УДК 004.3

ИСПОЛЬЗОВАНИЕ РС-КОДОВ ДЛЯ ПОВЫШЕНИЯ НАДЕЖНОСТИ SMART-КАРТ

Андрианова Е.Г., доцент, E-mail: andrianova@mirea.ru
Крюков Д.А., ассистент, E-mail: dk@memfis.su
МГТУ МИРЭА, Москва, Россия

Аннотация. SIM-карты, банковские карты, транспортные карты, карты доступа к цифровому телевидению, электронные ключи доступа – все эти устройства удовлетворяют стандарту SMART-карт (ISO 7816). Они реализуют крайне важные функции и являются носителями исключительных данных владельца. Ценность информации в таких устройствах намного превышают их рыночную стоимость. Потеря или искажение данных являются критичными и не допустимы для них, поэтому задача повышения надежности хранения информации в устройствах является актуальной. В статье предлагается подход к повышению надежности данных устройств, путем внедрения РС-кодов коррекции ошибок для данных, хранящихся в постоянной памяти.
Ключевые слова: SMART-картf, РС-кодов коррекции ошибок, надежность носителей данных.

USE PC-CODE TO IMPROVE THE RELIABILITY SMART - CARDS

Andrianova E.G., assistant professor, E-mail: andrianova@mirea.ru
Krykov D.A., assistant, E-mail: dk@memfis.su
MSTU MIREA, Moscow, Russia

Abstract. SIM-card, credit cards, transport cards, access to digital television, electronic key access - all these devices meet the standard SMART-cards (ISO 7816). They implement a very important function and are carriers of exceptional data owner. The value of information in such devices is much higher than their market value. Loss or corruption of data are critical and are not allowed for them, so the task of improving the reliability of information storage devices is important. The paper proposes an approach to improve the reliability of these devices by implementing PC-error correcting codes for data stored in non-volatile memory.
Keywords: SMART-card, PS-error correction codes, reliable carrier data.

1. Введение

SMART-карты являются устройствами персональной идентификации, максимально приближенными к человеку (владельцу). Их широкое распространение объясняется ориентацией на хранение личных сведений и подтверждением полномочий владельца, поэтому потеря или искажение хранимой информации является критичным и недопустима для них. Препятствием к использованию «типовых» решений помехоустойчивого кодирования информации является условия ограниченного объема памяти и сравнительно небольшие вычислительные ресурсы SMART-карт [1]. Предполагается разработка замкнутой системы удовлетворяющей принципам преемственности и не требующей изменения соответствующих хост-компьютеров.

Основной задачей повышения надежности SMART-карт является обеспечение восстановления данных, хранившихся в вышедшей из строя странице ЭСППЗУ (электрически-стираемого программируемого постоянного запоминающего устройства), и сохранение их в другой рабочей странице памяти. Работу процедур кодирования/декодирования можно считать эффективной, если будут восстановлены данные хотя бы при первом сбое одной страницы памяти.

В соответствии с границей Синглтона, для того, чтобы система кодирования обладала способностью восстановления одной страницы памяти, необходимо не менее двух страниц с избыточной информацией. Одним из методов хранения данных для исправления пакетных ошибок является перемежение кода. Данный метод может быть адаптирован для нашей ситуации. А именно, в качестве кодируемых блоков данных можно рассматривать последовательности из i-ых символов кодируемых страниц памяти. Каждый символ представляет какую-то часть данных фиксированной длины. В частности, в качестве таковых могут рассматриваться биты или байты данных страницы, или вся страница может рассматриваться как один символ.

Полученные кодовые слова будут содержать в себе не более одной ошибки в случае возникновения сбоя одной из страниц. Поэтому для данных последовательностей данных достаточно будет использовать код, исправляющий одну ошибку. Примером таких кодов могут служить коды Хемминга и коды Рида-Соломона. Но прежде чем анализировать и сравнивать данные коды для рассматриваемой ситуации, проведем общий анализ предложенной методики кодирования.

С целью сохранения структуры файловой системы и файлов данных, наиболее разумным будет использование систематических кодов. Иными словами, для кодирования данных необходимо резервировать какое-то место в памяти для избыточной информации контроля, а часть памяти с данными оставлять неизменной. Договоримся в дальнейшем называть страницы части памяти с избыточной информацией страницами контроля, а станицы части памяти с данными так и называть страницами данных.

