БЫСТРОДЕЙСТВУЮЩИЕ АЛГОРИТМЫ ВЫЧИСЛЕНИЯ КОНТРОЛЬНОЙ СУММЫ НА ПРИМЕРЕ CRC8

Мальчуков А.Н., Осокин А.Н.
Научный руководитель: Осокин А.Н.
Томский политехнический университет
malchukov@vt.tpu.ru, osokin@vt.tpu.ru

В данной работе предлагается быстродействующий матричный алгоритм вычисления контрольной суммы на примере CRC8, легко реализующийся на комбинационных схемах, и не требующий применения запоминающего устройства при аппаратной реализации. Матричный алгоритм требует меньших объёмов памяти запоминающего устройства при его программной реализации в отличие от табличного алгоритма.

Введение

Контрольная сумма – это некоторое значение, вычисленное для последовательности байт данных с помощью определённого алгоритма, которое используется на приёмной стороне для подтверждения корректности полученных данных. Первоначально контрольная сумма использовалась в системах с наличием обратной связи (переспроса) для обнаружения ошибок, возникающих в зашумлённых каналах связи. Позднее, с развитием криптографических хеш-функций (алгоритмов хеширования), контрольные суммы стали использоваться для подтверждения целостности и подлинности данных. Обычно контрольная сумма посылается (считывается) в конце сообщения:

<блок данных> <контрольная сумма>.

В настоящее время существует множество алгоритмов получения контрольной суммы: сложение байт, CRC (избыточный циклический код), MD5, SHA и т.д. CRC традиционно используется в проводных и беспроводных протоколах передачи данных (IEEE 802.3 – Ethernet, Bluetooth, ZigBee, CAN, Fibre Channel и т.д.) для контроля целостности управляющих фрагментов или кадров данных. Далее речь пойдёт об алгоритмах вычисления контрольных сумм CRC.

“Стандартный” алгоритм

Под стандартным алгоритмом подразумевается алгоритм, вычисляющий контрольную сумму CRC побитно, т.е. в каждом такте (итерации) данные последовательно продвигаются в некотором регистре на один бит, и в итоге в этом регистре получают контрольную сумму. Этот алгоритм широко известен по его аппаратной реализации на регистрах с обратной связью (рис. 1).

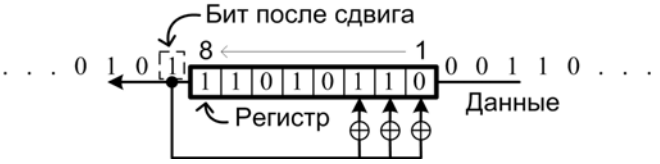

Рис. 1. Схематичное представление работы стандартного алгоритма на примере CRC8 (x^8+x^2+x+1)

На рис. 1 схематично показана работа простого алгоритма. Словесно для CRC8 его можно описать следующим образом.

Начало. Регистр (массив) 8 бит содержит нулевое значение, данные поступают в регистр через его младший разряд к старшим, начиная со старшего разряда данных последовательным сдвигом.

Шаг 1. Сдвигаем данные в регистре на один бит от младших к старшему разряду, в младший разряд регистра заносится бит из потока данных.

Шаг 2. Если выдвинутый бит из 8 разряда регистра равен 0, то переходим на шаг 4.

Шаг 3. Инвертируем содержимое 1, 2 и 3 разрядов регистра.

Шаг 4. Если ещё не все биты поступили в регистр из потока данных, то переходим на шаг 1.

Конец. В регистре содержится контрольная сумма CRC8.

При вычислении контрольной суммы CRC8 для последовательности бит данных в конец добавляют 8 нулей. При проверке контрольной суммы через регистр пропускают последовательность бит данных вместе с контрольной суммой в конце. В итоге проверки нулевое значение регистра соответствует безошибочному приёму. Если регистр содержит ненулевое значение, то произошло искажение данных в информационном блоке и/или в контрольной сумме.

Данный алгоритм вычисления контрольной суммы CRC8 требует выполнения множества итераций, что существенно замедляет процесс вычисления. В качестве ускорения вычисления контрольной суммы CRC в [1] предлагается сдвигать данные не по 1 биту за итерацию, а по 8 бит (байту). Предложенный в [1] алгоритм называется табличным.

Табличный алгоритм

При последовательном сдвиге данных по байту (вместо одного бита) за итерацию (такт) необходимо знать изменения, которые должны были происходить в течение 8 сдвигов при обычном алгоритме (для CRC8 – инвертирование трёх младших разрядов, в случае единичного значения выдвинутого разряда), поэтому их

(изменения) необходимо предварительно вычислить и занести в таблицу. Адресом в такой таблице будет служить содержимое регистра до сдвига (выталкиваемый байт при сдвиге). Содержимое таблицы необходимо будет сложить по модулю два со значением регистра, в котором содержится байт данных после сдвига (рис. 2).

