Реализация шифрования с использованием кватернионов на схемах программируемой логики с помощью Altera OpenCL SDK*

А.Е. Андреев[1], Е.И. Духнич[2], В.А. Егунов[1], Д.Н. Жариков[1], С.В. Ноздренков[1]

Волгоградский государственный технический университет[1],
Новороссийский государственный морской университет[2]

Рассматривается реализация шифрования с помощью гиперкомплексных чисел на базе аппаратурно-ориентированного алгоритма HW-QES на FPGA-сопроцессоре Altera с использованием Altera OpenCL SDK.

Ключевые слова: кватернион, Altera OpenCL SDK, FPGA, ПЛИС, HW-QES, ДЛП, CORDIC-подобные алгоритмы, OpenCL ядра, ГПСЧ

1. Введение

Современные вычислительные системы, от которых требуется высокая производительность, включая и встраиваемые решения, тяготеют к гибридизации. Применение различных вычислителей позволяет использовать их преимущества и частично компенсировать недостатки, одним из которых является высокая потребляемая мощность универсальных процессоров. В составе гибридных систем наряду с другими могут использоваться вычислители на базе программируемых логических интегральных схем (ПЛИС), в частности, FPGA. Они позволяют реализовывать аппаратно требуемые алгоритмы вычислений за счет возможности менять аппаратную конфигурацию микросхемы, задавая выполняемые логическими блоками функции и схемы соединения между ними в специальной внутренней памяти. Традиционно на подобной элементной базе эффективно реализуются алгоритмы потоковой обработки информации, допускающие конвейеризацию. К классу таких алгоритмов относятся и так называемые алгоритмы дискретных линейных преобразований (ДЛП), являющихся обобщением известного алгоритма CORDIC (их иногда называют CORDIC-подобными алгоритмами) [1-2]. Несмотря на то, что их сфера применения в основном – встраиваемые решения [2], некоторые алгоритмы ДЛП могут использоваться для создания решений на ПЛИС в составе гибридной системы, оснащенной мощными центральными процессорами (CPU). В таких системах решения на ПЛИС могут составлять конкуренцию даже графическим ускорителям, если учесть относительно низкие частоты работы и потребляемую мощность ПЛИС. В данной работе рассматривается реализация одного из таких алгоритмов – алгоритма шифрования на базе кватернионов, использующего ДЛП подход, для которого ранее (до 2014 года) были получены только приблизительные оценки эффективности реализации, но сама реализация не выполнялась.

Другим побудительным мотивом данной работы помимо реализации нового алгоритма на гибридной системе, включающей ПЛИС, являлось использование высокоуровневого средства разработки (high level synthesis - HLS) - OpenCL SDK от компании Altera. Это средство используется авторами с конца 2013 года для реализации или просто оценки характеристик аппаратной реализации различных алгоритмов на плате ускорителя DE5-Net компании Terasic [3]. И средство разработки, и ускорители предоставлены компанией Altera в рамках академической программы.

* при финансовой поддержке Российского фонда фундаментальных исследований (проекты №№ 15-07-06254, 15-01-04577, 16-07-00534

2. Алгоритм шифрования на базе кватернионов HW-QES, ориентированный на аппаратную реализацию

В основе схемы шифрования алгоритма HW-QES [4] лежит использование гиперкомплексного числа – кватерниона:

$$q = w + xi + yj + zk, \quad (1)$$

где w, x, y, z – вещественные числа, i, j, k — мнимые единицы со следующим свойством:

$$i^2 = j^2 = k^2 = ijk = -1. \quad (2)$$

Вектор данных B домножается слева на q, справа на q^{-1}, в результате мы получаем зашифрованный вектор B'. Это преобразование можно представить в виде матрично-векторного произведения $\Gamma(q)\,B$, где определяемая кватернионом q матрица $\Gamma(q)$ имеет вид :

$$\Gamma(q) = \frac{1}{|q|^2}\begin{pmatrix} |q|^2 - 2y^2 - 2z^2 & 2xy - 2zw & 2xz + 2yw \\ 2xy + 2zw & |q|^2 - 2x^2 - 2z^2 & 2yz - 2xw \\ 2xz - 2yw & 2yz + 2xw & |q|^2 - 2x^2 - 2y^2 \end{pmatrix} \quad (3)$$