После перезаписи любой из страниц данных должен происходить пересчет контрольных символов кода и перезапись страниц контроля. Логичным будет предположить, что для большинства кодов число страниц контроля значительно меньше числа страниц данных. Это означает, что число циклов стирания/записи для страниц контроля растет значительно быстрее, чем для страниц данных. Наиболее вероятной является ситуация, что первой страницей памяти ЭСППЗУ, давшей сбой, станет страница контроля.

Вопросы повышения надежности путем применения кодов Хемминга были рассмотрены в [2], [3], поэтому использование РС-кодов будет рассмотрено в сравнении с ранее полученными результатами.

2. Использование РС-кодов для повышения надежности SMART-карт

Коды Рида-Соломона в отличие от кодов Хемминга, способны исправлять более одной ошибки и всегда удовлетворяют границе Синглтона. В случае кодов, исправляющих $t = 1$ ошибок (где t – корректирующая способность кода), также можно строить кодовые слова, беря по одному символу из каждой страницы памяти.

В случае произвольного значения параметра t наиболее целесообразно построение кодовых слов таким образом, чтобы из каждой страницы данных бралось символов, а контрольных символов таким же образом размещались по двум страницам контроля. Так как код исправляет t ошибок, любой сбой одной страницы памяти будет скорректирован.

Подобное использование кодов, исправляющих более одной ошибки, имеет неоспоримое преимущество в плане исправления ошибок, несвязанных со сбоем страниц памяти. В случае возникновения более чем одной ошибки в пределах одного кодового слова, такой код сможет выявить ее в отличие от кода, исправляющего одну ошибку.

С целью сохранения файловой структуры данных, так же как и в случае кодов Хемминга, наиболее целесообразно применение систематического кодирования по формуле [4]:

$$c(x) = i(x) - r(x)x^k. \quad (1)$$

При любом значении параметра t код Рида-Соломона будет иметь две страницы контроля. Причем, при изменении любого информационного символа будет следовать обязательное изменение всех контрольных символов, а следовательно и обеих страниц контроля. Таким образом, каждая страница контроля зависит от всех страниц данных, – величины зависимостей γ_0^C и γ_1^C (см [2], [3]) имеют значения $q - 3$.

Также как и в случае кодов Хемминга, длина кодовых слов может не совпадать с длиной кода Рида-Соломона, и требуемый код может быть построен путем операции укорочения. Получаемый таким образом код будем обозначать $C_{RS}(N, N - m)$, где $m = 2t$, N – число страниц памяти, m – количество страниц контроля. Производный код $C_{RS}(N, N - m)$ очевидно должен удовлетворять условию:

$$q - 1 \le Nt. \quad (2)$$

Заметим, что также как и для кодов Хемминга, $q = 2^l$, а параметр t должен быть таким, чтобы число -символов одной страницы памяти делилось на t. Так как размер страницы памяти равен обычно степени двойки, параметр t сам должен быть степенью двойки.

Для кодов Рида-Соломона также возможно изучение расхода памяти. Но в отличие от кодов Хемминга, здесь ожидаемый расход памяти для всех кодов будет одним и тем же, так как число страниц контроля всегда равно двум, и они всегда зависят от всех страниц данных. Более того, коды Рида-Соломона будут расходовать память также как код $C_H(N, N-M)$.

Более принципиальные различия между кодами Хемминга и Рида-Соломона проявляются в характеристиках алгоритмов кодирования и декодирования.

Для систематических кодов Рида-Соломона основную вычислительную сложность процедуры кодирования по формуле (1) составляет деление многочлена степени N на порождающий многочлен степени. Если рассматривать обычную процедуру деления многочленов, то для ее осуществления должно быть произведено ***2t(N-2t+1)*** операций сложения, и ***2t(N-2t+1)*** операций умножения в поле $GF(q)$.

Так же как и в случае кодов Хемминга, ускорение операции умножения в поле $GF(q)$ может быть достигнуто с помощью создания таблицы умножения. При этом операция умножения будет сводиться не более чем к l^2 операций сложения в поле $GF(q)$.

