УДК 004.33.54

ПРОГРАММНОЕ СРЕДСТВО МОДЕЛИРОВАНИЯ СВЯЗАННЫХ НЕИСПРАВНОСТЕЙ НА ОСНОВЕ НЕИСПРАВНОСТЕЙ ВЗАИМНОГО ВЛИЯНИЯ

Тюшев Т.А., студент гр.951003, Деменковец Д.В., аспирант

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

Леванцевич В.А. – ст. преподаватель

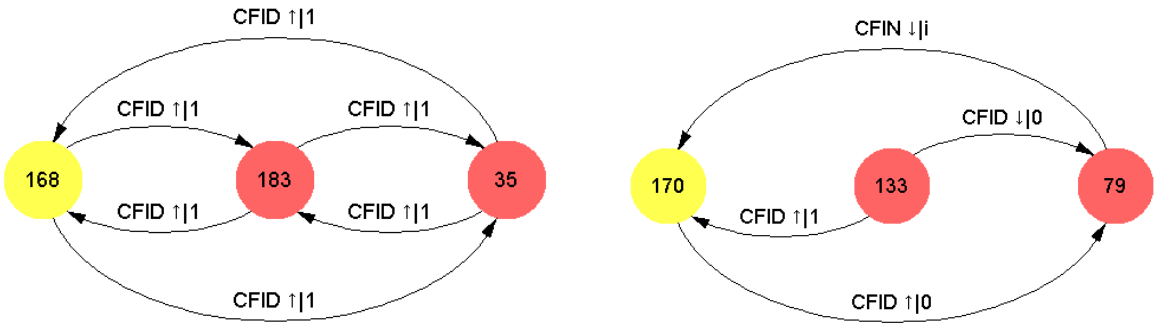
Аннотация. В данной работе рассматривается архитектура симулятора неисправностей запоминающих устройств, ориентированного на неисправности взаимного влияния трёх и более ячеек памяти. Описывается модель и нотация связанной неисправности, состоящей из трех ячеек. Представлен алгоритм генерации неисправностей и внесения их в модель памяти. Описывается программное средство, моделирующее процесс тестирования модели памяти с помощью маршевых тестов.

Ключевые слова. ОЗУ, запоминающие устройства, тестирование ОЗУ, маршевые тесты, неисправности взаимного влияния, программное средство.

В настоящее время ОЗУ является неотъемлемой частью большинства современных вычислительных систем. Совершенствование технологий процесса изготовления элементов ОЗУ позволяют повышать степень интеграции, но также приводит к появлению большего количества дефектов. Это обусловлено увеличением плотности размещения элементов и уменьшением техпроцесса. Поэтому на сегодняшний день остается актуальной задача тестирования памяти [1].

Среди моделей неисправности запоминающих устройств связные неисправности занимают одно из особых мест в силу сложности их обнаружения. При этом возможны различные типы и подтипы неисправностей, участвующих в конкретной связной неисправности. В данных неисправностях часто участвуют более двух ячеек памяти. Их способность влиять на поведение друг друга приводит к маскированию неисправности и делает разработку алгоритма тестирования очень сложной задачей [2].

Для реализации симулятора необходимо создать модель взаимодействия неисправных ячеек, обеспечивающую взаимное влияние на несколько других ячеек. Например, при неисправности трёх ячеек может быть ситуация, когда каждая из них влияет на поведение остальных двух. Для визуализации неисправности используется граф, на котором изображены три ячейки памяти и стрелками показано влияние ячеек друг на друга. Над стрелками показан тип неисправности. Пример графа такой неисправности представлен на рисунке 1 (a).



a b

Рисунок 1 – Графы неисправностей взаимного влияния для трёх ячеек

Для удобного ввода данных об неисправностях используется нотация, описывающая неисправности взаимного влияния нескольких ячеек. Примером описания связанной неисправности для трёх ячеек может быть следующая строка: ***<-,↑,1,i><↑,↓,-,-><↓,-,0,0>.*** Модель неисправности, соответствующая данной нотации представлена на рисунке 1 (b).

Для программной реализации поведения данных неисправностей необходимо, чтобы каждая ячейка поддерживала список ячеек, на которые она, как агрессор, влияет (массив жертв). Также необходимо хранить и тип ошибки каждой жертвы (тип влияния на ту или иную ячейку). Каждая ячейка поддерживает набор методов для тестирования её поведения, которые соответствуют стандартным операциям с памятью – r0 (чтение 0), r1 (чтение 1), w0 (запись 0), w1 (запись 1).

Ячейка жертва является отдельной сущностью инкапсулирует в себе ячейку памяти и тип неисправности. Диаграмма классов модели памяти представлена на рисунке 2 (a).

Основой модели всех неисправностей является интерфейс Fault. Для связных неисправностей предусмотрен отдельный интерфейс LinkedFault, который реализует методы, оповещающие ячейку жертву, что произошла определённая операция для ячейки агрессора. Диаграмма классов модели неисправностей представлена на рисунке 2 (b).

