УДК 681.3.07

О.П. Малофей

Ставропольский военный институт связи РВ, Россия

АЛГОРИТМ ЗАПИСИ И СЧИТЫВАНИЯ ИНФОРМАЦИИ В МАЖОРИТАРНО РЕЗЕРВИРОВАННОЙ ПАМЯТИ

Предлагается подход к созданию структурного кода, на основе которого разработан способ записи и считывания данных в запоминающем устройстве, сокращающий в полтора раза объем задействованной памяти и, следовательно, улучшающий её надёжностные, массогабаритные и энергоёмкостные показатели. Синтезированный код гарантированно исправляет модульные ошибки и применим для контроля работоспособности элементов памяти. Представлены также отказоустойчивые структуры памяти для 8, 16 и 32-х разрядных форматов представления данных.

запись и считывание информации, мажоритарно резервированная память, структурный код

Введение

Рост объема массивов данных и интеграции элементной базы с одной стороны и уменьшение времени обращения к запоминающему устройству (ЗУ) с другой стороны накладывают жесткие рамки на вероятностные характеристики, тракта «записи – хранение – воспроизведение» информации в автоматизированной системе управления (АСУ). При этом очевидно, что для ЗУ имеют место как случайные неповторяющиеся сбои (ошибки), так и систематические ошибки различной кратности. Причем интенсивность сбоев интегральных схем (ИС) ЗУ на один – два порядка выше, чем интенсивность отказов [1]. Борьба с данными ошибками наиболее сложна и разделяется на методы аппаратного и программного контроля. Из всего многообразия методов борьбы с ошибками в ЗУ для АСУ, не допускающей факта наличия катастрофической ошибки, широкое применение нашел метод мажоритарной обработки хранимой информации.

Очевидно, что при известной избыточности данного метода он полностью оправдывает себя с точки зрения исправления ошибок [2], однако его применение в ЗУ АСУ связано со следующими трудностями:

– при N = (2m – 1)-кратном повторении сообщения и m > 2 сложность второй решающей схемы, реализующей мажоритарный метод повышения достоверности (МПД), возрастает [3];

– мажоритарный метод аппаратурного резервирования абсолютно неприемлем для контроля технического состояния ЗУ, так как при исправлении ошибки невозможно определить место ее нахождения.

Решение данной проблемы возможно, если использовать подход создания структурного кода, удовлетворяющего следующим условиям: Он должен надежно защищать каждый байт информации (машинного слова) от сбоев и групповых ошибок длины до j = 8. Код должен иметь избыточность, существенно меньшую, чем троирование. Процедуры кодирования и декодирования не должны требовать дополнительного тактирования и вносить задержку во временную диаграмму «запись – считывание». Код должен быть приемлем для контроля технического состояния ЗУ.

Алгоритм кодирования

Вход: машинное слово, состоящее из двух байтов $W = A|B$, где $A = (a_1, a_2, \ldots, a_8)$; $B = (b_1, b_2, \ldots, b_8)$.

Выход: слово памяти из четырех байтов, являющееся кодовым вектором кода V

$$V = A|B|C|D.$$

Метод: для байтов входного машинного слова AIB и матрицы P размера 8 × 8 вида

$$P = \begin{bmatrix} \vdots\, I \\ Y \vdots \cdots\cdots \\ \vdots\, X \end{bmatrix},$$

где I – единичная матрица размера 7 × 7; Y – вектор-столбец из единиц; X – вектор-строка из нулей, определим проверочные байты C и D как

$$C = A \oplus B; \quad D = A \oplus BP, \qquad (1)$$

где $BP = (b_0, b_1, \ldots, b_7)$; $b_0 = \sum_{i=1}^{8} b_i \bmod 2$.

Алгоритм декодирования

Вход: слово памяти из четырех байтов, возможно, содержащее ошибки

$$V^* = A^* \left| B^* \right| C^* \left| D^* \right. .$$

Выход: машинное слово из двух байтов с исправленными ошибками $W = A \,|\, B$ или сигнал отказа от декодирования q = 1.

Метод: полагая q = 0 (т.е. процедура декодирования выполняется), вычислим синдром ошибки $S = S_1 \,|\, S_2$, где

$$S_1 = A^* \oplus B^* \oplus C^*; \; S_2 = A^* \oplus B^* P \oplus D^*. \qquad (2)$$

Рассмотрим возможные значения синдромов при возникновении ошибок в различных байтах, а также

докажем, что предложенный алгоритм удовлетворяет исходному условию 2, т.е. код исправляет любой одиночный модуль ошибки длиной j = 8 в слове памяти.

