Proc. Int. Conf. SCALNET'2004. – Bulletin of Kremenchuk State Polytechn. Univ. Trans. SCALNET'2004. –2004. –p.121–123.
УДК 681.322

КОНФИГУРИРУЕМЫЙ КРИПТОГРАФИЧЕСКИЙ ПРОЦЕССОР

Василенко В.И., Лепеха В.Л., Сергиенко А.М.
Национальный технический университет украины "КПИ"

Введение. Алгоритмы криптографии относятся к числу алгоритмов, требющих большого объема вычислений. Во многих случаях микропроцессоры общего назначения вычисляют эти алгоритмы параллельно с выполнением основных задач. Но в этом случае существенно сокращается эффективная производительность такого микропроцессора. Кроме того, такие вычисления сопровождаются с дополнительным энергопотреблением, что критично для портативных устройств. Поэтому за многие годы в мире потрачено много усилий для того, чтобы реализовать алгоритмы криптографии в спецпроцессорах. Для этого обычно используют заказные СБИС, сигнальные микропроцессоры и программируемые логические интегральные схемы (ПЛИС) [1],[2].

Многими авторами проверено, например, в [2], [3], [4],[5], что ПЛИС являются лучшей программируемой вычислительной средой для скоростной реализации криптографических алгоритмов. В работе [5] показано, что в криптографический процессор на основе ПЛИС выполняет такой сложный алгоритм, как RSA более чем в десять раз быстрее, чем при его программной реализации.

Криптографические алгоритмы обычно вычисляются в некотором программном окружении. Такое окружение выполняет необходимые форматирование данных, их обмен, сетевые протоколы, а также системные функции и т.п. Но в существующих спецпроцессорах эти функции и алгоритм криптографии выполняются в отдельных микросхемах. Такие системы имеют ряд недостатков, связанных с ограниченностью скорости передачи данных по межпроцессорному интерфейсу, с дополнительными аппаратными затратами, вызванными необходимостью подключения различных периферийных устройств и т.п.

В работе [6] был предложен криптографический процессор, состоящий из аппаратного ускорителя для реализации алгоритма RSA и ядра микропроцессора, размещенных в одном корпусе ПЛИС Altera Apex. Ядро этого микропроцессора, аппаратно реализованное на ПЛИС, имеет архитектуру 32-разрядного микропроцессора ARM, а ускоритель реализует алгоритм модульного умножения Монтгомери. В результате применения такой архитектуры, протокол удаленной аутентификации пароля (Secure Remote Password authentication protocol -SRP), который сводится, в основном, к повторению модульного умножения и возведения в степень, вычисляется в 20 раз быстрее, чем при его программной реализации в микропроцессоре ARM.

Недостаток такого криптопоцессора, построенного на базе готового модуля микропроцессора, состоит в высокой трудоемкости переноса проекта с одной серии ПЛИС на другую. Например, упомянутый выше процессор невозможно исполнить в современных ПЛИС типа Altera Stratix или Xilinx Virtex2 без серьезной переделки проекта. Кроме того, обычно ПЛИС со встроенным ядром микропроцессора имеет увеличенную цену.

Цель работы. Авторами предлагается выполнять указанные программы окружения непосредственно на ПЛИС, где реализуется алгоритм криптографии в гибком ядре микропроцессора, описанном на языке VHDL. Тогда криптографический процессор имеет все преимущества аналогичных процессоров, но также может быть реализован в любой серии ПЛИС, становится независимым от платформы процессора, к которому он подключается и может быть автономным. Для проверки этого подхода была разработана система из RISC - микропроцессора и аппаратного ускорителя алгоритмов криптографии, которая конфигурируется в современных ПЛИС.

Архитектура ядра микропроцессора ARM. Микропроцессоры с архитектурой ARM – наиболее распространенные 32-разрядные RISC – микропроцессоры, которые используются в виде готовых вычислительных модулей (IP core) в современных системах на кристалле (СНК). Они широко применяются в мобильных телефонах, средствах обработки сигналов, вокодерах и для видео обработки. Ядро ARM также широко используется в системах защиты информации, благодаря его высокому отношению производительность-стоимость, надежности матобеспечения и низкому энергопотреблению.

В стандартной архитектуре ARM v.4 выполняется система 32-разрядных команд ARM – процессора и набор 16-разрядных упрощенных команд, называемых Thumb - командами. Ядро микропроцессора содержит 32 четырехбайтных общих регистра, сдвигатель, АЛУ, умножитель, блок выборки данных и буфер записи. Микропроцессор кроме ядра содержит также кеш ОЗУ данных, кеш ОЗУ команд, сопроцессор с плавающей запятой, интерфейс AMBA и отладочный интерфейс. Такая архитектура обеспечивает как высокую производительность, так и ведение виртуальной памяти, включая механизмы защиты памяти. Платформа микропроцессора ARM обеспечивает исполнение таких широко распространенных операционных систем, как EPOC, Linux, Windows CE.

Следующие особенности отличают архитектуру ARM от других RISC - архитектур. В каждом такте программно доступны 16 регистров. Остальные регистры имеют адреса – синонимы и сгруппированы в банки регистров. Эти банки могут переключаться, в момент, когда ядро микропроцессора изменяет свой

режим. Этот механизм используется для быстрой отработки прерываний или перехода в привилегированный режим. Одним из специальных режимов является прерывание - ловушка (trap) неопределенной команды. Это такая команда, которая не реализована аппаратно в данной версии ядра микропроцессора, но могут быть вычислены по подпрограмме.

Ядро микропроцессора АЯМ. Гибкое ядро 32-разрядного микропроцессора, названное АЯМ за свою похожесть с ядром ARM - микропроцессора (АЯМ - ARM – подобное Ядро Микропроцессора) было разработано на кафедре вычислительной техники Национального технического университета Украины “КПИ”. Это ядро выполняет систему команд архитектуры ARM v4. Целью разработки ядра АЯМ были получение гибкого ядра современного эффективного микропроцессора, которое может быть использовано в современных СНК, конфигурируемых в ПЛИС. Другой целью было использование этого ядра в научных исследованиях в области высокоуровневого синтеза и совместного аппаратно-программного проектирования.

Ядро АЯМ имеет следующие особенности. Оно может исполнять систему команд ARM. Но не все команды этой системы исполняются непосредственно ядром. Команды с плавающей запятой и некоторые другие вычисляются по подпрограммам в режиме неопределенной команды. Также не реализованы команды из системы Thumb.

Структура ядра АЯМ имеет много отличий от ядра ARM. Временные диаграммы исполнения многих команд отличаются. Некоторые команды, например, умножение, вычисляются быстрее.

Кеш ОЗУ в ядре не используются по следующим причинам. Когда ядро конфигурируется в ПЛИС, то тактовая частота процессора не может достигать высоких значений и равна 30-50 МГц. Большинство современных ОЗУ функционируют с такой частотой, не требуя буферной памяти. В этом случае кэш - ОЗУ функционировало бы неэффективно. Поэтому при доступе к медленной внешней памяти и портам ввода-вывода процессор АЯМ вводит такты ожидания до окончания доступа. Малое быстродействие внешней памяти может быть компенсировано оверлейной структурой данных и программ при использовании быстрых внутренних блоков двухпортовой памяти и механизма ПДП. Кроме того, отказ от кэш – ОЗУ обеспечил снижение аппаратных затрат и тактового интервала в полтора раза при конфигурировании ядра в ПЛИС. Аналогичную структуру имеют ядра микропроцессоров FS8051 и RISC_ST, также разработанных в НТУУ"КПИ" [7]-[10].

В ядре АЯМ банки регистров не переключаются при смене контекста и изменении режима. В этом случае содержимое банков переписывается на место соответствующих теневых регистров. Эти регистры реализованы на основе конвейерных регистров SRL16, входящих в состав ПЛИС Xilinx Virtex. Благодаря такой структуре, сеть мультиплексоров данных становится значительно проще и в результате, повышается тактовая частота ядра.

Так как ядро микропроцессора предназначено для исполнения программы одного пользователя, которая хранится в ПЗУ, то механизм виртуальной памяти не реализовался. Все пространство адресов – это физическое 32-разрядное пространство. Адреса, отличающиеся на младшую единицу, соответствуют соседним байтам.