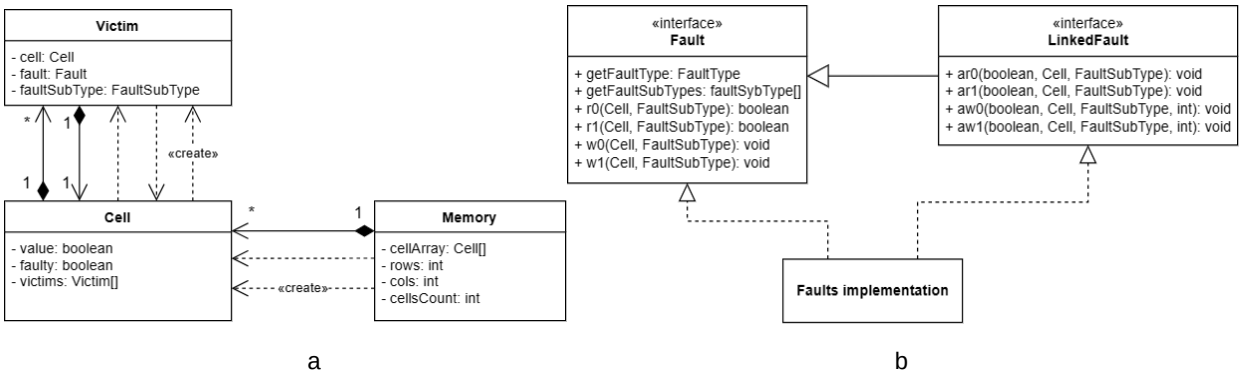


Рисунок 2 – Диаграмма классов программного средства

Таким образом, объект неисправности «следит» за поведением ячейки жертвы, тем самым является «наблюдателем». При этом происходит вызов определенных методов, когда выполняется какая-либо операция с ячейкой, у которой имеются ячейки жертвы. Реализация данных методов содержит логику неисправностей и изменяет состояние ячеек жертв в соответствии с типом неисправности и операцией над агрессором.

Каждый метод вызывающий изменение состояния ячейки жертвы содержит защиту от рекурсивного вызова для случая, когда две ячейки влияют друг на друга и могут бесконечно инвертировать значение друг друга. Данное решение позволяет смоделировать неисправности взаимного влияния любой сложности с любым количеством ячеек.

Для проверки алгоритмов тестирования в программном средстве предусмотрено два способа внесения неисправностей в модель памяти: с помощью текстового файла и случайной генерации.

Текстовый файл нотации содержит адреса ячеек, и тип неисправности в нотации, описанной выше. Пример записи в таком файле:

LR3<149,-,↓,-,i><113,-,↓,-,-><5,↓,-,0,1>

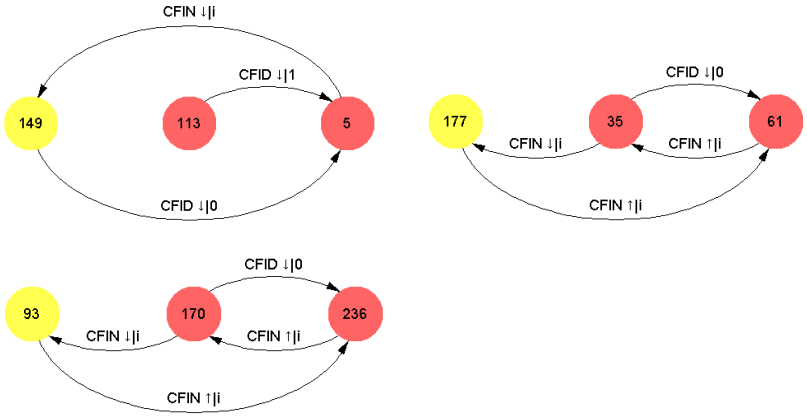
Пример генерации неисправности из файла данной нотации представлен на рисунке 3 (a).

Другой способ позволяет по заданному типу неисправности случайно сгенерировать нужное количество неисправностей. Для этого вводится тип неисправности в нотации, описанной выше и количество неисправностей. Для генерации адресов неисправных ячеек используется случайная последовательность неповторяющийся чисел.

Пример случайной генерации неисправностей:

- Размер памяти 16x16 ячеек
- Ввод типа неисправности: <-,↑,i,-><↓,↓,-,i><-,↑,i,0>;
- Ввод количества неисправностей: 2;
- Генерация последовательности адресов: 177, 35, 61, 93, 170, 236;
- Внесение неисправностей в модель памяти.