Таким образом, кодирование одного кодового слова будет составлять $\frac{l}{8}(2tl^2(N-2t+1)+2t(N-2t+1)) = \frac{lt}{4}(N-2t+1)(l^2+1)$ побайтовых операций сложения $\oplus$. Так как количество кодовых слов равно, где L – размер страницы памяти в битах, l – длинна кодового слова, суммарная трудоемкость вычислений при процедуре кодирования ***2L(N-2t+1)(ll+1)*** будет составлять операций сложения $\oplus$.

При использовании списков логарифмов элементов поля суммарная трудоемкость процедуры кодирования составит ***2L(N-2t+1)*** операций сложения $\oplus$ и $\frac{16L}{l(N-2t+1)}$ операций сложения чисел размером $\frac{l}{8}$ байт по модулю $q-1$.

Также как и для кодов Хемминга, для кодов Рида-Соломона аналогичным образом может быть организована сокращенная процедура кодирования, когда учитываются только обновляемые информационные символы.

Процедура декодирования состоит из нескольких этапов. Первый этап, –

вычисление компонент синдрома S_i, $(1 \le i \le 2t)$, состоит в вычислении значений кодовых многочленов в корнях порождающего многочлена $g(x)$. Так как работаем в поле характеристики 2, в общей сложности для всех кодовых слов должно быть вычислено значений S_i для нечетных значений i. Значения остальных синдромов S_i для четных i вычисляются путем возведения в квадрат синдромов с подходящим индексом.

Вычисление значения кодового многочлена степени $tN-1$ с помощью схемы Горнера составляет $tN-1$ операцию умножения и tN операций сложения в поле $GF(q)$. С использованием таблицы умножения это будет составлять $\frac{l}{8(l^2(tN-1)+tN)}$ операций сложения $\oplus$ и в общей сложности трудоемкость первого этапа декодирования составит $LNt(l^2+1)$ операций сложения $\oplus$. При использовании списка логарифмов элементов поля трудоемкость первого этапа составит LNt операций сложения $\oplus$ и операций модульного арифметического сложения чисел размером $\frac{l}{8}$ байт по модулю $q-1$.

Второй этап алгоритма, – вычисление коэффициентов многочлена локаторов ошибок $\Lambda(x)$, может осуществляться прямым методом Питерсона-Горенстейна-Цирлера, либо методом Берлекэмпа-Месси.

Прямое вычисление предполагает перебор веса ошибки от t до 1 до тех пор, пока матрица из уравнения (4) не станет невырожденной, после чего следует обращение матрицы.

$$\begin{pmatrix} S_1 & S_2 & \cdots & S_l \\ S_2 & S_3 & \cdots & S_{l+1} \\ \vdots & \vdots & \ddots & \vdots \\ S_l & S_{l+1} & \cdots & S_{2l-1} \end{pmatrix} \begin{pmatrix} \Lambda_l \\ \Lambda_{l-1} \\ \vdots \\ \Lambda_1 \end{pmatrix} = \begin{pmatrix} -S_{l+1} \\ -S_{l+2} \\ \vdots \\ -S_{2l} \end{pmatrix} \quad (4)$$

Для нахождения обратной матрицы необходимо выделение в оперативной памяти $\frac{lt^2}{8}$ байт для ее размещения, а потому при больших значениях параметра t размер матрицы может быть сопоставим с размером оперативной памяти SMART-карт.

Поиск веса ошибки и обращение матрицы могут быть совмещены путем применения алгоритма Гаусса для решения уравнения (4). При этом будет совершено не более $\frac{t(t-1)(t+3)}{2}$ операций сложения, $\frac{t(t-1)(t+1)}{2}$ операций умножения, $\frac{t(t+1)(t+2)}{2}$ операций деления.