Спроектированное ядро микропроцессора имеет минимизированные аппаратурные затраты и сравнительно высокую тактовую частоту. Результаты конфигурирования ядра в ПЛИС Xilinx показаны в таблице 1.

Таблица 1.
Результаты конфигурирования ядра АЯМ

Микросхема ПЛИС	Аппаратные затраты, CLB slices	Тактовая частота, МГц
VirtexE	2200	33
Virtex2	2000	50

Проект ядра АЯМ описан на языке VHDL и параметризован. Это означает, что как система команд, так и структура ядра могут быть изменены подстановкой соответствующего набора настроечных констант. Если какие-либо команды не используются в данном приложении, то с удалением этих команд из списка команд уменьшается оборудование ядра, которое необходимо было для их реализации. Например, если команды умножения не реализованы, то аппаратные затраты ядра становятся меньшими на 500 или 200 CLB slices для микросхем VirtexE и Virtex2, соответственно. Также могут быть добавлены любые команды пользователя, которые могут быть реализованы как в режиме неизвестной команды, так и аппаратно. Такой способ параметризации ядер микропроцессоров подробно описан в [7],[8],[9].

Ускоритель вычисления модульного умножения Монтгомери. Среди многих криптографических алгоритмов для работы с открытым ключом криптосистема RSA является наиболее распространенной и гибкой. В алгоритмах, подобных RSA, наиболее значимыми операциями являются операции модульного умножения и возведения в степень. Однако возведение в степень чисел длиной 512 бит и более приводит к очень большой трудоемкости этой операции, не позволяющей получить высокую пропускную способность криптопроцессора. Для быстрой реализации такой операции наиболее эффективными считаются алгоритмы, основанные на модульном умножении Монтгомери [5],[11],[12]. Поэтому алгоритм умножения Монтгомери реализован в рассматриваемом конфигурируемом криптопроцессоре.

Современные ПЛИС, такие как Xilinx Virtex2, имеют в своем составе десятки аппаратных умножителей, а сотни параллельных сумматоров могут в них быть сконфигурированными. Поэтому прямая форма алгоритма Монтгомери не может быть эффективно реализована в таких ПЛИС из-за высоких простоев аппаратных блоков. Например, умножитель Монтгомери, реализация в ПЛИС которого описана в [3]

использует большое количество программируемых логических элементов, при этом аппаратные умножители вообще не используются.

В работе [12] предложен модифицированный алгоритм Монтгомери, который можно эффективно использовать в современных ПЛИС. В этом алгоритме модульное умножение расщеплено на фазу умножения целых чисел и фазу модульной коррекции Монтгомери. При этом исходными данными являются: модуль N в n-битовом двоичном представлении, множитель $A=(a_{n-1}, a_{n-2},\ldots,a_0)_2$, множимое B – n-битовые двоичные числа. Результатом выступает произведение по модулю $R \equiv AB2^{-n} \bmod N$ и остаток $Q=(q_{n-1}, q_{n-2},\ldots,q_0)_2$.

В фазе умножения вычисляется произведение

$$AB=C=C_1 2^n + C_0, \qquad (1)$$

где $C_0=(c_{n-1}, c_{n-2},\ldots,c_0)_2$. Модульная коррекция Монтгомери состоит в n – кратном повторении итерации:

$$q_i = (P_i + c_i) \bmod 2; \qquad (2)$$
$$P_{i+1} = (P_i + q_i N + c_i) \mathrm{div} 2,$$

где $P_0 = 0$; $i=0,1,\ldots,n-1$; результат равен $R = P_n + C_1$.

В аппаратном криптографическом ускорителе этот алгоритм реализован для n=512 и n=1024. Он имеет ступени умножения и коррекции, вычисляющие целочисленное умножение и модулярную коррекцию Монтгомери, соответственно (см. рис. 1).