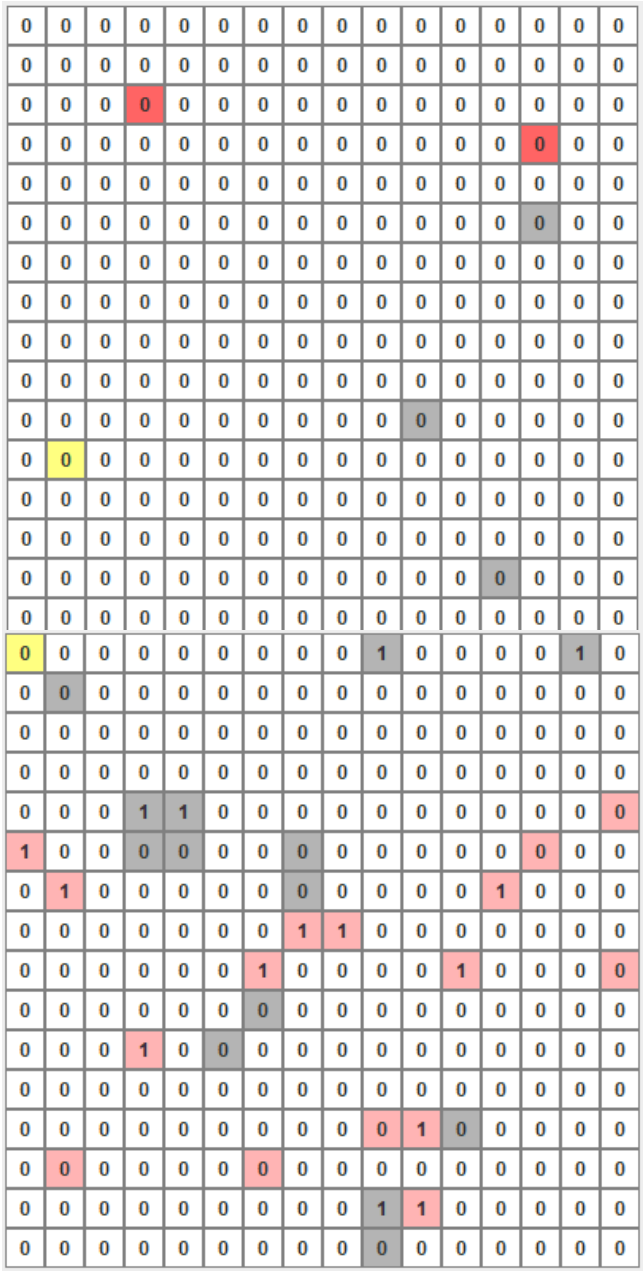
Результат генерации неисправностей случайным образом представлен на рисунке 3 (b) и (c).



a b c

Рисунок 3 – Примеры сгенерированных неисправностей

Модель памяти после внесения случайно сгенерированных неисправностей представлена на рисунке 4 (a). Результат тестирования модели памяти с обнаруженными и необнаруженными неисправностями на рисунке 4 (b).



a b

Рисунок 4 – Примеры моделей памяти

Стоит отметить, что данная модель отражает логические адреса ячеек запоминающего устройства, а не физические и не отражает расположение ячеек на кристалле микросхемы памяти. Поэтому ячейки, влияющие друг на друга могут быть сколь угодно удалены и иметь любые логические адреса [2].

Для тестирования модели памяти используются тесты линейной длины семейства «Марш» (March tests). В качестве последовательностей адресов ячеек памяти для маршевых тестов были реализованы различные псевдослучайные и квазислучайные последовательности (счетчиковая, М-последовательность, Холтона, Корпута, Грея) [3-4].

Процесс тестирования отображается в реальном времени с помощью графического интерфейса. При этом разными цветами отображаются ячейки с обнаруженными и необнаруженными неисправностями.

В разработанном программном средстве реализована функция сохранения модели памяти в файл и загрузки из файла. При этом происходит сериализация и десериализация объекта памяти, с сохранением состояния ячеек.

В результате было разработано программное средство, позволяющее моделировать память запоминающего устройства. ПС позволяет генерировать различные виды связанных неисправностей различными способами. Также позволяет выполнять тестирование этой модели при помощи маршевых тестов с использованием различных адресных последовательностей. Позволяет визуально оценивать эффективность работы тестирования.

Список использованных источников:

1. Функциональное тестирование микросхем ОЗУ // Радиоавтоматика URL: https://radioautomatic.ru/news_and_articles/article/137/ (дата обращения: 20.03.2021).

2. Ярмолик, В. Н. Контроль и диагностика вычислительных систем: [монография] / В. Н. Ярмолик. – Минск: Бестпринт, 2019. – 387 с: ил. 75.

3. Петровская, В. В. Программное средство для анализа маршевых тестов / Петровская В. В., Деменковец Д. В. // Компьютерные системы и сети: сборник статей 58-й научной конференции аспирантов, магистрантов и студентов, Минск, 18–22 апреля 2022 г. / Белорусский государственный университет информатики и радиоэлектроники. – Минск, 2022. – С. 59–61.

4. Деменковец, Д. В. Программное средство моделирования и поиска неисправностей запоминающих устройств / Деменковец Д. В. // Компьютерные системы и сети: сборник тезисов докладов 56-й научной конференции аспирантов, магистрантов и студентов, