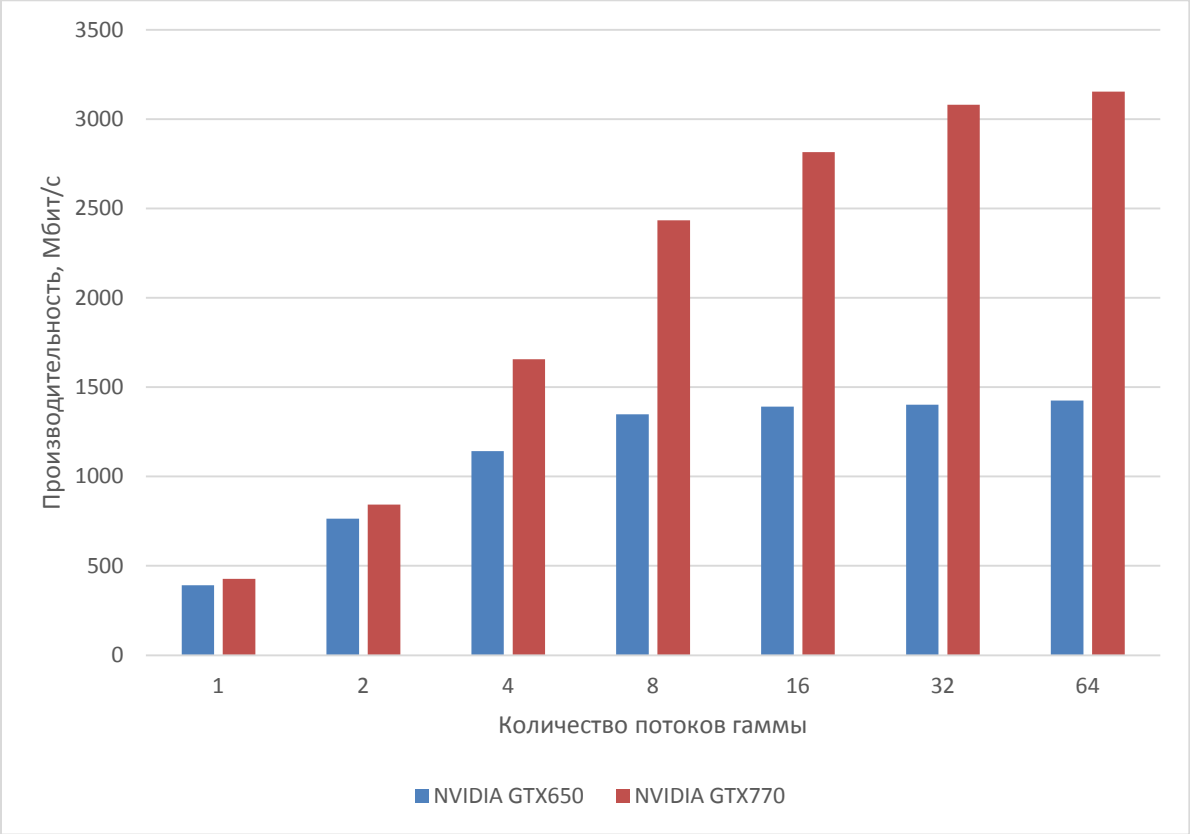

Рис. 3. Результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 462; 510

В таблице 5 и на рис. 4 приведены результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 1010, 1022. В этом тесте также приняли участие только видеоадаптеры с процессорами от NVIDIA.

Таблица 5. Результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 1010; 1022

Количество потоков гаммы	Скорость, Мбит/сек	
	NVIDIA GTX650	NVIDIA GTX770
1	474	527
2	924	1016
4	1117	1967
8	1128	2403
16	1178	2420
32	1200	2651
64	1210	2675