Мы можем выбрать параметры кватерниона d следующим образом [4]:

$$w = 2^m + 1,\ x = 2^{m-i}\alpha,\ y = 2^{\frac{m+1}{2} - i}\beta,\ z = 2^{-i}\gamma, \quad (4)$$

где $m = 2k + 1$, $k = 0, 1, 2\ldots, K\text{-}1$,

$$|d|^2 = (2^m + 1)^2\left(1 + 2^{-2i}\right), \quad (5)$$

$\alpha, \beta, \gamma \in \{-1,1\}$, $i = 0, -1, -2\ldots,\ K > k \geq 0,\ |i| < I$

При подстановке данных значений в матрицу (3) мы получим новую матрицу вида:

$$\Gamma(d) = \frac{1}{|d|^2}\begin{bmatrix} (2^m+1)^2 + 2^{2m-2i} - 2^{m+1-2i} - 2^{-2i} & \alpha\beta 2^{3(m+1)/2-2i} + \gamma(2^{m-i+1} + 2^{-i+1}) & \alpha\gamma 2^{m-2i+1} + \beta(2^m+1)2^{(m+3)/2-i} \\ \alpha\beta 2^{3(m+1)/2-2i} - \gamma(2^{m-i+1} + 2^{-i+1}) & (2^m+1)^2 - 2^{2m-2i} + 2^{m+1-2i} - 2^{-2i} & \beta\gamma 2^{(m+3)/2-2i} - \alpha 2^{-i+m+1}(2^m+1) \\ \alpha\gamma 2^{m-2i+1} - \beta(2^m+1)2^{(m+3)/2-i} & \beta\gamma 2^{(m+3)/2-2i} + \alpha 2^{-i+m+1}(2^m+1) & (2^m+1)^2 - 2^{2m-2i} - 2^{m+1-2i} + 2^{-2i} \end{bmatrix} \quad (6)$$

(В формулах (4 - 6) i – это не мнимая единица i, обозначенная в формулах (1-2) курсивом, а целое число – параметр кватерниона).

Для выполнения шифрования выполняется l последовательных умножений исходного вектора на матрицы (6) по модулю выбранной степени двойки, на каждом шаге выбираются разные параметры кватерниона d. Подобный выбор компонентов матрицы (6) позволяет использовать в алгоритме только простые арифметические операции сложения и сдвига.

В работе [7] была предпринята попытка реализации описанной выше схемы шифрования в вычислительной системе, оснащенной устройством DE5-Net, с использованием технологии Altera OpenCL SDK. Рассматривались варианты реализации, при которых компоненты матриц преобразования (6) рассчитываются заранее на хосте (программой для CPU) и затем либо передаются в ядро, реализующее последовательность умножений на матрицы (6), либо хранятся в этом ядре в виде констант. В результате сама последовательность умножений на (6) выполня-

лась ядром относительно эффективно, особенно при большом числе итераций и матриц (более 15), в частности, наблюдалось ускорение вычислений до 1,7 раз по сравнению с GPU TESLA K20 для больших блоков данных (в основном за счет конвейеризации). По сравнению с выполнением преобразования на CPU ускорение достигало 30 раз.

Однако этот вариант реализации не полностью соответствует алгоритму HW-QES [4]. Во-первых, в работе [7] не указано явно, что умножение на матрицы (6) в алгоритме выполняется по модулю степени двойки, в частности, по модулю 256, если элементами шифруемых векторов являются байты, хотя реализация выполнялась именно таким образом. Во-вторых, сам принцип обработки потока данных (или файлов большого объема) с помощью одних и тех же предварительно рассчитанных матриц (6) приведет к корреляции между разными блоками исходных и зашифрованных потоков и тем самым снижает криптостойкость схемы.