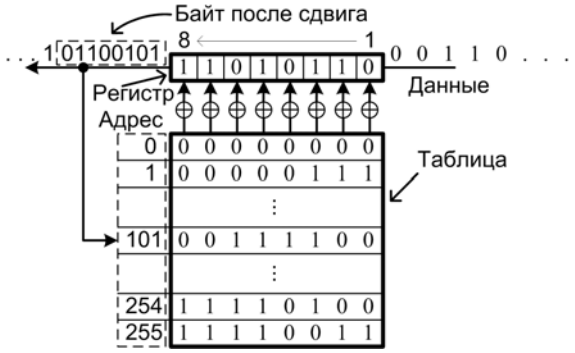

Рис. 2. Схематичное представление работы табличного алгоритма на примере CRC8 (x^8+x^2+x+1)

Словесно описать табличный алгоритм вычисления контрольной суммы CRC8 можно следующим образом.

Начало. Регистр (массив) 8 бит содержит нулевое значение, данные поступают в регистр начина со старшего разряда данных последовательным сдвигом побайтно.

Шаг 1. Сдвигаем данные в регистре на один байт от младших к старшему разряду, в регистр заносится новый байт из потока данных.

Шаг 2. Выдвинутый из регистра байт задаёт адрес в предварительно подготовленной таблице.

Шаг 3. Выбранный по заданному адресу из таблицы байт складывается по модулю два с регистром.

Шаг 4. Если ещё не все байты прошли через регистр из потока данных, то переходим на шаг 1.

Конец. В регистре содержится контрольная сумма CRC8.

Процесс вычисления и проверки контрольной суммы не отличается от обычного алгоритма. В табличном алгоритме вместо побитого сдвига каждую итерацию данные сдвигаются на байт. В связи с этим длина потока данных должна быть кратной 8 бит, т.е. байту.

Однако, вместо использования таблицы из 256 байт (для CRC8) можно обойтись матрицей в 8x8 бит (8 байт), которая легко реализуется на комбинационных схемах.

Матричный алгоритм

Процесс вычисления и проверки контрольной суммы CRC8 в матричном алгоритме осуществляется также как и в табличном, за исключением того, что вместо таблицы используется операция умножения вектора (выдвинутый байт) на матрицу (рис. 3). Матрица была специальным образом сформирована для кодового слова длиной 16 бит и образующего полинома x^8+x^2+x+1 [2].

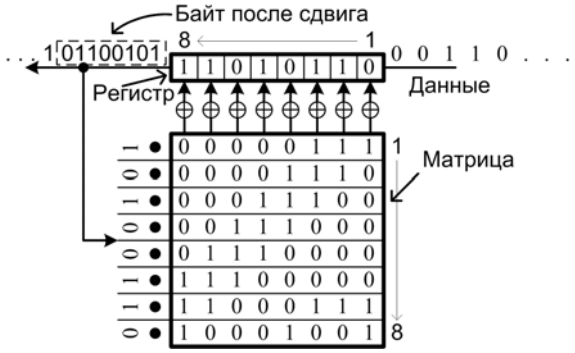

Рис. 3. Схематичное представление работы матричного алгоритма на примере CRC8 (x^8+x^2+x+1)

В заключении необходимо отметить, что в результате умножения вектора на матрицу получаем значение, идентичное содержимому таблицы по соответствующему адресу в табличном алгоритме. На нашем примере: 11000111 ⊕ 11100000 ⊕ 00011100 ⊕ 00000111 = 00111100.

Заключение

Матричный алгоритм вычисления контрольной суммы CRC позволяет обходиться без использования запоминающего устройства при аппаратной реализации. А при программной реализации требуемый объём памяти для хранения матрицы значительно меньше, чем требуется для хранения таблицы при использовании табличного алгоритма. Например, для CRC8 при вычислении по одному байту за итерацию матричному алгоритму потребуется 8 байт, табличному – 256 байт; при вычислении по два байта за итерацию матричному алгоритму потребуется 16 байт, табличному – 64 Кбайта; при вычислении по 4 байта за итерацию матричному – 32 байта, табличному – 4 Гб.

Литература

1. Ross N. Williams. A Painless Guide to CRC Error Detection Algorithms [Электронный ресурс]. – Режим доступа: http://www.ross.net/crc/download/crc_v3.txt, свободный.

2. Буркатовская Ю.Б., Мальчуков А.Н., Осокин А.Н. Быстродействующие алгоритмы деления полиномов в арифметике по модулю два. // Известия Томского политехнического университета. – Томск: изд-во ТПУ, 2006 – №1 – С. 19-24.