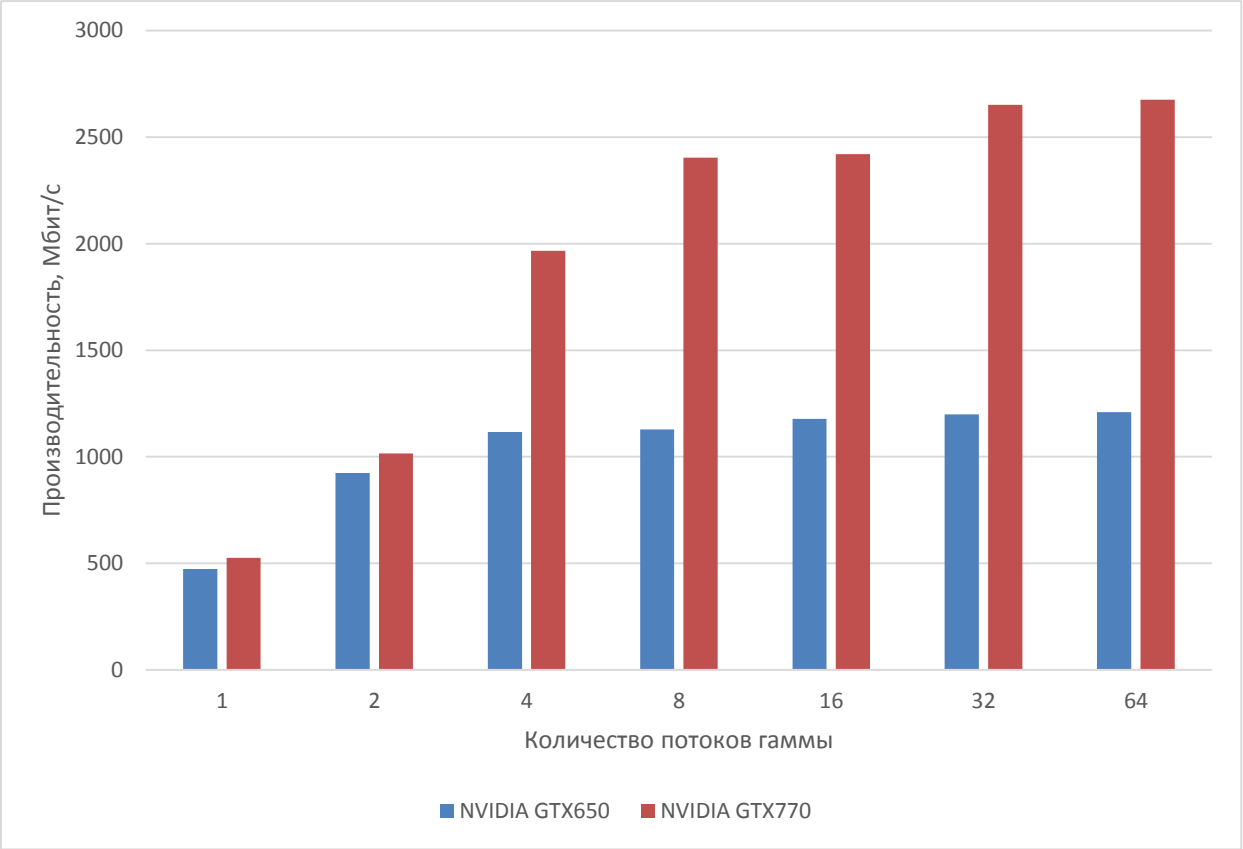

Рис. 4 - результаты тестов производительности для алгоритма шифрования, использующего обобщенные клеточные автоматы, размера 1010; 1022

Обсуждение результатов

Из результатов тестирования видно, что с увеличением количества потоков гаммы вычисления достаточно хорошо масштабируются – вычислительные возможности графических процессоров используются более полно. При этом производительность не достигает рекордов, полученных при аппаратной реализации того же алгоритма шифрования на программируемых логических интегральных схемах, которые достигают сотен Гбит/с (см.

[9]). В то же время, скорость работы современных поточных шифров на CPU не превышает нескольких Гбит/с (например, для шифра Rabbit [12], являющегося победителем конкурса eStream [20] и рассчитанного на программную реализацию, производительность составляет около 3.7 тактов на байт, т.е., для современных десктопных процессоров, 3 – 7 Гбит/с). Это совпадает по порядку с производительностью, полученной в данной статье (до 6.6 Гбит/с, в зависимости от параметров и графического процессора). Таким образом, при реализации данного шифра на графических процессорах, можно получить хороший уровень производительности.

