3. Delphi 7 : Справочное пособие / А. Я. Архангельский. — Москва: Бином-Пресс, 2003. — 1024 с.: ил. — Предм. указ.: с. 1005-1022. — ISBN 5-9518-0027-7.

РАЗРАБОТКА АЛГОРИТМА ВЕРИФИКАЦИИ DES

Кожевников Д. С.[1], Халеев А. А.[1], Торгаев С.Н.[1,2]
[1]Томский государственный университет, г. Томск
[2]Томский политехнический университет, г. Томск
Научный руководитель: Евтушенко Г.С., д.т.н., профессор кафедры промышленной и медицинской электроники

Информационные технологии, в том числе технологии передачи информации, активно и непрерывно развиваются как в России, так и за рубежом, и соответственно повышается необходимость защиты используемой и передаваемой информации. Криптографические методы защиты информации (ЗИ) используют математические методы и модели для шифрования и дешифрования защищаемой информации, причем шифрование и дешифрование могут осуществляться как программно, так и аппаратно; используется также программно-аппаратное шифрование. В случае аппаратного шифрования по алгоритму шифрования реализуется логическая схема (шифратор): входные последовательности, поступающие на данную схему, представляют собой защищаемую информацию, а соответствующие выходные последовательности – зашифрованную информацию. В настоящее время активно расширяются масштабы использования аппаратных шифров, обеспечивающих высокую скорость шифрования (более 1500 Мбит/с). Логические схемы шифраторов и дешифраторов постоянно совершенствуются. Поскольку ни один из известных методов синтеза не доставляет оптимальные схемы, то одним из способов совершенствования таких схем является их оптимизация относительно различных критериев [1].

Аппаратные реализации на сегодняшний день становятся все более и более распространёнными, поскольку они могут обеспечивать более высокую производительность по сравнению с программными. В настоящее время практически каждая телекоммуникационная система имеет интегрированные (встроенные) аппаратные компоненты. Эти компоненты могут быть использованы в различных целях, начиная со специализированных устройств, которые достаточно редко используются в ходе работы технической системы и заканчивая компонентами, которые контролируют работу всей системы. Таким образом, аппаратные компоненты должны быть тщательно протестированы и верифицированы

по отношению ко всем функциональным и не функциональным требованиям. В данной работе мы в качестве исследуемой системы принимаем алгоритм шифрования, создаем его программную и аппаратную реализации и верифицируем их при помощи мутационного тестирования. Главным критерием при верификации являлось создание полноты синтезируемых тестов, которые могут покрыть все случайно вносимые ошибки в код алгоритма [1, 2].

Описание работы схемы управления

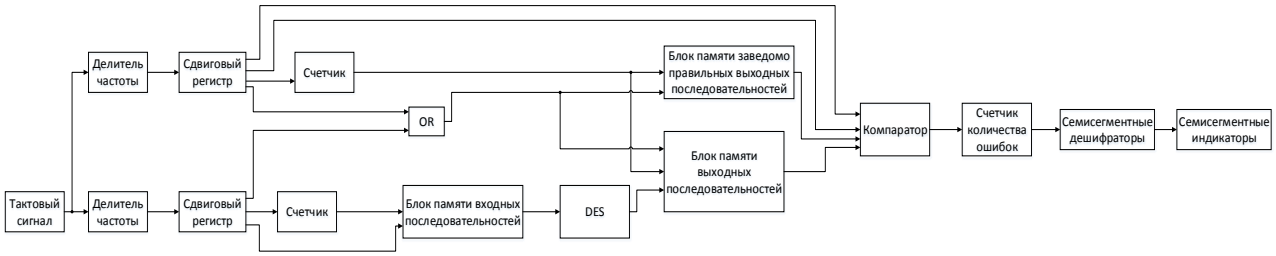

Рис. 1. Схема управления

На рисунке 1 представлена функциональная схема работы схемы управления. Входные последовательности шифруются, записываются в память. Далее проводится сравнение правильности шифрования и выдается результат о количестве ошибок.

