Мыцко Евгений Алексеевич, студент кафедры вычислительной техники Института кибернетики ТПУ.
E-mail: EvgenRus70@mail.ru
Область научных интересов: программная и аппаратная реализации алгоритмов вычисления контрольной суммы, тестирование и сравнение по быстродействию алгоритмов вычисления CRC.

Мальчуков Андрей Николаевич, канд. техн. наук, доцент кафедры вычислительной техники Института кибернетики ТПУ.
E-mail: jgs@tpu.ru
Область научных интересов: помехоустойчивое кодирование, полиномиальные коды, системы проектирования помехоустойчивых полиномиальных кодов, алгоритмы поиска образующих полиномов, быстродействующие алгоритмы кодирования и декодирования данных полиномиальными кодами.

УДК 681.3.06

ИССЛЕДОВАНИЕ АППАРАТНОЙ РЕАЛИЗАЦИИ АЛГОРИТМОВ ВЫЧИСЛЕНИЯ КОНТРОЛЬНОЙ СУММЫ CRC32

Е.А. Мыцко, А.Н. Мальчуков

Томский политехнический университет
E-mail: EvgenRus70@mail.ru, jgs@tpu.ru

Проведено исследование аппаратных реализаций матричного и табличного алгоритмов вычисления контрольной суммы CRC32 на ПЛИС Cyclone фирмы Altera макета SDK-6.1 на основе сравнения блоков вычисления CRC32 по занимаемым логическим ячейкам и временным задержкам.

Ключевые слова:
Контрольная сумма, табличный алгоритм, матричный алгоритм, CRC32, аппаратная реализация, логическая ячейка.

В работе [1] рассмотрена программная реализация матричного алгоритма вычисления контрольной суммы CRC32 в сравнении с табличным алгоритмом. При программной реализации матричный алгоритм требует в 8 раз меньше памяти, чем табличный, в то время как табличный алгоритм выигрывает по скорости вычисления. Для исследования аппаратной реализации данных алгоритмов спроектированы устройства с использованием блочно-ориентированного подхода [2] и языком описания аппаратуры VHDL [3]. В среде QuartusII сформированы конфигурации для программируемой логической интегральной схемы (ПЛИС) Cyclone фирмы Altera [4] макета SDK-6.1 (рис. 1).

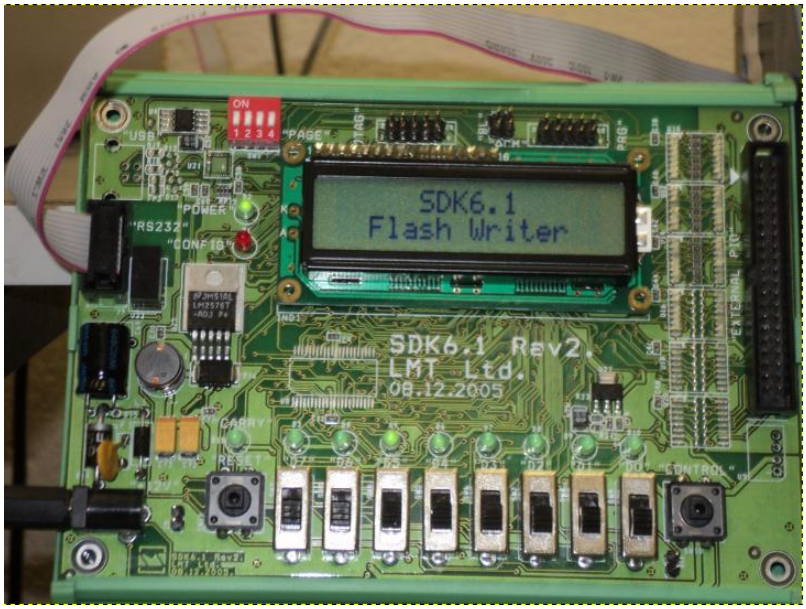

Рис. 1. Учебно-лабораторный макет SDK-6.1

Исследование аппаратной реализации алгоритмов

Для проектирования схем в среде Quartus II фирмы Altera, используя блочно-ориентированный подход, применяют два типа логики: комбинаторную и регистровую [5]. Логическая ячейка (Logic Cell) – общий термин для основных блоков микросхем, на которых реализуется проект [5]. Логическая ячейка состоит из элемента регистровой логики (триггера) и универсального четырехвходового логического элемента LUT (Look Up Table), который может быть запрограммирован на реализацию любой четырехвходовой логической функции. Именно на этих элементах реализуется вся комбинаторная логика проекта в ПЛИС. Если нужна функция больше 4-х входов, то используется два таких элемента, больше 7 входов – три ячейки. Регистр (триггер) может быть сконфигурирован как latch (защёлка), D, T, RS, JK триггер или зашунтирован для реализации только комбинаторной логики.