Доказательство: положим:

$A^* = A \oplus e_a$; $B^* = B \oplus e_b$; $C^* = C \oplus e_c$; $D^* = D \oplus e_d$,

где e_x – модуль ошибки в байте X, возможно, имеющей место.

Тогда из (1) и (2) следует:

$$S_1 = e_a \oplus e_b \oplus e_c;\ S_2 = e_a \oplus e_b P \oplus e_d. \quad (3)$$

Анализ значения синдромов S_1 и S_2 дает следующие ситуации:

1. Если $e_a = e_b = e_c = e_d = 0$, то $S_1 = S_2 = 0$, что соответствует отсутствию ошибок.

2. Если $e_a = e_b = e_c = 0$, а $e_d \neq 0$, то $S_1 = 0$; $S_2 \neq 0$, что соответствует ошибке в байте D. Исправление имеющейся ошибки e_d имеет вид:

$$D = D^* \oplus S_2. \quad (4)$$

3. Если $e_a = e_b = e_d = 0$, а $e_c \neq 0$, то $S_1 \neq 0$; $S_2 = 0$, что соответствует ошибке в байте C. Исправление имеющейся ошибки e_c имеет вид:

$$C = C^* \oplus S_1. \quad (5)$$

4. Если, $e_b = e_c = e_d = 0$, а $e_b \neq 0$, то $S_1 = S_2 \neq 0$, что соответствует ошибке в байте A. Исправление имеющейся ошибки e_a имеет вид

$$A = A^* \oplus S_1. \quad (6)$$

5. Если $e_a = e_c = e_d = 0$, а $e_b \neq 0$, то $S_1 = e_b$; $S_2 = e_b P$, то есть $S_1 P = S_2 \neq 0$, что соответствует ошибке в байте B. Исправление имеющейся ошибки e_b имеет вид

$$B = B^* \oplus S_1. \quad (7)$$

6. Все остальные случаи ведут к отказу от декодирования ($q \geq 1$), то есть вызваны отказом более одного корпуса микросхем памяти ЗУ, при этом групповыми ошибками поражены два и более байта слова памяти.

Таким образом, случаи 1...5 охватывают все возможные виды ошибок в одном модуле слова памяти. Для завершения доказательства осталось лишь показать, что при одиночном модуле ошибок ситуации по п.п. 4 и 5 не могут иметь место одновременно, то есть нужно показать, что X = XP тогда и только тогда, когда X = 0.

Пусть $X = (x_1, x_2, x_3, ..., x_8)$, тогда $XP = (x_0, x_1, x_2, ..., x_7)$, где $x_0 = \sum_{i=1}^{8} x_i \bmod 2$.

Если $(x_1, x_2, x_3, \ldots x_8) = (x_0, x_1, x_2, \ldots x_7)$, то $x_0 = x_1 = x_2 = \ldots = x_7 = x_8$.

Это возможно лишь при $x_i = 0$, так как при $x_i = 1$ сумма $x_0 = 0$, где $i = \overline{1,8}$.

Определим избыточность полученного кода как отношение информационных символов к сумме информационных и проверочных:

$$\xi = \frac{|A| + |B|}{|A| + |B| + |C| + |D|} = \frac{16}{32} = 0{,}5,$$

где $|A| = |B| = |C| = |D|$ – длина информационного байта.

Рассмотрим теперь предложенный код и процедуру декодирования с точки зрения дополнительных условий, заданных вначале:

1. Код защиты двухбайтового слова имеет избыточность 16 бит (0,5) и исправляет модуль ошибки длины 8, согласно [3] такой код удовлетворяет условию минимальной избыточности.

Проведем сравнение известного и предложенного алгоритмов записи по величине объема задействованной памяти.

Для выполнения трижды дублированной записи информационного слова размера n двоичных символов известный способ требовал наличия памяти объемом $П_1 = 3n$. Предложенный способ имеет избыточность 0,5, то есть для записи информационного слова той же длины потребуется объем памяти $П_2 = 2n$, при этом сохранится возможность мажоритарной обработки информации. Сокращение ресурса задействованной памяти составляет

$$\xi = \frac{П_1}{П_2} = \frac{3n}{2n} = 1{,}5.$$

2. Как было доказано выше, декодер исправляет сбои и ошибки в каждом машинном слове, независимо от их характера и причин возникновения [4].

3. Схема кодера и декодера строится из стандартных логических элементов без памяти и поэтому не нуждается в тактировании, а вносимая ими задержка не превышает время переключения данных элементов.

4. Оценка частоты возникновения ненулевых синдромов S_1 и S_2 позволяет делать вывод о случайности или систематичности возникновения ошибок, что применимо для контроля технического состояния элементов (модулей) ЗУ.

Следовательно, синтезированный код удовлетворяет заданным нами условиям и применим для исправления случайных и систематических групповых ошибок, возникающих в ЗУ модульного типа, а также для технического контроля элементов данных ЗУ в резервированных системах памяти [5], что позволяет достичь сокращения задействованного объема ЗУ в 1,5 раза и во столько же раз улучшить его надежностные и массо-габаритные показатели.

Структурная схема реализации предложенного алгоритма приведена на рис. 1.

Рассмотренная отказоустойчивая структура для 8-разрядных ЭВМ строится из модулей памяти вида I × NK. Однако постоянный рост скорости обработки массивов информации требует ее дальнейшего распараллеливания, т.е. применения 16- и 32-разрядных шин и соответствующего формата представления данных. Выполним математическое описание предложенного способа и построим отказоустойчивую структуру памяти для 16- и 32-разрядных ЭВМ из модулей вида 4 × NK [6].

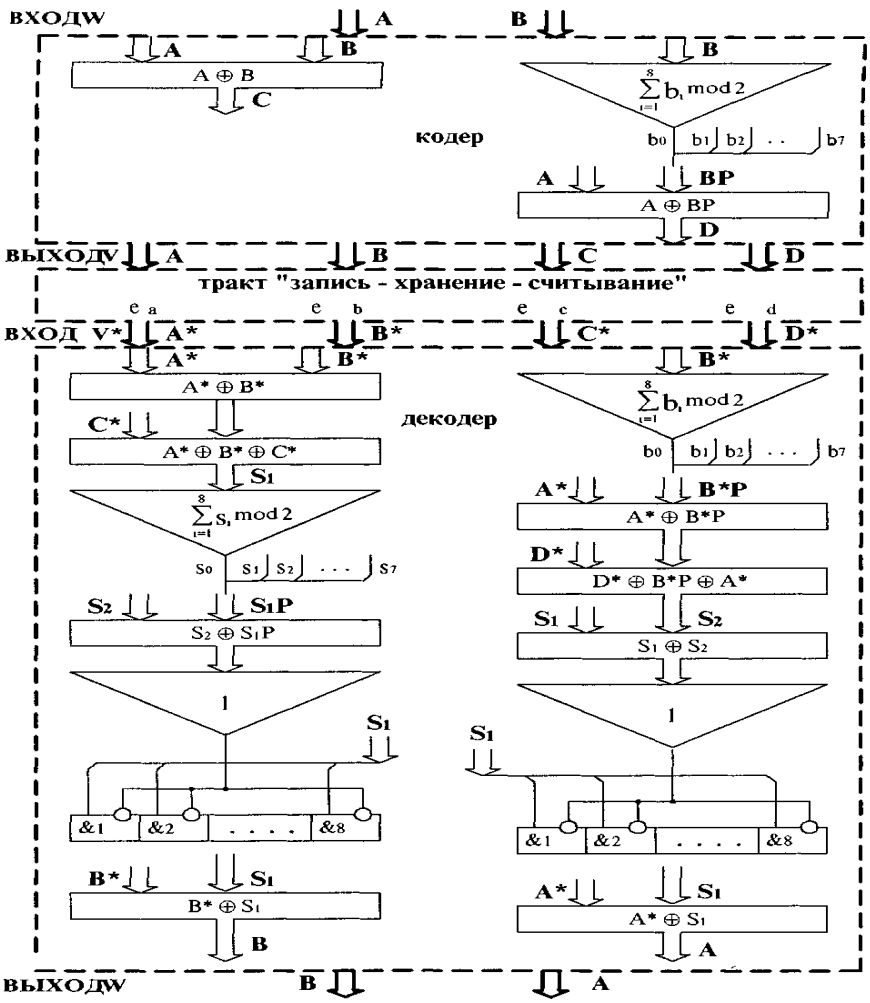

Рис. 1. Способ записи и считывания информации с исправлением модульных ошибок

Отказоустойчивую структуру будем представлять четверкой неотрицательных чисел (ν, N, m, d), где ν – разрядность модуля памяти; N – число модулей памяти, покрывающее слово избыточной структуры; m – число модулей, покрывающее машинное слово; d – допустимое число отказов модулей. В этих обозначениях (1, 16, 16, 0) и (1, 32, 16, 1) представляют безубыточную структуру и структуру с кодом Хэмминга из модулей вида I × NK, а четверка (ν, 3m, m, 1) – троированную структуру из модулей вида ν × NK. Модульный код для отказоустойчивой структуры памяти должен также удовлетворять четырем условиям, указанным ранее.

Введем обозначения: a – машинное слово, вектор над GF [3]; $|a|$ – длина вектора a; wt(a) – вес Хэмминга вектора a; $a = |a_0|a_1|..|a_{m-1}|$ – представление вектора a в виде последовательности подблоков длиныν; (a)(i) – циклический сдвиг вектора a на i разрядов вправо; $a^{>}$ – вектор длины $|a|-1$, полученный из вектора a удалением крайней правой координаты.

Определим код V как множество векторов вида

$$V = |a_0|a_1|...|a_{m-1}|c_0|c_1| \ , \qquad (8)$$

где a_i – подблоки машинного слова; c_i – проверочные подблоки.

$$c_0 = \sum_{i=0}^{m-1} a_i \ ; \ c_1 = \left(\sum_{i=0}^{m-1} \left(|a_i| \, 0 \, | \right)(i) \right)^{>} . \qquad (9)$$

Соотношения (8) и (9) задают процедуру кодирования в систематической форме для кода V с параметрами (ν(m + 2), νm). При считывании информации из памяти на вход декодера поступает слово

$$V \oplus e = |a_0 \oplus e_0|a_1 \oplus e_1|...|a_{m-1} \oplus e_{m-1}|c_0 \oplus e_m|c_1 \oplus e_{m+1}| , \qquad (10)$$

где e – вектор ошибки, который в наших предположениях содержит ровно один ненулевой подблок e_i. По (10) вычисляется вектор синдрома ошибки:

$$\begin{cases} S = |S_0|S_1|; \\ S_0 = c_0 \oplus e_m \oplus \sum_{i=0}^{m-1} (a_i \oplus e_i) = e_m \oplus \sum_{i=0}^{m-1} e_i; \\ S_1 = c_1 \oplus e_{m+1} \oplus \left(\sum_{i=0}^{m-1} \left(|a_i \oplus e_i| 0 | \right)(i) \right)^{>} = \\ = e_{m+1} \oplus \left(\sum_{i=0}^{m-1} \left(|e_i| 0 | \right)(i) \right)^{>} . \end{cases} \qquad (11)$$

Из (11) следует, что:

$$S_0 = \begin{cases} e_m, & \text{если } e_m \neq 0; \\ 0, & \text{если } e_{m+1} \neq 0; \\ e_i, & \text{если } e_i \neq 0,\ 0 \le i \le m; \end{cases} \quad (12)$$

$$S_1 = \begin{cases} 0, & \text{если } e_m \neq 0; \\ e_{m+1}, & \text{если } e_{m+1} \neq 0; \\ \left(\left(|e_i|0|\right)(i)\right)^{>}, & \text{если } e_i \neq 0,\ 0 \le i \le m. \end{cases} \quad (13)$$

Таким образом, если S_0 или S_1 равны нулю, то информационные подблоки слова V не содержат ошибок. При S_0, $S_1 \neq 0$, S_0 дает значение модуля ошибки, а S_1 позволяет найти номер i подблока, в котором эта ошибка произошла. Число i может быть определено перебором, как решение уравнения:

$$S_1 = \left(\left(|S_0|0|\right)(i)\right)^{>}; \qquad i = \overline{0, m-1}. \quad (14)$$

Рассмотрим условия, при которых (14) имеет единственное решение.

Теорема: если $\nu + 1$ простое число и $m \le \nu + 1$, то уравнение (14) имеет единственное решение, отличное от нулевого.

Доказательство:

1. Для того, чтобы уравнение x = (x)(i) имело бы решение, необходимо, чтобы x был периодической последовательностью с длиной периода, равной общему делителю чисел $\nu + 1$ и i, но при простом $\nu + 1$ имеется только тривиальный делитель – 1. Имеются две последовательности, состоящие из последовательностей периода 1: x = (0 0 ... 0) – соответствует отсутствию ошибок; x = (1 1 ... 1) – такой вид вектора не допускается по построению (см. (14)).

2. $wt(x - (x)(i)) \ge 2$, так как x и (x)(i) одинаковые.