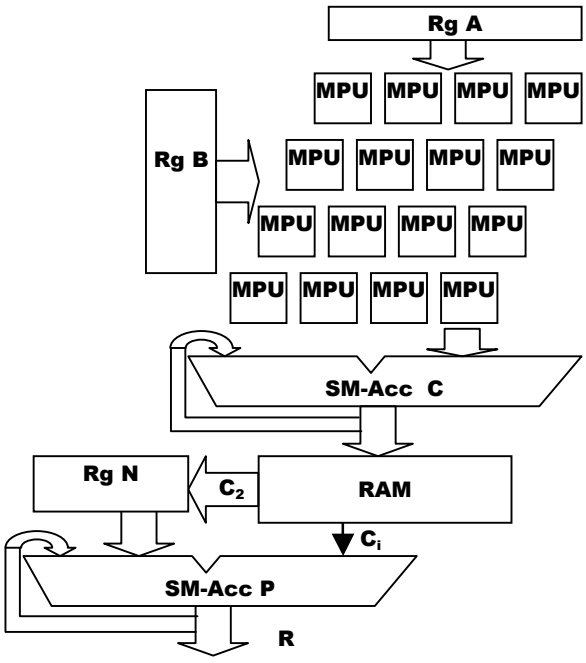

Рис.1 Структура аппаратного ускорителя модульного умножителя Монтгомери.

Первая ступень построена на базе аппаратного умножителя с размерами $k*l$=64*64 бит и сумматора частных произведений. Эта ступень полностью конвейеризована и исполняет умножение (1) для 1024-разрядных чисел с тактовой частотой до 150 МГц за 256 тактов, не считая тактов загрузки и очистки конвейера. 64-разрядные срезы результата C_0 запоминаются в буферном ОЗУ.

Вторая ступень состоит из накапливающего сумматора, который в каждом такте выполняет сложение операндов P_i , q_i N, и c_i, и выдает промежуточный результат со сдвигом вправо на один разряд. Операнд c_i приходит с буферного ОЗУ, в котором сконфигурирован однобитный выход. Такая структура адаптирована к реализации в ПЛИС Xilinx Virtex и не требует построения многовходового мультиплексора. Сумматор n=1024 – разрядных чисел разделен на части, образующие полностью конвейеризованную структуру.

Приведенные особенности аппаратного ускорителя вычисления модулярного произведения обеспечили повышение максимальной тактовой частоты до 150 МГц при его конфигурировании в ПЛИС Virtex2. Умножение Монтгомери для 1024 – разрядных операндов длится около 1800 тактов и может быть выполнено за 12 мкс. Аппаратные затраты такого ускорителя составляют 900 CLB slices и 16 блоков умножения.

Если входные данные поступают потоком, то два этапа алгоритма могут вычисляться параллельно. Для того. Чтобы ускорить выполнение второго этапа в двое, вторая ступень ускорителя должна иметь два сумматора, которые вычисляют две соседние итерации алгоритма модульной коррекции. В этом случае производительность ускорителя возрастает в 3.5 раза. При увеличении в k раз размеров $k*l$ аппаратного умножителя первой ступени и числа сумматоров второй ступени, соответственно, в k раз может вырасти производительность ускорителя.

Экспериментальные результаты. Ядро АЯМ и ускоритель вычисления модульного умножения Монтгомери были объединены в проекте экспериментального конфигурируемого криптопроцессора. Команды, по которым вычисляются модульные умножения Монтгомери для двух 512 или 1024 – разрядных чисел по модулю N, были встроены в систему команд ядра АЯМ. Считается, что операнды модульного умножения хранятся в предопределенных массивах длиной 16 или 32 слов, хранящихся во внутренней памяти данных, присоединенной к микропроцессорному ядру. В эту же память записывается массив результата. При исполнении такой команды ядро микропроцессора передает управление аппаратному ускорителю модульного умножения. Исходные данные читаются ускорителем из блока памяти данных, и в конце команды ускоритель выдает в память слова результата.

Вычисление модульного возведения в степень состоит в повторении команды модульного произведения. Для возведения в степень, представленную 32 – разрядным числом, необходимо выполнить, в среднем, 50 модульных произведений [11].

Экспериментальный криптопроцессор был сконфигурирован в ПЛИС Xilinx Spartan3-400. Эта

микросхема по структуре похожая на структуру ПЛИС Virtex2, но имеет существенно меньшую цену. Ядро АЯМ было сконфигурировано без аппаратного умножителя. Поэтому объем аппаратных затрат всего криптопроцессора был уменьшен до 2750 CLB slices. Аппаратный ускоритель функционирует с максимальной тактовой частотой 100 МГц, а ядро микропроцессора имеет тактовую частоту 50 МГц.