В среде проектирования Quartus II для каждого блока функциональной схемы приводится ряд значений, таких как Logic Cells (общее количество ячеек для элемента), LC Resisters (общее количество регистров, входящих в ячейку), LUT-Only LCs (количество элементов только комбинаторной логики, входящих в ячейку), Register-Only LCs (количество элементов только регистровой логики, входящих в ячейку), LUT/Register LCs (количество элементов, использующих как комбинаторную, так и регистровую логику).

Для исследования аппаратной реализации на основе полученных данных была составлена сводная таблица занимаемых логических ячеек для блока вычисления контрольной суммы «CRC32_vhdl» в зависимости от алгоритма (табл. 1).

Таблица 1. Количество занимаемых логических ячеек для блока «CRC32_vhdl»

Алгоритм/ лог. ячейки	Количество занимаемых ячеек, шт.				
	Logic Cells	LC Registers	LUT-Only LCs	Register-Only LCs	LUT/Register LCs
Табличный	125	65	60	0	65
МС 1 байт	125	65	60	0	65
МС 2 байта	169	65	105	0	65
МС 4 байта	280	66	214	0	66

Как видно из табл. 1, значение Logic Cells состоит из суммы значений LC Registers и LUT-Only LCs, а значение LC Registers, в свою очередь, состоит из суммы значений Register-Only LCs и LUT/Register LCs.

По полученным данным была составлена таблица изменений количества занимаемых логических ячеек в процентах для аппаратных реализаций различных вариантов матричного алгоритма относительно табличного (табл. 2).

Таблица 2. Изменение количества логических ячеек относительно табличного алгоритма

Алгоритм/ лог. ячейки	Изменение количества логических ячеек, %				
	Logic Cells	LC Registers	LUT-Only LCs	Register-Only LCs	LUT/Register LCs
МС 1 байт	0	0	0	0	0
МС 2 байта	+35	0	+73	0	0
МС 4 байта	+124	+1,5	+256	0	+1,5

Из табл. 2 видно, что реализация матричного однобайтового алгоритма в данном случае требует столько же логических ячеек, как и для реализации табличного алгоритма.

При реализации двухбайтового матричного алгоритма общее количество логических ячеек увеличивается на 35 %, а количество элементов комбинаторной логики увеличивается на 73 %. Для реализации четырёхбайтового матричного алгоритма потребовалось на 256 % больше элементов комбинаторной логики. Данные изменения связаны с особенностью аппаратной реализации матричных алгоритмов. При реализации двухбайтового и четырёхбайтового алгоритмов увеличивается разрядность блока (с 8 до 16 и 32 бит соответственно), который вычис-

ляет контрольную сумму CRC32. Также на блоке расчёта CRC32 появляется дополнительный вход для вычисления контрольной суммы в случаях, когда объём данных не кратен требуемому при реализации определённого варианта матричного алгоритма. Данные факторы приводят к увеличению количества LUT элементов, требуемых для построения блока вычисления CRC32.

Преимущество многобайтовых матричных алгоритмов заключается в уменьшении числа тактов, требуемых для расчёта CRC32. Так, если при передаче данных доступны блоки по 4 байта, то для расчёта CRC четырёхбайтовым матричным алгоритмом требуется в 4 раза меньше тактов, чем по табличному или матричному однобайтовому алгоритму.

Помимо количества логических ячеек, занимаемых блоком вычисления CRC32, также исследованы временные задержки вычислительных блоков для каждого алгоритма.

На основе полученных данных составлена таблица временных задержек блоков вычисления CRC32 для каждого алгоритма (табл. 3). При этом учитывалось, что если в системе доступно сразу по 2 байта для обработки за такт, то задержка блоков однобайтовых алгоритмов увеличивается в 2 раза. При доступных 4-х байтах для обработки за такт задержка блока двухбайтового алгоритма увеличивается в 2 раза, а для однобайтовых – в 4 раза.

Таблица 3. Временные задержки для блока вычисления CRC32

Алгоритм	Задержка при последовательной побайтной передаче, нс	Задержка при передаче по 2 байта, нс	Задержка при передаче по 4 байт, нс
Табличный	7,308	14,616	29,232
Матричный 1 байт	7,309	14,618	29,236
Матричный 2 байта	8,627	8,627	17,254
Матричный 4 байта	11,025	11,025	11,025

На основе табл. 3 была составлена таблица ускорений блоков вычисления CRC32 матричного алгоритма относительно табличного (табл. 4).