Этапы работы схемы:

1) Считывание из памяти входных последовательностей, шифрование алгоритмом DES и запись в память.

После включения тумблера SW0 (PIN_AC9) на вход разрешения делителя поступает неактивный сигнал и на выходе появляются импульсы с частотой 10 кГц. Самый первый импульс приходит на сдвиговый регистр и устанавливает на SO0 «1» это является тактовым сигналом для памяти ROM, где хранятся входные последовательности для шифрования алгоритмом DES. Так как в начальный момент времени на выходе счетчика адреса установлен «0» на всех выходах – по тактовому сигналу из памяти будет прочтена ячейка с нулевым адресом и данные из нее отправлены на вход алгоритма шифрования. Зашифрованные данные с выхода алгоритма поступают на вход блока памяти RAM. На эту память дублируется адрес ячейки с памяти ROM, поэтому данные из нулевой ячейки одной памяти запишутся по тактовому сигналу в нулевую ячейку другой памяти. Для того, чтобы не получилось одновременного считывания-записи и данные успели зашифроваться, реализован регистр сдвига, который подает тактовые сигналы на блоки памяти, сдвинув их во времени. Пока счетчик адреса не досчитал до предельного значения, на его выходах A = «0» и B = «1». Эти сигналы подключены к ножкам управления чтения/записи на память RAM. Следующий тактовый сигнал

на регистре сдвига приведет к установке «1» на выходе SO2. Это будет являться тактовым сигналом для счетчика адреса. Прибавив единицу к своему предыдущему значению, счетчик будет удерживать это значение на адресных входах памяти. Далее регистр сдвига сбрасывается и цикл начинается сначала, но уже считывание и запись происходит в других ячейках. Так повторяется до тех пор, пока счетчик адреса не достигает до предельного значения. Предельным значением является количество входных последовательностей. После этого на выходах счетчика адреса A = «1», B = «0» - это установка разрешения считывания с памяти RAM.

2) Считывание зашифрованных данных из памяти RAM, сравнение их с заведомо правильными выходными последовательностями и вывод количества несовпадений.

После включения тумблера SW1 на вход разрешения верхнего делителя частоты поступает неактивный сигнал и на выходе делителя появляются импульсы с частотой 10 кГц. Самый первый импульс приходит на верхний регистр сдвига, на его выходе SO0 появляется единица – это является тактовым сигналом для памяти RAM и ROM одновременно. Так как в этот момент на всех выходах верхнего счетчика адреса был «0»– на выходных шинах данных памяти RAM и ROM устанавливаются данные из нулевой ячейки. Следующий тактовый импульс на регистр сдвига приведет к смене адреса на шинах адреса блоков памяти. Третий импульс будет являться тактовым для цифрового компаратора. По этому импульсу он побитово сравнивает выходные шины данных с блоков памяти. Если данные не отличаются, то на выходе компаратора установлен низкий уровень. При обнаружении ошибки на выходе устанавливается высокий уровень, а следующий тактовый импульс на регистре сдвига приводит к установке высокого уровня на ножке R компаратора и высокий уровень на выходе сбрасывается. Это сделано для того, чтобы не случилось наложения – высокий уровень не был установлен на выходе компаратора при смене адреса и счетчик ошибок, работающий по фронту, корректно подсчитывал ошибки. Как только на выходе компаратора появляется нарастающий фронт – это является для счетчика ошибок тактовым сигналом. Он прибавляет единицу и выводит количество на выходные шины. Далее с помощью дешифраторов происходит преобразование двоичного кода с выхода счетчика ошибок в шестнадцатеричный и вывод на семисегментные индикаторы. Данный цикл повторяется предельное количество + 1 раз. Предельным количеством является количество входных последовательностей. Прибавить единицу необходимо по причине того, что счетчик сменив адрес на предельную ячейку останавливает счет и останавливает поступление импульсов на регистр сдвига. Следовательно, на компаратор

не приходит тактовый импульс для сравнения данных. Данная проблема и решается увеличением счета на один такт.