3. $wt(x^* - ((x)(i))^*) \ge 1$. Соотношения 1, 2 и 3 выполняются при $0 < i < \nu$, что и требовалось доказать.

Для 16-разрядной ЭВМ и модулей памяти вида $4 \times NK$ код V непосредственно приводит к структуре (4, 6, 4, 1) [6]. Для 32-разрядной ЭВМ непосредственное использование кода V невозможно, так как m = 8, что больше $\nu + 1$.

Определим код W как множество

$$W = |a_0|a_1|..|a_7|c_0|c_1|c_2|,$$

где

$$c_0 = \sum_{i=0}^{7} a_i;\ c_1 = \left(\sum_{i=0}^{3}\left(|a_i|0|\right)(i)\right)^{>};$$

$$c_2 = \left(\sum_{i=0}^{3}\left(|a_{4+i}|0|\right)(i)\right)^{>}. \quad (15)$$

Вектор синдрома ошибки $S = |S_0|S_1|S_2|$, при этом

$$\begin{cases} S = |e_8|0|0|, & \text{если } e_8 \neq 0; \\ S = |0|e_9|0|, & \text{если } e_9 \neq 0; \\ S = |0|0|e_{10}|, & \text{если } e_{10} \neq 0; \\ S = |e_i|\left(\left(|e_i|0|\right)(i)\right)^{>}|0|, & \text{если } e_i \neq 0;\ 0 \le i \le 3; \\ S = |e_i|0|\left(\left(|e_i|0|\right)(i)\right)^{>}, & \text{если } e_i \neq 0;\ 4 \le i \le 7. \end{cases} \quad (16)$$

Таким образом, для 32-разрядных ЭВМ и модулей памяти вида $4 \times NK$ код W приводит к структуре (4, 11, 8, 1) [6].

Осталось лишь показать, что предложенный способ допускает реализацию мажоритарной обработки информации при декодировании, для чего выполним с учетом (1) следующие действия:

$$\begin{cases} A = A_1; \\ C \oplus B = A \oplus B \oplus B = A_2; \\ D \oplus BP = A \oplus BP \oplus BP = A_3; \end{cases} \quad (17)$$

$$\begin{cases} B = B_1; \\ C \oplus A = A \oplus B \oplus A = B_2; \\ D \oplus A = A \oplus BP \oplus A = B_3P. \end{cases} \quad (18)$$

Следовательно, по значениям декодируемых слов кодового вектора $W = |A|B|C|D$ каждое из исходных слов A и B может быть трижды восстановлено независимыми путями, после чего возможна их мажоритарная обработка по методу "два из трех".

Выводы

Предложен подход к созданию структурного кода, на основе которого разработан способ записи и считывания данных в запоминающем устройстве, сокращающий в полтора раза объем задействованной памяти и, следовательно, улучшающий в соответствующее число раз её надёжностные, массогабаритные и энергоёмкостные показатели. Синтезированный код гарантированно исправляет модульные ошибки и применим для контроля работоспособности элементов памяти. Представлены также отказоустойчивые структуры памяти для 8, 16 и 32-х разрядных форматов представления данных

Список литературы

1. Бородин Г.А., Иванов В.А. Методы расчета надежности запоминающих устройств // Зарубежная радиоэлектроника. – 1989. – № 2. – С. 92-111.

2. Саганович Ю.Л. Кодовая защита оперативной памяти ЭВМ от ошибок // Автоматика и телемеханика. – 1991. – № 5. – С. 3-45.

3. Бояринов Н.М. Помехоустойчивое кодирование числовой информации. – М.: Наука, 1983. – 348 с.

4. АС СССР № 256418. Способ записи и считывания данных на информационном носителе / А.В. Камыш, О.П. Малофей, С.В. Кузнецов и др., 1987.

5. Баринов В.В., Березин А.С., Вернер В.Д. и др. Сверхбольшие интегральные микросхемы оперативных запоминающих устройств. – М.: Радио и связь, 1991. – 294 с.

6. Камыш А.В., Ткаченко А.В. Отказоустойчивая структура памяти для 8-, 16- и 32-разрядных ЭВМ // Сб. тезисов III НТК КВВКИУ РВ, 19-20 сентября 1995 г. –

Краснодар, 1996. – С. 144-146.

Поступила в редакцию 29.05.2006

Рецензент: д-р техн. наук, проф. Ю.В. Стасев, Харьковский университет Воздушных Сил им. И. Кожедуба, Харьков.