Как видно из графиков на рис. 1–2, трехмерный итеративный код, как со стандартной, так и с усеченной проверочной матрицей, при значениях k от 2 до 8 является совершенным (кривая граничных соотношений данного кода расположена между границами Хэмминга и Варшамова–Гилберта). Трехмерный итеративный код с диагональными проверками в случае стандартной матрицы является совершенным при значениях k от 2 до 4, а в случае усеченной матрицы – при значениях k от 2 до 8. Хотя при остальных значения k с точки зрения избыточности, как трехмерный итеративный код, так и трехмерный итеративный код с диагональными проверками не являются совершенными, но у них есть иное достоинство – структура кодов близка к архитектуре полупроводниковой памяти.

ЛИТЕРАТУРА

1. Мак-Вильямс Ф., Слоэн Н. Теория кодов, исправляющих ошибки / Пер. с англ. под ред. Л.А. Басалыго.– М.:Связь, 1979. –746 с.

2. Урбанович П.П., Романенко Д.М. Свойства и алгоритмы аппаратной реализации нового вида итеративных кодов для систем памяти / Новые информационные технологии: Третья Международная конференция NITE'2000.– Мн.: БГЭУ, 2000. Т. 2 – С. 159–164.

3. Fuja T., Heegard C., Goodman R. Liner sum codes for random acces memories // IEEE Transaction on computers. Vol.37. No.9, September 1988.– P.1030–1041.

4. Майоров С.А., Урбанович П.П. Декодирование кодов, основанных на двойном прямом произведении кодов на четность // Научно-техническая конференция профессорско-преподавательского состава, посвященная 30-летию БГУИР: Тез. докл.– Минск, 1994.–354 с.

5. Петерсон У., Уэлдон Э. Коды, исправляющие ошибки // Москва: Мир, 1976. – 321 с.

УДК 681.3.07

С.Б. Макась, ассистент

ПРИМЕНЕНИЕ АЛГОРИТМОВ МАТРИЧНЫХ ИГР ПРИ ПОСТРОЕНИИ СТАТИСТИЧЕСКОЙ МОДЕЛИ НАДЕЖНОСТИ ПОЛУПРОВОДНИКОВОЙ ПАМЯТИ С АППАРТНОЙ ИЗБЫТОЧНОСТЬЮ

The matrix game algorithm for creating a statistical model of DRAM reliability described. It examines the selection of optimal method for error logging and handling as well as simple and fast analysis for the case of hardware memory fault when the next simulated error occurs.

Введение

В настоящее время существует достаточно широкий круг программных алгоритмов, реализующих статистические модели надежности полупроводниковой памяти. Эти алгоритмы различаются как по уровню сложности, принципу построения, так и по объему решаемой задачи. Большинство из них в своей реализации либо совсем не учитывают применение технологий повышения надежности хранения данных, либо реализуют одну из них. Это ставит актуальной проблему создания динамически компонуемых моделей, поддерживающих одновременно несколько способов повышения надежности полупроводниковой памяти. Как следствие, особое внимание нужно уделить процессу оптимального выбора конкретной стратегии резервирования в соответствии с текущим событием из потока ошибок.

Представление совокупности событий в модели

Программная модель надежности ДОЗУ оперирует набором данных о типах элементов ДОЗУ, количестве элементов одного типа, значении параметров элемента, размере внесенной избыточности.

Для достижения достаточной точности моделирования необходимо подробно указывать тип, подтип элемента устройства, номер и один из его параметров. Для оптимизации выбора типа сбойного элемента удобно сгруппировать все ошибки по типам элементов моделируемого устройства. Это удобно сделать при помощи древовидного списка. В этом случае все ошибки делятся на группы, специфичные для каждого блока устройства. Каждая группа представлена узлом, который может быть началом ветви узлов ошибок, специфических для данной группы. Рассмотрим это на конкретном примере.

