ПРОГРАММНОЕ СРЕДСТВО ДЛЯ АНАЛИЗА МАРШЕВЫХ ТЕСТОВ

Петровская В.В., Деменковец Д.В.,

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Ярмолик В.Н. – д.т.н., профессор

В работе рассматриваются основные модели неисправностей запоминающих устройств. Механизмы тестирования ЗУ на основе маршевых тестов и программное средство для анализа маршевых тестов.

Полупроводниковая память является основополагающим и важным сегментом рынка полупроводников. Микросхемы памяти необходимы практически во всех электронных продуктах. Согласно данным World Semiconductor Trade Statistics [1], в 2020 году мировые продажи памяти составили самую большую долю любого вспомогательного продукта – 27% от общего рынка. Результаты исследований показывают, что отказы оперативной памяти (ОЗУ) составляют до 70% от общего числа отказов вычислительных систем. Причиной неисправных состояний является наличие физических или механических дефектов, либо множества подобных дефектов [2].

Поскольку механизм тестирования основан на сравнении логического поведения неисправной памяти с исправной памятью, физические дефекты представляют в виде математических моделей неисправностей [3]. Неисправности классифицируют по количеству участвующих в них ячейках памяти. К одиночным моделям неисправностей относятся константные (stuck-at faults – SAF) и переходные неисправности (transition faults – TF), к неисправностям, в которых участвуют несколько ячеек памяти – неисправности взаимного влияния (coupling faults – CF) и кодочувствительные неисправности (pattern sensitive faults – PSF) [4,5].

В настоящее время для тестирования запоминающих элементов (ЗУ), как массива памяти используются маршевые тесты. Преимущества маршевых тестов заключаются в двух фактах. Во-первых, покрытие известных моделей неисправностей может быть математически доказано, хотя нельзя иметь никакого представления о связи между моделями и физическими дефектами реальных микросхем. Во-вторых, время тестирования для маршевых тестов обычно линейно зависит от размера памяти, что делает их приемлемыми с промышленной точки зрения [6].

Классический маршевый тест состоит из маршевых элементов, которые последовательно применяются к каждой ячейке памяти. В начале маршевого элемента указывается порядок следования адресов, в состав элемента входят операции чтения и записи. Например, маршевый тест MATS+ {⇑⇓(w0), ⇑(r0, w1), ⇓(r1, w0)} состоит из трех маршевых элементов [4]. Первый записывает в каждую ячейку памяти ноль, знак ⇑⇓ показывает порядок адресов, убывающий или возрастающий. Второй маршевый элемент считывает из ячейки памяти ноль и записывает единицу, адресная последовательность при этом будет возрастающая (⇑). Последний элемент в маршевом тесте MATS+ считывает единицу и записывает ноль во все ячейки, порядок адресов убывающий (⇓).

С помощью разработанных алгоритмов маршевые тесты преобразуются в неразрушающие и могут быть запущены во время работы системы без потери исходных данных. В неразрушающих версиях маршевых тестов операции чтения/записи нуля и единицы преобразуются в операции чтения/записи прямого или инверсного содержимого ячейки. Маршевый тест TMATS+, преобразованный по методу Николаидиса, состоит из начального {⇑(rd); ⇑⇓(rd*)} и базового {⇑(rd, wd*); ⇑⇓(rd*, wd)} тестов. Результаты всех операций чтения сжимаются в сигнатуру. Начальный тест формирует эталонную сигнатуру, а в результате выполнения базового теста получают реальную сигнатуру. Неравенство сигнатур сигнализирует о наличии в памяти дефектов [4,7].

С целью автоматизации проведения и анализа маршевых тестов было разработано программное средство. Приложение позволяет моделировать и тестировать неисправности в массиве ЗУ. На основе полученных с помощью программы результатов тестирования проводится анализ эффективности маршевых тестов.

В разработанной программе реализовано классическое и неразрушающее тестирование по методу Николаидиса, а также новый метод неразрушающего тестирования, основанный на базе четных адресных последовательностей [8]. Программное средство позволяет моделировать одиночные неисправности, неисправности взаимного влияния и кодочувствительные неисправности. В качестве порядка адресов используется счетчиковая, псевдослучайная последовательности и код Грея. Приложение поддерживает многократное тестирование с изменяемыми адресными последовательностями. Изменение последовательностей адресов при многократном тестировании происходит с применением масок.

Вначале работы программного средства пользователь выбирает тип тестирования, размер памяти, вид неисправности, название маршевого теста и адресную последовательность. При необходимости пользователь может записать и выполнить новый маршевый тест. В случае многократного тестирования необходимо выбрать количество тестов и механизм изменения адресной последовательности, для неразрушающего тестирования дополнительно выбирается степень сигнатурного анализатора. После проверки корректности введенных данных инициализируются все необходимые объекты-фабрики, счетчики и списки, формируется конфигурация выбранного типа неисправности. Далее создается массив ячеек ЗУ, маршевый тест и выбираются виды тестируемых неисправностей. В случае неразрушающего тестирования инициализируются сигнатурные анализаторы для исходного и конечного содержимого памяти. В цикле по количеству тестируемых неисправностей выполняется маршевый тест. Если было обнаружено несоответствие ожидаемого и прочитанного значения текущая неисправность добавляется в список обнаруженных. Если тест неразрушающий, то сравниваются эталонные и реальные сигнатуры. Итерации продолжаются, пока не будут проверены все тестируемые неисправности.