Описание DES

DES представляет собой блочный шифр, которые преобразует данные открытого текста 64-битовыми блоками. Соответственно, на вход алгоритма подаётся 64-битовый блок открытого текста, а с выхода снимается 64-битовый блок шифротекста. Отметим, что DES является симметричным алгоритмом, а именно, для шифрования и дешифровки используются одинаковые алгоритм и ключ (за исключением небольших различий в использовании ключа). [1]

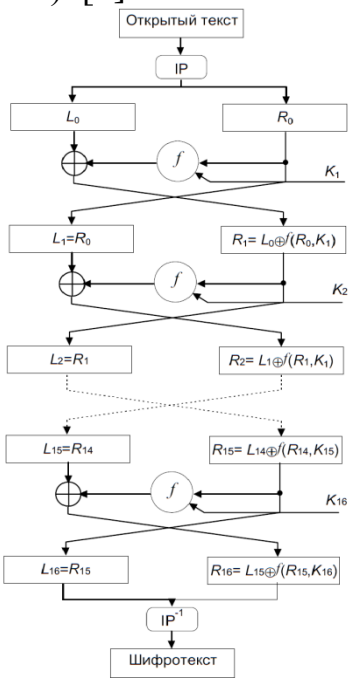

Рис. 2. Алгоритм шифрования DES.

Длина ключа равна 56 битам. Отметим, однако, что ключ, как правило, представляется 64-битовым числом, но каждый восьмой бит используется для проверки четности и игнорируется; Биты четности являются наименьшими значащими битами байтов ключа. Согласно стандарту DES, ключ, который может быть любым 56-битовым числом, можно изменить в любой момент времени. Тем не менее, известно, ряд чисел являются слабыми ключами в смысле стойкости шифра, поскольку первоначальное значение ключа, в ходе алгоритма шифрования, расщепляется на две половины, каждая из которых сдвигается независимо. Примером может служить вектор, наполовину состоящий из единиц и наполовину из нулей. В этом случае для всех этапов алгоритма используется один и тот же ключ. [1] На рисунке 2 приведена схема алгоритма шифрования DES.

Алгоритм DES был реализован аппаратно на ПЛИС типа FPGA Cyclone II Starter Development Board компании Altera (рисунок 3).

Рис. 3. Внешний вид платы Cyclone II Starter Development Board

Помимо аппаратной реализации докладчиками была разработана программная реализация алгоритма шифрования DES. Соответствующая программа была написана на языке C++ с использованием интегрированной среды разработки Microsoft Visual Studio 2013 Express Edition.

Программа принимает на вход 64-битовый вектор (блок открытого текста) и случайным образом сгенерированный ключ той же размерности. На выходе получается 64-битный блок шифротекста. Отметим, что программная реализация использует технологию ООП, а именно, для обработки булевых векторов размерности более 32 бит был реализован класс «Длинный булев вектор».

При использовании FPGA технологии поведение синтезируемой схемы и/или ее компонентов описывается на языке логического проектирования, например, VHDL или Verilog. Далее с использованием подходящего программатора проектируется схема из программируемых логических блоков (ПЛБ), которая впоследствии преобразуется в логическую схему соответствующего цифрового устройства. Поэтому для верификации функционирования аппаратных реализаций, синтезированных с применением технологии FPGA, имеет смысл, в первую очередь, рассматривать возможные ошибки в описаниях в соответствующих языках логического проектирования Verilog, VHDL и др. В данной работе мы рассматриваем верификацию Verilog-описанием, тем не менее, мы отмечаем, что предлагаемый метод естественным образом переносится и на другие описания в языках логического проектирования.

Тестирование проводилось в два этапа: на первом этапе на вход проверяемой реализации подавалось 10000 тестовых наборов с различным шагом. Для оценки полноты тестирования случайным образом строились одиночные и кратные мутанты Verilog-кода типа замены элемента XOR на элемент OR в первом раунде DES. Все кратные неисправности

обнаруживаются достаточно просто большим количеством тестовых наборов. Результаты экспериментов приведены в таблице 1.