Следует также отметить, что графы меньшего размера использовать выгоднее, как за счет большей производительности, так и за счет того, что в этом случае, шифр можно реализовать не только на графических ускорителях NVidia, но и AMD.

Заключение

Таким образом была продемонстрирована возможность эффективной программной реализации на графических процессорах поточных шифров, основанных на обобщенных клеточных автоматах.

Работа выполнена при поддержке РФФИ, проект №16-07-00542.

Список литературы

1. Быков А.Ю. Алгоритмы распределения ресурсов для защиты информации между объектами информационной системы на основе игровой модели и принципа равной защищенности объектов // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2015. № 9. С. 160-187. DOI: 10.7463/0915.0812283
2. Быков А.Ю., Артамонова А.Ю. Модификация метода вектора спада для оптимизационно-имитационного подхода к задачам проектирования систем защиты информации // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2015. № 1. С. 158-175. DOI: 10.7463/0115.0754845
3. Быков А.Ю., Панфилов Ф.А., Ховрина А.В. Алгоритм выбора классов защищенности для объектов распределенной информационной системы и размещения данных по объектам на основе приведения оптимизационной задачи к задаче теории игр с непротивоположными интересами // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2016. № 1. С. 90-107. DOI: 10.7463/0116.0830972
4. Ключарев П.Г. Клеточные автоматы, основанные на графах Рамануджана, в задачах генерации псевдослучайных последовательностей // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2011. № 10. С. 1-15. Режим доступа: http://www.technomag.edu.ru/doc/241308.html (Дата обращения: 05.06.16).
5. Ключарев П.Г. Криптографические хэш-функции, основанные на обобщённых клеточных автоматах // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2013. № 1. С. 161-172. DOI: 10.7463/0113.0534640

6. Ключарев П.Г. О вычислительной сложности некоторых задач на обобщенных клеточных автоматах // Безопасность информационных технологий. 2012. № 1. С. 30-32. Режим доступа: http://pvti.ru/data/file/bit/2012_1/part_4.pdf (дата обращения 01.03.2016).
7. Ключарев П.Г. О периоде обобщённых клеточных автоматов // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2012. № 2. С. 1-2. Режим доступа: http://technomag.edu.ru/doc/340943.html (дата обращения: 29.05.16).
8. Ключарев П.Г. Обеспечение криптографических свойств обобщённых клеточных автоматов // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2012. № 3. С. 1-8. Режим доступа: http://technomag.edu.ru/doc/358973.html (дата обращения: 01.06.16).
9. Ключарев П.Г. Производительность и эффективность аппаратной реализации поточных шифров, основанных на обобщенных клеточных автоматах // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2013. № 10. С. 299-314. DOI: 1013.0624722
10. Ключарёв П.Г. Реализация криптографических хэш-функций, основанных на обобщенных клеточных автоматах, на базе ПЛИС: производительность и эффективность // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2014. № 1. С. 214-223. DOI: 10.7463/0114.0675812
11. Сухинин Б.М. Разработка генераторов псевдослучайных двоичных последовательностей на основе клеточных автоматов // Наука и образование. МГТУ им. Н.Э. Баумана. Электрон. журн. 2010. № 9. С. 1-21. Режим доступа: http://technomag.edu.ru/doc/159714.html (дата обращения: 24.05.16).
12. Boesgaard M., Vesterager M., Pedersen T., Christiansen J., Scavenius O. Fast Software Encryption. Rabbit: A new high-performance stream cipher. Springer, 2003. Pp. 307-329. DOI: 10.1007/978-3-540-39887-5_23
13. Charles D.X., Goren E.Z., Lauter K.E. Families of Ramanujan graphs and quaternion algebras // Groups and symmetries: from Neolithic Scots to John McKay. 2009. Vol. 47. Pp. 53-63. Режим доступа https://www.researchgate.net/publication/228745797_Families_of_Ramanujan_graphs_and_quaternion_algebras (дата обращения: 01.06.16)
14. Eberly D.H. GPGPU Programming for Games and Science. Taylor & Francis, 2014. 441 p.
15. Gaster B., Howes L., Kaeli D.R., Mistry P., Schaa D. Heterogeneous Computing with OpenCL: Revised OpenCL 1.2 Edition. Elsevier Science, 2012. 291 p.
16. Hoory S., Linial N., Wigderson A. Expander graphs and their applications // Bulletin-American Mathematical Society. 2006. Vol. 43. № 4. Pp. 439-561. DOI: 10.1090/S0273-0979-06-01126-8
17. Kaeli D.R., Mistry P., Schaa D., Zhang D.P. Heterogeneous Computing with OpenCL 2.0. Elsevier Science, 2015. 330 p.