Сложные модели неисправностей, в которых участвует две и более ячейки памяти, реализованы в программе с помощью реактивных расширений. На рисунке 1 представлена UML диаграмма, отражающая связи между классами ячейки памяти (Cell) и сложными неисправностями (CouplingFault, NeighborhoodPatternSensitiveFault). Когда содержимое ячейки изменяется, в неисправность, в которой она участвует, приходит уведомление. Если переход состояния ячейки является условием активизации данной неисправности, то обработчик уведомления нарушает содержимое зависимых ячеек. Рассмотренный подход отражает идею поведенческого шаблона проектирования «наблюдатель» (Observer).

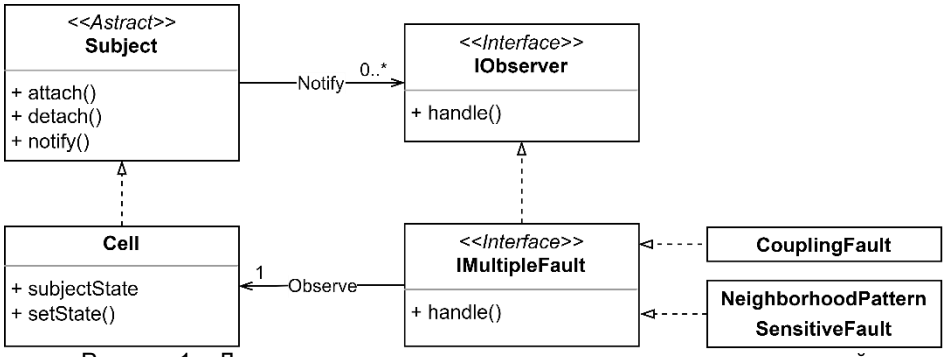

Рисунок 1 – Диаграмма классов, отражающая механизм уведомлений

Эффективность маршевого теста зависит от его покрывающей способности и времени, которое требуется для его выполнения. Покрывающая способность теста – это отношение числа обнаруженных неисправностей к количеству всех возможных неисправностей. В таблице 1 представлена оценка покрытия неисправностей тестами MATS+ и March Y, полученные в результате работы программы.

Таблица 1 – Оценка покрытия неисправностей тестами MATS+ и March Y

Тест	Сложность	Неисправность				
		SAF	TF↑	TF↓	CFin	CFid
MATS+	5N	100%	100%	0%	75%	37,5%
March Y	8N	100%	100%	100%	100%	50%

Полученные результаты аналогичны представленным ранее в работах по тестированию ОЗУ, что свидетельствует о корректности алгоритма программного средства. Наиболее актуальными являются исследования эффективности неразрушающих тестов с четным повторением адресов, в частности, их способности покрывать кодочувствительные неисправности.

Список использованных источников:

1.World Semiconductor Trade Statistics [Электронный ресурс] – Режим доступа: https://www.wsts.org – Дата доступа: 21.02.2022.

2. Селедец, В.Н. Программное средство сравнительной оценки маршевых тестов ОЗУ / В.Н. Селедец, В.А. Леванцевич // Компьютерные системы и сети: 55-я юбилейная научная конференция аспирантов, магистрантов и студентов, Минск, 22-26 апреля 2019 г. / Белорусский государственный университет информатики и радиоэлектроники. – Минск, 2019. – С. 152–153.

3. Anumol, A. Detection of Faults in SRAM Using Transient Current Testing / A. Anumol, K. Kumar // IOSR journal of VLSI and Signal Processing. – 2013. – № 2(1). – P. 21–26.

4. Ярмолик, С.В. Маршевые тесты для самотестирования ОЗУ: монография / С.В. Ярмолик, А.П. Занкович, А.А. Иванюк; под общ. ред. В.Н. Ярмолика. – Минск: Изд. центр БГУ, 2009. – 271 с.

5. Ярмолик, В.Н. Анализ и синтез маршевых тестов запоминающих устройств / В.Н. Ярмолик, В.А. Леванцевич, Д.В. Деменковец // Цифровая трансформация – 2021. – № 2. – С. 45–55.

6. Hamdioui, S. Testing Embedded Memories: A Survey / S. Hamdioui // International Doctoral Workshop on Mathematical and Engineering Methods in Computer Science. – 2013. – P. 32–42.

7. Mrozek, I. MATS+ transparent memory test for Pattern Sensitive Fault detection / I. Mrozek, V.N. Yarmolik // 15th International Conference on Mixed Design of Integrated Circuits and Systems / Poznan, Poland – 2008. – P. 493–498.

8. Неразрушающее тестирование запоминающих устройств на базе двойных адресных последовательностей / В.Н. Ярмолик [и др.] // Доклады БГУИР. – 2021. – 19(4). – С. 43–51.