В оригинальном алгоритме HW-QES предусматривается выбор параметров кватерниона и соответствующей ему матрицы (6) с помощью псевдослучайной процедуры (генератора псевдослучайных чисел - ГПСЧ) непосредственно перед каждым умножением вектора на матрицу (6) на каждой итерации шифрования, что делает данную схему более устойчивой к атакам. Реализация этого варианта на реконфигурируемой системе с ПЛИС-ускорителем и является предметом данного исследования. Кроме того целью работы является определение возможности применения технологии Altera OpenCL SDK при выполнении подобного прототипирования и оценка характеристик получаемого решения.

3. Программная реализация

В рассматриваемом алгоритме HW-QES с точки зрения реализации можно выделить ряд шагов:

1) генерация псевдослучайной последовательности с секретным начальным значением;
2) выбор параметров кватерниона (4) α, β, γ, i, k на основе случайного значения от ГПСЧ;
3) построение матрицы вращения (6) на основе выбранного кватерниона;
4) умножение вектора на матрицу вращения, сконструированную на 3 шаге, по модулю 256;
5) шаги 2 – 4 повторяются l раз для текущего обрабатываемого входного вектора из 3 элементов;
6) по завершении l итераций вектор домножается на коэффициент $K_i = \frac{1}{|d|^2}$ в (6);
7) шаги 2-7 повторяются для каждого 3-элементного вектора из входной последовательности.

В данном описании пункт 1 предполагает предварительное формирование псевдослучайной последовательности, что упрощает реализацию, в том числе – на ПЛИС (а особенно – на GPU), но требует значительных дополнительных затрат памяти. Возможен вариант, при котором ГПСЧ работает последовательно, но достаточно быстро, чтобы обеспечить по запросу все модули, параллельно либо конвейерно реализующие шаги 2-7 для разных входных векторов. Этот вариант более перспективен для реализации на ПЛИС, но на данном этапе работы не рассматривался.

Что касается выполнения шагов 2-7, то помимо конвейерной реализации, в данном случае, как следует из описания HW-QES, возможна параллельная обработка различных векторов из входной последовательности, поскольку параметры преобразования каждого из них в явном виде независимы и зависят только от членов ряда псевдослучайных чисел.

Далее нужно решить, какие из перечисленных шагов следует выполнять в кристалле ПЛИС, то есть в данном случае – в коде ядра OpenCL [5]. С точки зрения упрощения реализации на данном этапе было решено реализовать ГПСЧ на хосте и передавать сгенерированную последовательность в ядро.

Для реализации пункта 6 – масштабирования результата с учетом выполнения операций по модулю 256 применяется подход, основанный на бинарном возведении в степень по модулю, требующий всего 7 шагов и хорошо конвейеризуемый в ПЛИС [6].

4. Результаты экспериментов

В таблице 1 приведены результаты выполнения предложенной реализации схемы HW-QES в режиме шифрования на CPU Intel Core i7 920 2.7 ГГц (на одном ядре, язык C), на GPU NVidia GeForce GTX 650 ti и на устройстве DE5-Net с ПЛИС Altera Stratix V (OpenCL C) для l = 14 итераций при обработке блоков данных разного объема.

Из таблицы видно, что без учета времени генерации ПСЧ в среднем реализация HW-QES на ПЛИС с помощью OpenCL позволяет выполнять шифрование в 60 - 64 раза быстрее, чем одно ядро относительного быстрого центрального процессора и в 2 - 3 раза медленнее мощного GPU ускорителя. Устройство на ПЛИС при этом потребляет примерно в 1,5 – 2 раза меньше мощности по сравнению с GPU и примерно в 2 раза меньше – по сравнению с CPU.

При реализации на ПЛИС также и ГПСЧ реализация на ПЛИС может оказаться предпочтительнее, так как реализация ГПСЧ на одном ядре GPU, скорее всего, будет заметно уступать реализации на ПЛИС, а параллельная реализация на GPU осложнена необходимостью получать одну и ту же последовательность при шифровании и дешифровании. При увеличении количества итераций шифрования также можно ожидать роста производительности ПЛИС - реализации по сравнению с реализацией на GPU, что показано в [7].

Таблица 1. Время выполнения шифрования

Размер блока данных, байт	**Время решения на CPU Core i7, с**	**Время решения на GPU, OpenCL, с**	**Время решения на FPGA Stratix V, OpenCL, с**
10 000 000	5,703	0,027	0,078
20 000 000	10,625	0,053	0,167
30 000 000	15,695	0,076	0,248
40 000 000	20,961	0,191	0,332
50 000 000	26,192	1,127	0,406
60 000 000	31,358	0,149	0,5
70 000 000	36,671	0,278	0,578
80 000 000	41,905	0,249	0,656
90 000 000	47,061	0,319	0,749
100 000 000	52,389	0,289	0,828

Также отметим, что в таблице 1 приведены результаты при использовании в ПЛИС однопоточного конвейера, несмотря на то, что решение занимает менее 25% ресурсов микросхемы ПЛИС. На данном этапе попытки разместить в ПЛИС более одного конвейера с использованием штатных средств Altera OpenCL SDK (в частности, атрибутов num_compute_units() или num_simd_work_items()) пока не привели к росту производительности. Также в работе не рассмотрен рекомендуемый Altera в новых версиях OpenCL SDK вариант реализации последовательных конвейеризуемых алгоритмов в виде single work-item ядра, что также может дать дополнительный прирост производительности [6].

Время решения на FPGA по сравнению с CPU/GPU может сократиться при использовании FPGA ядра в составе серверного процессора, что минимизирует расходы на пересылку данных, это предполагается реализовать уже в ближайших версиях Intel Xeon Skylake.

Предложенный подход к реализации можно применить и для построения схемы шифрования на базе октонионов – HW-OES, также рассмотренной в [4]. При этом возможности конвейеризации данного решения окажутся очень кстати, поскольку в отличие от схемы HW-QES схема на базе октонионов использует последовательный характер формирования следующих октонионов на базе предыдущих, как и в схеме M-QES. Применение схемы с распараллеливанием, а значит, и многопоточных вычислений на GPU, здесь будет затруднено, если вообще возможно.

5. Заключение

В работе рассмотрена реализация прототипа, выполняющего шифрацию по алгоритму HW-QES на ПЛИС-сопроцессоре DE5-Net, оснащенном FPGA Altera Stratix V с использованием средства высокоуровневой разработки Altera OpenCL SDK. Полученные предварительные результаты показывают возможность ускорения вычислений в десятки раз по сравнению с одним ядром центрального процессора (и в 2-3 раза меньшую производительность по сравнению с графическим ускорителем) при меньшем энергопотреблении. Полученные результаты могут быть улучшены при дальнейшей оптимизации кода ядер на Altera OpenCL SDK.

Литература

1. Pramod K. Meher, Javier Valls, Tso-Bing Juang, K. Sridharan, Koushik Maharatna. 50 Years of CORDIC: Algorithms, Architectures, and Applications // IEEE Transactions on Circuits and Systems I: Regular Papers. 2009. Vol. 56, No. 9, P. 1893-1907.
2. Doukhnitch E., Salamah M., Andreev A. Effective processor architecture for matrix decomposition // Arabian Journal for Science and Engineering. 2014. Vol. 39, No. 3. P. 1797-1804.
3. Andreev A., Doukhnitch E., Egunov V., Zharikov D., Shapovalov O., Artuh S. Evaluation of Hardware Implementations of CORDIC-Like Algorithms in FPGA Using OpenCL Kernels // Knowledge-Based Software Engineering (JCKBSE 2014) : Proceedings of 11th Joint Conference, (Volgograd, Russia, September 17-20, 2014) / ed. by A. Kravets, M. Shcherbakov, M. Kultsova, Tadashi Iijima, Volgograd State Technical University, Springer International Publishing, 2014. P. 228-242. (Series: Communications in Computer and Information Science. 2014. Vol. 466).
4. Doukhnitch E., Chefranov A., Mahmoud A. Encryption Schemes with Hyper-Complex Number Systems and Their Hardware-Oriented Implementation. Chapter 5 in Theory and Practice of Cryptography Solutions for Secure Information Systems. IGI Global, 2013. P. 110-132. DOI:10.4018/978-1-4666-4030-6.ch005
5. The open standard for parallel programming of heterogeneous systems.URL: http://www.khronos.org/opencl (дата обращения: 01.12.2015)
6. Altera SDK for OpenCL Optimization Guide. URL:http://www.altera.com/literature/hb/opencl-sdk/aocl_optimization_guide.pdf (дата обращения: 01.12.2015)
7. Андреев А.Е., Красников А.А., Коржова С.А. Применение OpenCL для шифрования изображений на базе кватернионов в неоднородных вычислительных системах // Известия ВолгГТУ. Серия "Актуальные проблемы управления, вычислительной техники и информатики в технических системах". Вып. 21: межвуз. сб. науч. ст. / ВолгГТУ. Волгоград, 2014. № 12 (139). С. 129-135.