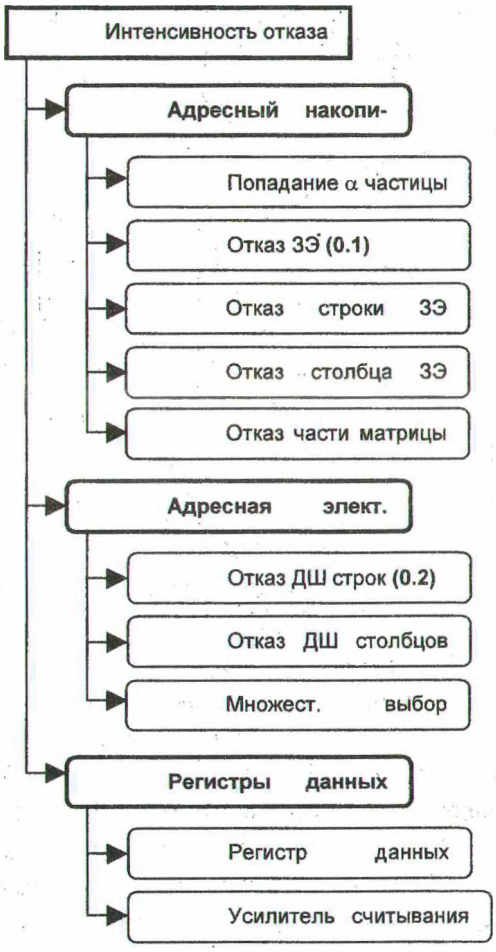

Рис. 1

Основное моделируемое событие – полный отказ моделируемого устройства. Данное событие имеет определенную интенсивность $\lambda_{ЗУ}$ – интенсивность отказа запоминающего устройства (ЗУ). Согласно законам статистического моделирования, интенсивность отказа ЗУ складывается из суммарных интенсивностей отказов каждого типа элементов $\lambda_{\Sigma i}$ $(i = 1..n)$, где n – количество различных типов элементов памяти. Другими словами, сумма интенсивностей отказов каждого типа в процентном соотношении к интенсивности отказов ЗУ должна быть равна 1. Исходя из этого, сумма интенсивностей отказов, выраженная в процентах для каждого узла одной ветви, должна быть равна 1. Другими словами, если узел имеет свою ветвь, то сумма узлов этой ветви должна быть равна 1.

Рассмотрим это подробнее, на рис. 1, где изображена двухуровневая древовидная схема представления полной группы событий, специфических для данного моделируемого устройства.

Алгоритм моделирования отдельного события довольно прост и заключается в следующем. Моделирование начинается с выбора узла корневой ветви. После получения номера группы ошибок данный узел проверяется на наличие собственной ветви событий, для которой он будет считаться суммарной интенсивностью ошибок. Если полученный узел не имеет своей ветви событий, то фиксируется событие, определяемое данным узлом.

При регистрации события указываются две вероятности: вероятность события в группе, а так же вероятность события в общей совокупности. Вторая из вероятностей

равна $P_{ij} = \lambda_i * \lambda_j$, где i – номер корневой группы событий; j – номер события в i-й группе.

В случае двухуровневого дерева ошибок для относительных интенсивностей всех событий должно выполняться следующее равенство:

$$\sum_{i=1}^{n}\left(\lambda_i \cdot \sum_{j=1}^{m_i}\lambda_{ij}\right) = 1,$$

где n – количество групп событий (количество узлов в корне дерева), в данном примере $n = 3$; λ_i – интенсивность i-й группы событий; λ_{ij} – интенсивность события j в i-й группе; m_i – количество событий в i-й группе.

Представление полной группы событий в виде дерева, сгруппированного по типам элементов моделируемого ДОЗУ, представляет простой и удобный способ структуризации ошибок по типам, а также возможность подробного подсчета событий по окончании процесса моделирования.

Алгоритмы матричных игр в обработке событий

Обработка смоделированного отказа или сбоя представляет собой алгоритм применения внесенной избыточности при обнаружении ошибки. Программа, реализующая модель, должна попытаться исправить ошибку имеющимися у нее в наличии средствами резервирования, т.н. стратегиями повышения надежности. Определенная модель имеет собственный набор стратегий резервирования. Объем и тип аппаратной избыточности указываются при задании элементарной части моделируемого устройства.

Исходя из этого, стоит вопрос о корректном и оптимальном выборе стратегии повышения надежности по отношению к определенной группе событий. Наиболее оптимальным способом диспетчеризации между фиксацией события и его обработкой будет использование алгоритмов матричных игр, предназначенных для оптимизации принятия решения в условиях неопределенности [2].

Для этого будем использовать алгоритм игры с несознательным противником (игры с природой). Он представляет собой математическую модель игры, когда система точно не знает, с каким событием столкнется в данный момент времени. В этом случае должны быть известны вероятности каждого события (стратегии противника) при условии, что полная совокупность вероятностей состояний природы подчиняется равенству [2]:

$$S_Q = \sum_{j=1}^{n} Q_j = 1.$$

Применение этого типа матричных игр накладывает одну особенность: более высокий выигрыш (успешность применения стратегии системой) в данный момент времени обеспечивается за счет более выгодного состояния природы.

Применительно к статистической модели надежности в качестве совокупности стратегий противника будет выступать полная группа событий, определенная для данной модели. В качестве стратегий противодействия будет выступать аппаратная информационная или структурная избыточность. Определим конфигурацию моделируемого ЗУ, а также объем и тип внесения аппаратной избыточности. В этом случае рассматривается 16Mbit чип с организацией 4Mbit X 4 и типом регенерации 4K. Это зна-

чит, что все четыре матрицы запоминающих элементов имеют по $N_{Rows} = 4000$ строк и $N_{Cols} = 1000$ столбцов.

Таблица 1

A_1	Контроль четности (Parity)	Фиксирует однобитные ошибки в информационных словах, ошибка исправляется попыткой перезаписи информации
A_2	L – резервных строк (0.5% от N_{Rows})	Включение резервной строки в случае отказа основной в адресном накопителе или отказе элемента памяти.
A_3	Резервный ДШ строк	Включение резервного дешифратора в случае отказа основного или в случае сбоев типа «залипание» и «множественная выборка»
A_4	Резервный ДШ столбцов	

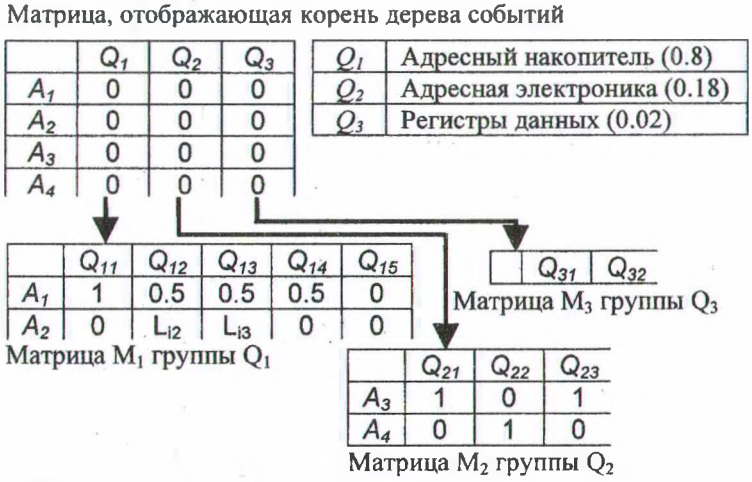

Рис. 2

Рассмотрим использование алгоритма игры с природой применительно к совокупности событий, описанных в предыдущем пункте. Ввиду того, что совокупность событий представлена в виде древовидного списка, традиционная матрица игры должна будет претерпеть некоторые изменения. Рассмотрим это подробнее на рис. 2.

В отличие от традиционных алгоритмов, в данном случае количество столбцов корневой матрицы будет определяться количеством групп событий (количеством узлов в корне дерева событий). Если данный узел имеет собственную ветвь событий, то весь его столбец заполняется нулями. Это будет признаком того, что данный столбец имеет свою уточняющую матрицу, к которой и нужно обратиться за выбором стратегии, предварительно повторив процедуру моделирования события уже в пределах выбранной группы. В случае, когда корневой узел не имеет собственной ветви событий, соответствующий столбец заполняется коэффициентами (выигрышами), которые оценивают количественно возможность применения одной из определенных стратегий резервирования по отношению к событию, моделируемому данным узлом.

Как видно из рис. 2, матрица каждой группы событий имеет различное число строк, в которых представлены различные стратегии резервирования. Это обусловлено тем, что нет смысла перечислять для каждой группы событий все стратегии ввиду того, что некоторые из них имеют нулевую эффективность по отношению ко всем событиям группы. Вследствие чего в игровой матрице получаются нулевые строки, что приводит к вырождению матрицы. Чтобы этого избежать, исключаются из каждой матрицы группы событий нулевые строки. Это не запрещается теорией матричных игр, не накладывающей ограничений на количественное соотношение строк и столбцов.