Таблица 4. Ускорение блока вычисления CRC матричного алгоритма относительно табличного

Матричный алгоритм	Ускорение при последовательной побайтной передаче, %	Ускорение при передаче по 2 байта, %	Ускорение при передаче по 4 байта, %
1 байт	0	0	0
2 байта	-18	+69	+69
4 байта	-50	+32	+165

Как видно из табл. 4, блок вычисления CRC32 матричного однобайтового алгоритма имеет такую же задержку данных, как и для табличного алгоритма, в то время, как блок вычисления матричного четырёхбайтового алгоритма отстаёт по скорости от табличного на 50 % при последовательной побайтной передаче данных. В случаях, когда при передаче за цикл доступно больше одного байта, многобайтовые алгоритмы опережают по скорости однобайтовые.

Для наглядного представления данных на рис. 2–4 в виде гистограмм приведены результаты исследования по скоростным характеристикам аппаратных реализаций матричного алгоритма относительно табличного. В гистограммах по горизонтали результаты реализаций матричных алгоритмов с разным сдвигом: 1, 2 и 4 байта.

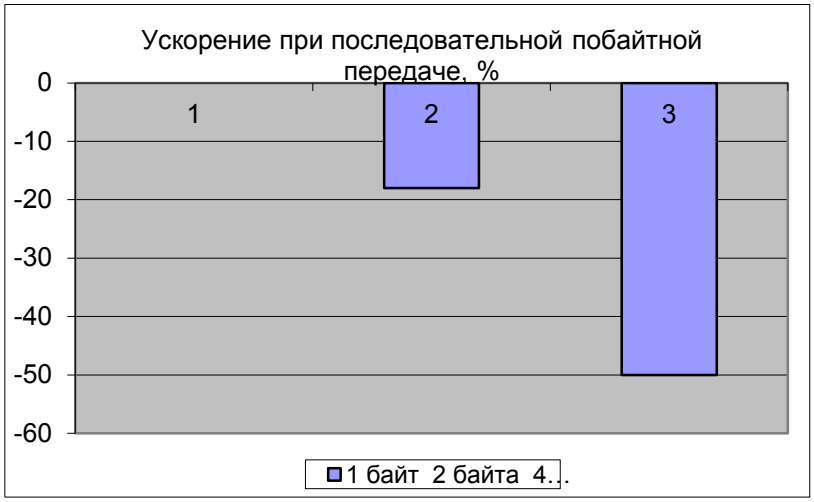

Рис. 2. Ускорение при последовательной побайтной передаче данных

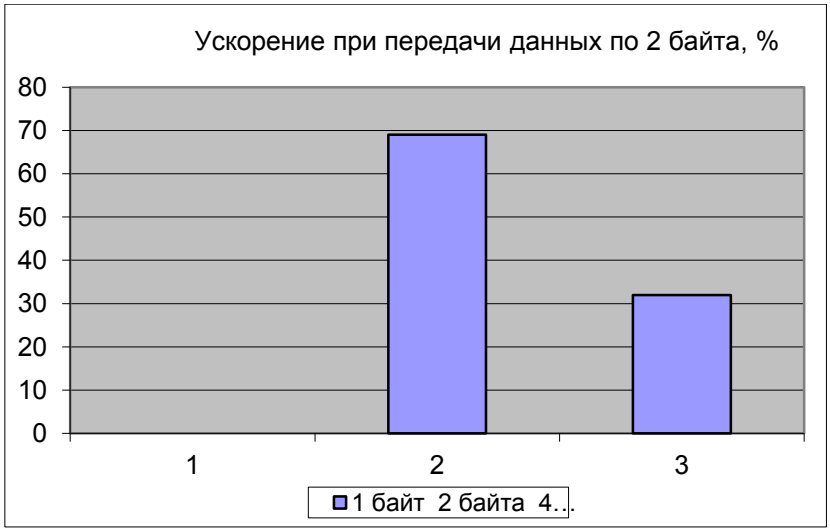

Рис. 3. Ускорение при передаче данных по 2 байта

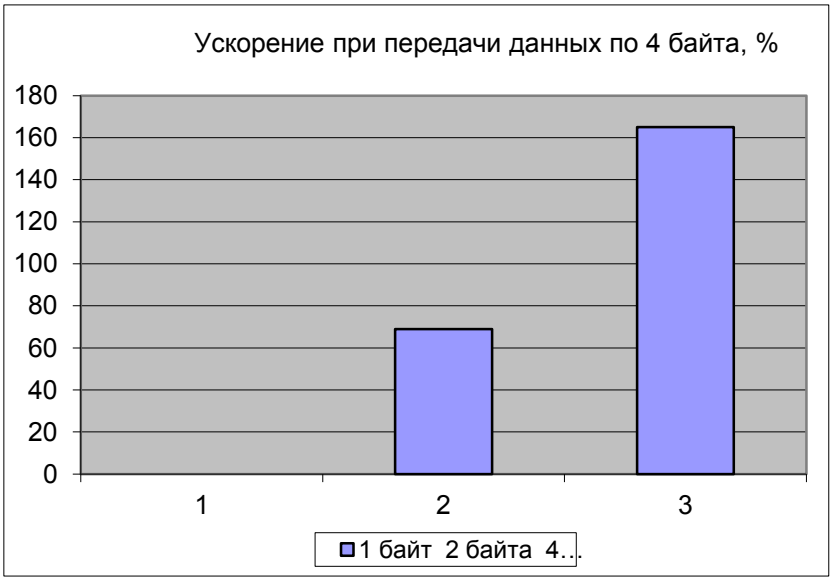

Рис. 4. Ускорение при передаче данных по 4 байта

Таким образом, картина, полученная в результате аппаратной реализации, в целом отличается от полученной в работе [1]. В программной реализации матричные алгоритмы выигрывали у табличных по требуемому объёму памяти на хранение матрицы, в то время как в аппаратной реализации для построения блока вычисления матричных алгоритмов требуется большее количество логических ячеек, но имеется преимущество в скорости вычисления контрольной суммы при наличии сразу нескольких байт данных для обработки.

Заключение