Implementing Encryption with Quaternions on the basis of Programmable Logic using Altera OpenCL SDK*

Andreev A. E.[1], E.I. Doukhnitch[2], V.A. Egunov[1], D. N. Zharikov[1], S.V. Nozdrenkov[1]
Volgograd state technical university[1], Novorossisk state maritime university[2]

The implementation of encryption by using hypercomplex numbers based on a hardware-oriented algorithm on FPGA Altera coprocessor with Altera OpenCL SDK is considered.

Keywords: quaternion, Altera OpenCL SDK, FPGA, HW-QES, DLT, CORDIC-like algorithms, OpenCL kernel, RNG.

References

1. Pramod K. Meher, Javier Valls, Tso-Bing Juang, K. Sridharan, Koushik Maharatna. 50 Years of CORDIC: Algorithms, Architectures, and Applications // IEEE Transactions on Circuits and Systems I: Regular Papers. 2009. Vol. 56, No. 9, P. 1893-1907.
2. Doukhnitch E., Salamah M., Andreev A. Effective processor architecture for matrix decomposition // Arabian Journal for Science and Engineering. 2014. Vol. 39, No. 3. P. 1797-1804.
3. Andreev A., Doukhnitch E., Egunov V., Zharikov D., Shapovalov O., Artuh S. Evaluation of Hardware Implementations of CORDIC-Like Algorithms in FPGA Using OpenCL Kernels. Knowledge-Based Software Engineering (JCKBSE 2014) : Proceedings of 11th Joint Conference, (Volgograd, Russia, September 17-20, 2014) / ed. by A. Kravets, M. Shcherbakov, M. Kultsova, Tadashi Iijima, Volgograd State Technical University, Springer International Publishing, 2014. P. 228-242. (Series: Communications in Computer and Information Science. 2014. Vol. 466).
4. Doukhnitch E., Chefranov A., Mahmoud A. Encryption Schemes with Hyper-Complex Number Systems and Their Hardware-Oriented Implementation. Chapter 5 in Theory and Practice of Cryptography Solutions for Secure Information Systems. IGI Global, 2013. P. 110-132. DOI:10.4018/978-1-4666-4030-6.ch005
5. The open standard for parallel programming of heterogeneous systems.URL: http://www.khronos.org/opencl (accessed: 01.12.2015)
6. Altera SDK for OpenCL Optimization Guide. URL:http://www.altera.com/literature/hb/opencl-sdk/aocl_optimization_guide.pdf (accessed: 01.12.2015)
7. Andreev A.E., Krasnikov A.A., Korzhova S.A. Primenenye OpenCL dlya shifrovanya izobrazheniy na baze quaternionov v neodnorodnykh vychislitelnyh systemah. [The use of OpenCL for Encryption of Images Based on Quaternions in heterogeneous computing systems]. Izvestya VolgGTU. Seria "Actualnye problemy upravlenya, vychislitelnoi tekhniki i informatiki v tekhnicheskyh systemah" [Bulletin of VSTU. Series "Actual problems of control, computer science and Informatics in technical systems"]. 2014, No 12 (139). P. 129-135.

* With the financial support of RFBR (projects ## 15-07-06254, 15-01-04577, 16-07-00534)