Рис. 3

Отдельного внимания заслуживает набор правил для определения коэффициента эффективности стратегии по отношению к каждому событию группы:

1) если данная стратегия A_i никак не может противодействовать определенному событию Q_j, то значение коэффициента в позиции a_{ij} равно 0;

2) если данная стратегия A_i может противодействовать событию Q_j, тогда ее коэффициент эффективности a_{ij} определяется количественным выражением объема внесенного резервирования, выраженного в процентах к единице;

3) в случае применения стратегии A_i в ответ на событие Q_j необходимо проводить пересчет значения коэффициента эффективности стратегии. Это необходимо делать потому, что резервный элемент вводится вместо отказавшего основного, повторно уже не может выступать с той же эффективностью против этого же события Q_j Пересчет осуществляется в процентном отношении от первоначального количества;

Например, в адресном накопителе осуществляется внесение избыточности на уровне $L < N_{Rows}$ резервных строк (скользящее резервирование). Принимая L за единицу (100%), определим начальную величину коэффициента эффективности стратегии A_2 по отношению к событиям Q_{12} и Q_{13} (см. рис. 2, матрица M_2). В случае события Q_{12} или Q_{13} стратегия A_2 их однозначно нейтрализует, но при этом количество резервных строк уменьшится и соответственно уменьшится эффективность стратегии, выраженная коэффициентами L_{i2} и L_{i3} на соответствующую величину: $\Delta L = L - L_R$, где L_R — количество строк, взятых из резерва. Соответственно коэффициенты L_{i2} и L_{i3} уменьшатся и станут уже не 1 (100%), а $L_{next} = \dfrac{L_{prev} - L_R}{L_{prev}}$. Данный пересчет нужно осуществить для всех столбцов в строке A_2.

Из данного примера видно, что эффективности стратегий резервирования не постоянны со временем, а уменьшаются пропорционально количеству введенных в работу резервных элементов. Пример зависимости уменьшения эффективности стратегии скользящего резервирования, использующего $L = 20$ резервных строк при условии, что в момент события Q_{13} (см. рис. 2 и рис. 1) отказывает только одна строка и заменяется одной резервной при условии, что введенная в работу резервная строка не отказывает.

Из диаграммы видно, что данная стратегия резервирования, как и всякая другая, имеет свой запас эффективности. В случае обращения коэффициента эффективности

стратегии в 0 из матрицы удаляется вся строка, и последующее событие группы Q_j вызовет полный отказ устройства памяти при условии, что в строках других стратегий матрицы группы нет ненулевых коэффициентов эффективности.

Заключение

Описанный способ применения алгоритмов матричных игр дает эффективный механизм связи генерируемого потока событий со способами их обработки. Данный алгоритм не дает однозначного ответа насчет того, что именно данная стратегия A_i однозначно исправит очередную смоделированную неисправность. В результате работы алгоритма осуществляется выбор наиболее предпочтительной стратегии обработки события, а также достаточно простой и быстрый анализ на предмет возможности отказа запоминающего устройства при наступлении очередного события.

Как сказано выше, алгоритм матричных игр не определят конкретно то, что именно эта стратегия нейтрализует очередное событие. После осуществления выбора стратегии запускается реализующий ее программный алгоритм, который, в отличие от матричных алгоритмов, оперирует конкретными аппаратными характеристиками и адресом неисправности. В дальнейшем, на основании результатов работы алгоритма выносится заключение о том, исправлена ошибка или нет.

ЛИТЕРАТУРА

1. Харин Ю.С. Основы имитационного и статистического моделирования. Мн., 1997.

2. Варфоломеев В. И., Воробьев С.Н. Принятие управленческих решений. М.: Кудиц-образ, 2001.

3. Макась С.Б. Имитационная модель надежности динамического оперативного запоминающего устройства // Труды БГТУ. Серия физ.-мат. наук и информат. Вып. X. Мн., 2002.

УДК 681.325.3

Н.В. Пацей, ассистент

ПРИНЦИП ПОСТРОЕНИЯ КОНКАТЕНТНОГО ПАРАЛЛЕЛЕПИПЕДНОГО ТУРБО-КОДА

The paper introduces the new concatenated parallelepiped-coding scheme in the area of turbo codes. Phase recovery algorithm within the maximum a posteriori probability decoding iterations are considered in detail. The qualities of the code are demonstrated by comparing performance and implementation issues.

Наиболее заметным достижением в теории помехоустойчивого кодирования за последнее десятилетие, безусловно, является изобретение турбо-кодов. Турбо-коды – класс кодов с широким диапазоном гибкости за счет использования объединенного SISO (Soft-Input-Soft-Output) алгоритма декодирования с практически оптимальной производительностью [1]. Турбо-коды относятся к классу параллельных каскадных кодов.

Классический турбо-код представляет собой систематический код, в котором проверочная группа образуется из проверочных битов, генерируемых двумя кодерами составных рекурсивных сверточных кодов (РСК), причем информационная последовательность подается в кодер первого РСК (РСК1) непосредственно, а в кодер второго РСК (РСК2) — через устройство псевдослучайного перемежения [2–4]. Сочетание та-