18. Kowalik J., Puźniakowski T. Using OpenCL: Programming Massively Parallel Computers. IOS Press, 2012. 295 p.
19. Krebs M., Shaheen A. Expander families and Cayley graphs. Oxford; New York : Oxford University Press, 2011. 288 p.
20. Robshaw M., Billet O. New Stream Cipher Designs: The ESTREAM Finalists. Springer, 2008. 300 p. DOI: 10.1007/978-3-540-68351-3
21. Scarpino M. OpenCL in Action: How to Accelerate Graphics and Computation. Manning, 2012. 458 p.

Science & Education of the Bauman MSTU
Electronic journal
ISSN 1994-0408

Science and Education of the Bauman MSTU, 2016, no. 06, pp. 200–213.

DOI: **10.7463/0616.0842091**

Received: 09.05.2016
Revised: 23.05.2016

Performance of Cellular Automata-based Stream Ciphers in GPU Implementation

P.G. Klyucharev[1,*]

*pgkl@yandex.ru

[1]Bauman Moscow State Technical University, Moscow, Russia

Keywords: cellular automata, stream cipher, GPU

Earlier the author had developed methods to build high-performance generalized cellular automata-based symmetric ciphers, which allow obtaining the encryption algorithms that show extremely high performance in hardware implementation. However, their implementation based on the conventional microprocessors lacks high performance. The mere fact is quite common - it shows a scope of applications for these ciphers. Nevertheless, the use of graphic processors enables achieving an appropriate performance for a software implementation.

The article is extension of a series of the articles, which study various aspects to construct and implement cryptographic algorithms based on the generalized cellular automata. The article is aimed at studying the capabilities to implement the GPU-based cryptographic algorithms under consideration.

Representing a key generator, the implemented encryption algorithm comprises 2k generalized cellular automata. The cellular automata graphs are Ramanujan's ones. The cells of produced *k* gamma streams alternate, thereby allowing the GPU capabilities to be better used.

To implement was used OpenCL, as the most universal and platform-independent API. The software written in C ++ was designed so that the user could set various parameters, including the encryption key, the graph structure, the local communication function, various constants, etc. To test were used a variety of graphics processors (NVIDIA GTX 650; NVIDIA GTX 770; AMD R9 280X).

Depending on operating conditions, and GPU used, a performance range is from 0.47 to 6.61 Gb / s, which is comparable to the performance of the countertypes.

Thus, the article has demonstrated that using the GPU makes it is possible to provide efficient software implementation of stream ciphers based on the generalized cellular automata.

This work was supported by the RFBR, the project №16-07-00542.

References

1. Bykov A.Yu. The Algorithms of Resource Distribution for Information Security Between Objects of an Information System Based on the Game Model and Principle of Equal Securi-

ty of Objects. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2015, no. 9, pp. 160-187. (In Russian). DOI: 10.7463/0915.0812283