При использовании таблицы умножения, операция деления может быть сведена к

$\frac{l(l-1)(l+3)}{2}$ операциям сложения в поле, или, по-другому, к $\frac{l^2(l-1)(l+3)}{16}$ побайтовых операций сложения $\oplus$. Таким образом, вычисление коэффициентов многочлена $\Lambda(x)$ равносильно выполнению не более чем $\frac{l}{16}\left((t+1)l\left(\frac{(t+2)}{2}(l^2-3)+(t^2+2)l\right)+t(t-1)(t+3)\right)$ побайтовых операций $\oplus$ Суммарная вычислительная сложность второго этапа составит $\frac{L}{2t}\left((t+1)l\left(\frac{(t+2)}{2}(l^2-3)+(t^2+2)l\right)+t(t-1)(t+3)\right)$ операций $\oplus$.

Использование списка логарифмов позволяет сократить число операций по выполнению второго этапа до $\frac{(t-1)(t+3)}{2}L$ операций $\oplus$ а также операций арифметического сложения чисел размером $\frac{l}{8}$ байт по модулю $q-1$.

Так как характеристика поля $GF(q)$ равна 2, алгоритм Берлекэмпа-Месси будет состоять из t итераций. При каждой *i*-ой итерации должно быть произведено не более одной операции деления, операций умножения и операций сложения в поле $GF(q)$ при вычислении приближения многочлена , и не более операций умножения и операций сложения при вычислении невязки.

При использовании таблиц умножения трудоемкость алгоритма Берлекэмпа-Месси будет составлять не более $\frac{lt}{8}\left((t+1)(l^2+1)+\frac{l(l-1)(l+3)}{2}\right)$ операций сложения $\oplus$ для одного кодового слова, и не более $L\left((t+1)(l^2+1)+\frac{l(l-1)(l+3)}{2}\right)$ операций $\oplus$ для второго этапа в целом.

При использовании списка логарифмов вычисления второго этапа декодирования будут составлять не более $L(t+1)$ операций сложения $\oplus$ и операций модульного арифметического сложения.

Третий этап алгоритма, – поиск локаторов ошибок путем нахождения корней многочлена локаторов ошибок. Поиск корней осуществляется с помощью процедуры Чени, состоящей в последовательном вычислении значений многочлена при подстановке в него α^i, где α примитивный элемент поля, использованный при построении кода. Вычисления каждого значения многочлена сводятся не более чем к t операциям умножения и $t-1$ операции сложения в поле. Так как декодируем укороченный код Рида-Соломона, перебираться должны не все степени элемента α, а

только от 0 до $tN-1$. При использовании таблицы умножения данная процедура будет занимать не более $\frac{tNl}{8(tl^2+t-1)}$ побайтовых операций сложения $\oplus$ для каждого кодового слова, и не более $LN(tl^2+t-1)$ операций сложения $\oplus$ для всего этапа. Применение списков логарифмов снижает вычислительную сложность третьего этапа до $LN(t-1)$ операций сложения $\oplus$ и операций арифметического сложения по модулю $q-1$.

Четвертый этап, – поиск значений ошибок, может быть реализован либо прямым методом решения системы линейных уравнений, либо с помощью процедуры Форни.

При прямом методе решения, всю вычислительную сложность составляет обращение матрицы размерности не более $t \times t$, что равносильно выполнению не более $\frac{t(t-1)(t+3)}{2}$ операций сложения, $\frac{t(t-1)(t+1)}{2}$ операций умножения, $\frac{(t+1)(t+2)}{2}$ операций деления. Таким образом, вычислительная сложность четвертого этапа будет составлять при использовании таблиц умножения не более $\frac{L}{2t}\left((t+1)l\left(\frac{(t+2)}{2}(l^2-3)+(t^2+2)l\right)+t(t-1)(t+3)\right)$ операций $\oplus$, а при использовании списка логарифмов – не более $\frac{(t-1)(t+3)}{2}L$ операций $\oplus$ а также операций арифметического сложения чисел размером $\frac{l}{8}$ байт по модулю $q-1$.

Алгоритм Форни состоит из процедуры умножения двух многочленов степеней не более t и , вычисления производной многочлена, нахождения значений двух многочленов в точках и их частного. Всего это составляет не более t операций деления, операций умножения, операций сложения в поле для одного кодового слова.

Вычислительная сложность четвертого этапа с применением алгоритма Форни с использованием таблицы умножения составит $\frac{L}{t}\left(t+\frac{7t^2+5t-4}{2}l^2+\frac{7t^2+3t-4}{2}l(l-1)(l+3)\right)$ операций сложения $\oplus$, при применении списка логарифмов – $\frac{L}{2t(7t^2+3t-4)}$ побайтовых операций сложения $\oplus$ и арифметических операций сложения по модулю $q-1$.