Результаты тестирования аппаратной реализации Таблица 1

Одиночная замена XOR на OR		Кратная замена XOR на OR	
№ элемента XOR	Количество наборов, обнаруживающих мутанта	№ элементов XOR	Количество наборов, обнаруживающих мутанта
1,2,4,5,7-11,13-17,21-23,25,26,28-31,34,35,37,39-42, 45-47	0	3, 6	6605
3	3149	3, 6, 12	8301
6	4992	3, 6, 12, 17	8301
12,24,32,44	5000	18, 20	5002
18,20	5002	32,33	7616
19,36,38,43	10000	12,24,32,44	10000
27	4880	12,24	7500
33	5226	32,44	7500
48	5005		

Как видно из таблицы, полнота синтезируемых тестов практически не зависит от шага при синтезе тестов и остается достаточно низкой для функциональных ошибок (на уровне 50-60% для случайно вносимых ошибок в Verilog-описание), даже при небольшом шаге тестирования. На втором этапе тест достраивался с использованием мутантов verilog-описания, таких как замена элемента XOR на элемент OR в одном из раундов DES, обрыв связей, «залипание» некоторых входов и т.п. и построения соответствующей различающей последовательности для неэквивалентных мутантов. При рассмотрении достаточно большого количества мутантов полнота синтезируемых тестов для случайно вносимых ошибок в Verilog-описание тестов оказалась более 90% (при той же длине теста, что и при случайном тестировании). Таким образом, экспериментальные результаты показывают эффективность предложенного подхода к верификационному тестированию ПЛИС на основе FPGA технологии.

Работа частично поддержана грантом РФФИ № 14-08-31640 мол_а.

Список информационных источников

1. Шнайер Б. - Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке С. — М.: Триумф, 2002. — 816 с.

2. Shatilov N. P. , Kozhevnikov D. S. , Kushik N. G. , Torgaev S. N. On Using Program Specifications in Hardware Testing // 15th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM-2014): proceedings, Алтай, 30 June-4 July 2014. - Novosibirsk: NSTU, 2014 - p. 154-157

АКУСТИЧЕСКИЙ УРОВНЕМЕР ДЛЯ КОНТРОЛЯ УРОВНЯ СЫПУЧИХ МАТЕРИАЛОВ

Комаров С.Е., Цзюй Янян
Томский политехнический университет, г. Томск
Научный руководитель: Солдатов А.И.., д. т. н., профессор кафедры промышленной и медицинской электроники

Контроль уровня сыпучих материалов довольно распространенная задача в промышленности и сельском хозяйстве. Хранение сыпучих веществ может осуществляться в открытых и закрытых сосудах, резервуарах, хранилищах и других ёмкостях. Уровнемеры так же называют датчиками/сигнализаторами уровня, преобразователями уровня. Главное отличие уровнемера от сигнализатора уровня — это возможность измерять градации уровня, а не только его граничные значения [1].

Одним из наиболее распространенных методов бесконтактного измерения уровня в промышленности является акустическая или ультразвуковая эхолокация.

В сухом, чистом и неподвижном воздухе поглощение акустических колебаний имеет наименьшую величину и осуществляется молекулами кислорода. Во влажном воздухе поглощение возрастает, но остается меньшим по величине, чем в турбулентном воздушном потоке. Поглощение звука во влажном воздухе происходит за счет взаимодействия молекул кислорода и водяного пара. Часть звуковой энергии при неупругих столкновениях молекул переходит в колебательную энергию атомов в молекулах. Для всех частот с увеличением относительной влажности поглощение звука сначала возрастает, при влажности 10...20% достигает максимума и при дальнейшем увеличении влажности монотонно уменьшается. Воздух, наполненный туманом, не может не вызывать добавочного поглощения и рассеяния звука. Затухание звука в тумане происходит благодаря его рассеиванию на каплях, так как капли участвуют