2. Bykov A.Yu., Artamonova A.Yu. A Modified Recession Vector Method Based on the Optimization-Simulation Approach to Design Problems of Information Security Systems. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2015, no. 1, pp. 158-175. (In Russian). DOI: 10.7463/0115.0754845
3. Bykov A.Yu., Panfilov F.A., Khovrina A.V. The Algorithm to Select Security Classes for Objects in Distributed Information Systems and Place Data in the Objects Through Reducing the Optimization Problem to the Theory of Games with Non-conflicting Interests. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*. 2016, no. 1, pp. 90-107. (In Russian). DOI: 10.7463/0116.0830972
4. Klyucharev P.G. Cellular automations based on Ramanujan graphs in the field of the generation of pseudorandom sequences. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2011, no. 10, pp. 1-15. Available at: http://www.technomag.edu.ru/doc/241308.html (Accessed: 05.06.16). (In Russian).
5. Klyucharev P.G. Cryptographic hash functions based on generalized cellular automata. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2013, no. 1, pp. 161-172. (In Russian). DOI: 10.7463/0113.0534640
6. Klyucharev P.G. O On computational complexity of some generalized cellular automatons problems. *Bezopasnost Informatsionnykh Tekhnology*, 2012, no. 1, pp. 30-32. Available at: http://pvti.ru/data/file/bit/2012_1/part_4.pdf (accessed 01.03.2016). (In Russian).
7. Klyucharev P.G. About the period of generalized cellular automatons. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2012, no. 2, pp. 1-2. Available at: http://technomag.edu.ru/doc/340943.html (accessed: 29.05.16). (In Russian).
8. Klyucharev P.G. On cryptographic properties of generalized cellular automatons. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2012, no. 3, pp. 1-8. Available at: http://technomag.edu.ru/doc/358973.html (accessed: 01.06.16). (In Russian).
9. Klyucharev P.G. Performance and effectiveness of hardware implementation of stream ciphers based on generalized cellular automata. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2013, no. 10, pp. 299-314. (In Russian). DOI: 1013.0624722
10. Klyucharev P.G. FPGA implementation of general cellular automata based cryptographic hash functions: performance and effectiveness. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Education of the Bauman MSTU*, 2014, no. 1, pp. 214-223. (In Russian). DOI: 10.7463/0114.0675812
11. Sukhinin B.M. The Development of Pseudorandom Binary Sequences Generators Based on Cellular Automata. *Nauka i obrazovanie MGTU im. N.E. Baumana = Science and Educa-*

tion of the Bauman MSTU, 2010, no. 9, pp. 1-21. Available at: http://technomag.edu.ru/doc/159714.html (accessed: 24.05.16). (In Russian).

12. Boesgaard M., Vesterager M., Pedersen T., Christiansen J., Scavenius O. *Fast Software Encryption. Rabbit: A new high-performance stream cipher.* Springer, 2003. Pp. 307-329. DOI: 10.1007/978-3-540-39887-5_23
13. Charles D.X., Goren E.Z., Lauter K.E. Families of Ramanujan graphs and quaternion algebras. *Groups and symmetries: from Neolithic Scots to John McKay.* 2009. Vol. 47. Pp. 53-63. Available at: https://www.researchgate.net/publication/228745797_Families_of_Ramanujan_graphs_and_quaternion_algebras (accessed: 01.06.16)
14. Eberly D.H. *GPGPU Programming for Games and Science.* Taylor & Francis, 2014. 441 p.
15. Gaster B., Howes L., Kaeli D.R., Mistry P., Schaa D. *Heterogeneous Computing with OpenCL: Revised OpenCL 1.2 Edition*. Elsevier Science, 2012. 291 p.
16. Hoory S., Linial N., Wigderson A. Expander graphs and their applications. *Bulletin-American Mathematical Society*, 2006, vol. 43, no. 4, pp. 439-561. DOI: 10.1090/S0273-0979-06-01126-8
17. Kaeli D.R., Mistry P., Schaa D., Zhang D.P. *Heterogeneous Computing with OpenCL 2.0.* Elsevier Science, 2015. 330 p.
18. Kowalik J., Puźniakowski T. *Using OpenCL: Programming Massively Parallel Computers*. IOS Press, 2012. 295 p.
19. Krebs M., Shaheen A. *Expander families and Cayley graphs.* Oxford; New York: Oxford University Press, 2011. 288 p.
20. Robshaw M., Billet O. *New Stream Cipher Designs: The ESTREAM Finalists*. Springer, 2008. 300 p. DOI: 10.1007/978-3-540-68351-3
21. Scarpino M. OpenCL in Action: How to Accelerate Graphics and Computation. Manning, 2012. 458 p.