Алгоритм возведения в степень для 32-разрядного показателя вычисляется приблизительно за 900 микросекунд. Для сравнения, специализированная микросхема SLE88CFX4000P фирмы Infineon выполняет аналогичный алгоритм RSA за 4000 микросекунд при работе на предельной тактовой частоте 66 МГц.

Заключение. Предлагаемый криптографический процессор, основанный на гибком ядре микропроцессора и аппаратном ускорителе модульного умножения Монтгомери, имеет все преимущества специализорованных криптопроцессоров, такие как высокое быстродействие, быстрая смена криптографического алгоритма, простое включение интерфейса. Его ядро микропроцессора с RISC – архитектурой может выполнять форматирование данных, обмен ими с внешними устройствами, протоколы коммуникации, а также системные и другие функции. Высокая производительность криптопроцессора сравнима или выше производительности лучших промышленных специализированных микросхем для защиты данных. Малый объем аппаратуры криптопроцессора позволяет его конфигурирование в недорогих ПЛИС. При использовании более дорогих и емких ПЛИС возможно пропорциональное наращивание быстродействия криптопроцессора.

ЛИТЕРАТУРА

1. Wollinger T., Guajardo J., Paar C. Cryptography in Embedded Systems: An Overview. //Proc. of the Embedded World. –2003. Exhibition and Conf. –P.735-744.

2. Rouvroy G., Standaert F.-X., Quisquater J.-J. Efficient Uses of FPGAs for Implementations of DES and Its Experimental Linear Cryptanalysis. IEEE Trans. on Computers. Vol. 52. -N. 4. -2003. -P.1-10.

3. Tiountchik A., Trichina E.: Modular Exponentiation on Fine-Grained FPGA.// Proc. RSA Conf. CT-RSA 2001, San Francisco, CA, USA, April 8-12. –2001. -P.223-234.

4. Elbirt A. J., Yip W., Chetwynd B., Paar, C. An FPGA-Based Performance Evaluation of the AES Block Cipher Candidate Algorithm Finalists. //IEEE Trans. on VLSI Systems. –2001. –Vol.9, no4, pp. 545-557.

5. Blum T., Paar C. High-Radix Montgomery Modular Exponentiation on Reconfigurable Hardware.// IEEE Trans. on Computers. –Vol.50. –2001. -N7. -P.759 – 764.

6. Tikkanen K., Haennikaeinen M., Haemaelaeinen T., Saarinen J. Advanced Prototype Platform for a Wireless Local Area Network.// The European Signal Processing Conf. (EUSIPCO 2000). –Vol.4. -Sep 2000. -P.2309-2312.

7. Lepekha V., Sergienko A., Kaniewski J. VHDL-model of ultrafast microcontroller 8051.// Pracy III Konferencji Krajowej „Reprogramowalne uklady cyfrowe”, RUC’2000, Szczecin, Poland. -2000, pp.35-41.

8. O.Maslennikov, Ju.Shevtshenko, A.Sergyienko. Configurable microcontroller array.// Proc. of the 3-d Int. Conf. on Parallel Computing in Electrical Engineering. PARELEC'2002, Warsaw, Poland. 22-25 Sept. -2002. -P. 47-49.

9. Maslennikov O., Shevtshenko Ju., Sergyienko A. Configurable Microprocessor Array for DSP Applications.// Lecture Notes in Computer Science. -Vol.3019. –2004. -P.36-41.

10. Сергиенко А.М.: VHDL для проектирования вычислительных устройств. Киев: Диасофт. – 2003. 208с.

11. Kotch C.K., Acar T., Kaliski B.S. Analysing and Comparing Montgomery Multiplication algorithms.// IEEE Micro. –1996. -Vol.16. -N3. –P.26-33.

12. Yang C.-C., Chang T.-S., Jen C.-W. A new RSA cryptosystem hardware design based on Montgomery's Algorithm.// IEEE Trans. Circuits and Systems - II: Analog and Digital Signal Processing. -Vol.45. –1998. N7. -P.908-913.

13. Security & Chip Card ICs SLE 88CFX4000P. // Infineon Technologies. Preliminary Short Product Information. –2004. –10 P.