Заметим, что при декодировании обязательным является только первый этап вычисления компонент синдрома. В случае если они оказываются все равны нулю, дальнейшее декодирование не производится.

Алгоритм решения может быть существенно упрощен в случае $t = 1$. Поскольку декодирование сводится к решению системы нелинейных уравнений (5), достаточно заметить, что в этом случае уравнения

$$\begin{aligned} S_1 &= X_1 Y_1 + \cdots + X_l Y_l \\ &\vdots \\ S_{2t} &= X_1^{2t} Y_1 + \cdots + X_l^{2t} Y_l \end{aligned} \qquad (5)$$

приобретают вид:

$$\begin{aligned} S_1 &= Y_1 X_1 \\ S_2 &= Y_1 X_1^2 \end{aligned}.$$

Тогда локатор и величина ошибки могут быть легко найдены по формулам:

$$X_1 = \frac{S_2}{S_1}$$

$$X_2 = \frac{S_1^2}{S_1}.$$

Таким образом, при $t = 1$ второй, третий и четвертый этап равносильны выполнению $\frac{16L}{l}$ операций деления и операции умножения в поле $GF(q)$, что при использовании таблиц будет составлять $Ll(l^2 + 3l - 3)$ операций $\oplus$, а при использовании списка логарифмов элементов – $\frac{24L}{l}$ операций арифметического сложения по модулю $q - 1$.

Несмотря на упрощение процедуры декодирования кода Рида-Соломона при $t = 1$, процедура полного декодирования кода Хемминга все равно оказывается как проще в реализации, так и менее трудоемкой. При этом расход памяти и корректирующие свойства этих кодов в целом совпадают. Единственное преимущество кодов Рида-Соломона над кодами Хемминга может быть получено за счет увеличения значения параметра t.

Из представленного анализа алгоритма декодирования можно сделать следующие выводы:

1. Наиболее целесообразным с точки зрения быстродействия и расхода оперативной памяти является применение во втором этапе декодирования алгоритма Берлекэмпа-Месси, в четвертом этапе процедуры Форни;

2. Сложность декодирования зависит линейно от размеров страницы памяти L, количества N кодируемых страниц памяти и параметра t. Заметим, что сложность процедуры кодирования при увеличении параметра t уменьшается в отличие от процедуры декодирования. В случае использования таблиц умножения сложность

декодирования обладает кубической зависимостью от параметра l , при использовании же списков логарифмов элементов сложность декодирования практически не зависит от l ;

3. Применение списка логарифмов элементов поля обеспечивает значительное повышение скорости алгоритма по сравнению с использованием таблицы умножения, но создает большой расход памяти ПЗУ, и потому применение данного способа оптимизации возможно только при параметре $l \leq 8$;

4. Применение кодов Рида-Соломона целесообразно только при значениях параметра $t > 1$.

Если при $l \leq 8$ процедуры кодирования и декодирования могут быть сделаны сравнительно быстрыми за счет применения списков логарифмирования, то уже при больших значениях l их вычислительная сложность сильно возрастает. С другой стороны, длина кода Рида-Соломона при $l = 8$ ограничена 255 символами, а в реальной ситуации требуется кодирование большего числа страниц памяти. Кроме того, увеличение параметра t также требует увеличения длины кодовых слов при фиксированном числе страниц памяти.

Ускорения выполнения процедур кодирования и декодирования можно добиться за счет аппаратной реализации наиболее трудоемких участков процедур.

Наибольшую часть сложных вычислительных операций в процедурах кодирования и декодирования составляют операции умножения в поле $GF(q)$. В значительно меньшем количестве встречаются операции деления, но при этом они оказываются самыми трудоемкими. Поле $GF(q)$ имеет характеристику 2, а потому операции умножения и деления должны иметь сравнительно простую схемную реализацию, и могут быть реализованы в микроконтроллерах SMART-карт в командах сопроцессора.

3. Заключение

Из проведенного анализа применимости РС-кодов в SMART-картах, непосредственно следует, что параметр l следует выбирать равным степени двойки и не больше 16 (Для обычных размеров памяти ЭСППЗУ этого оказывается более чем достаточно). Схемные реализации операций умножения и обращения для полей $GF(2^l)$ при данных ограничениях параметра l достаточно хорошо исследованы, и могут быть реализованы в АЛУ микроконтроллера SMART-карты [5], [6], [7].

Заметим, что помимо аппаратной реализации арифметики конечных полей, для ускорения выполнения этих операций могут быть использованы также таблицы

умножения или списки логарифмов элементов поля, размещаемые в памяти ПЗУ или ЭСППЗУ.

Для SMART-карт с микросхемой памяти процедуры кодирования и декодирования могут быть реализованы только аппаратным образом. Кроме того, менее приемлемым с точки зрения безопасности и эффективности, но также допустимым, является вариант полной или частичной реализации указанных процедур на хост-компьютере.

Для микропроцессорных SMART-карт наилучшим вариантом является программная реализация процедур кодирования и декодирования с аппаратной реализацией арифметики конечных полей $GF(2^l)$ для соответствующих значений l. Более простым, но менее эффективным является вариант программной реализации арифметики конечных полей $GF(2^l)$ путем применения таблиц умножения и списков логарифмов элементов поля. Наибольшим недостатком такого подхода является фактическая невозможность реализации в памяти SMART-карты списка логарифмов элементов поля при $l = 16$.

Как было указано ранее, для кодирования разных информационных блоков в пределах одной SMART-карты могут быть использованы коды Хемминга и Рида-Соломона с различными параметрами. Увеличение разнообразия используемых помехоустойчивых кодов, в общем случае, приводит к усложнению их совместной реализации. Причем усложняется как реализация арифметики полей $GF(2^l)$, если используются различные значения l, так и реализация непосредственно процедур кодирования и декодирования.

Резюмируя, обозначим предлагаемые аппаратно-программные модификации SMART-карт, обеспечивающие эффективное выполнение процедур ввода-вывода: Аппаратная реализация арифметики конечных полей $GF(2^l)$ для $l = 2,4,8,16$ или для некоторых из указанных значений l. Данная модификация микроконтроллеров SMART-карт даст наибольший эффект ускорения работы процедур кодирования и декодирования;

Альтернативой аппаратной реализации арифметики полей $GF(2^l)$ является создание в памяти ПЗУ и ЭСППЗУ таблиц умножения или списков логарифмов элементов поля;

Программно-аппаратная реализация функции обнаружения сбоев в страницах памяти ЭСППЗУ. Данная модификация позволит использовать в качестве помехоустойчивого кода продольный контроль избыточности, что позволит максимально упростить процедуры ввода-вывода, а также существенно сократить размер контрольной памяти.

Список литературы:

1. Андрианова Е.Г., Крюков Д.А., Петров А.Б. Повышение надежности хранения информации в микрокомпьютерных комплексах персональной идентификации. Труды МАИ. 2012. № 59. С. 18.

2. Крюков Д.А. Помехоустойчивое кодирование в персональных устройствах идентификации. Журнал Промышленные АСУ и контроллеры. – М.: Научтехлитиздат, вып. 12, 2012, стр. 50-58.

3. Крюков Д.А. Модели функционирования персональных систем идентификации с процедурами помехоустойчивого кодирования и декодирования хранимых данных. – М.: электронное научно-техническое издание «Наука и образование: научное издание МГТУ им. Н.Э. Баумана», №10, октябрь 2012.

4. Морелос-Сарагоса Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение – М.: Техносфера, 2005.

5. Юрьев И.В., Дяченко О.Н. Определение оптимальных параметров кодов Рида-Соломона // Інформатика та комп’ютерні технології 24-26 стр. Донецк, ДонНТУ. – 2009. – С. 82-88.

6. Дяченко О.Н. Аппаратная реализация и корректирующие возможности кодов Рида-Соломона// Наукові праці Донецького національного технічного університету. Випуск: 6 (127) - Донецьк: ДонНТУ. - 2007. – С.113-121.

7. Берлекемп Э.Р. Алгебраическая теория кодирования - М.: Мир, 1971.