На основе алгоритмов расчёта контрольной суммы CRC32, используемых в программных реализациях [1], были спроектированы устройства для аппаратных реализаций алгоритмов на ПЛИС Cyclone. При проектировании функциональной схемы использовался блочно-ориентированный подход (BBD). Было установлено, что для однобайтовых алгоритмов (табличный и матричный) количество задействованных логических ячеек для блока вычисления CRC32 одинаково. Для двухбайтового и четырёхбайтового алгоритмов блок вычисления CRC32 имеет существенные отличия от блоков однобайтовых алгоритмов. Для их реализации потребовалось значительно больше логических ячеек (на 35 % для двухбайтового и на 124 % для четырёхбайтового алгоритма).

По результатам временных задержек блоков вычисления CRC32 можно сказать, что при аппаратной реализации однобайтовые алгоритмы имеют одинаковую скорость вычисления, в то время как у двухбайтового матричного алгоритма задержка блока вычисления на 18 % больше, чем у табличного алгоритма, а четырёхбайтовый алгоритм отстаёт по скорости на 50 %. Однако при доступном блоке данных, который можно обрабатывать за такт объёмом в 4 байта, матричный четырёхбайтовый алгоритм значительно опережает по скорости (на 165 %) однобайтовые алгоритмы.

СПИСОК ЛИТЕРАТУРЫ

1. Мыцко Е.А., Мальчуков А.Н. Исследование программных реализаций табличного и матричного алгоритмов вычисления контрольной суммы CRC32 // Вестник науки Сибири. Серия Информационные технологии и системы управления. – 2011. – № 1 (1). – С. 273–278. URL: http://sjs.tpu.ru/journal/issue/view/2/ showToc/sect/4 (дата обращения: 06.08.2012).
2. Еремин В.В., Мальчуков А.Н. О применении блочно-ориентированного подхода к разработке устройств на ПЛИС // Вестник науки Сибири. Серия Информационные технологии и системы управления. – 2011. – № 1 (1). – С. 379–381. URL: http://sjs.tpu.ru/journal/issue/view/2/showToc/sect/4 (дата обращения: 06.08.2012).
3. Бибило П.Н. Основы языка VHDL. 3-е изд., доп. – М.: Изд-во ЛКИ, 2007. – 328 с.
4. EP1C3T144C8 datasheet // DATASHEET.SU. 2007–2012. URL: http://pdf3.datasheet.su/Altera/EP1C3T144C8.pdf (дата обращения: 06.08.2012).
5. Мяльк Р.А., Шестаков В.Ю. Терминология проектирования цифровых устройств на ПЛИС: учебное пособие // Санкт-Петербургский Государственный университет аэрокосмического приборостроения. 2005. URL: http://guap2151.narod.ru/ 8_ucos_Altera_Glossary.pdf (дата обращения: 06.08.2012).

Поступила 10.09.2012 г.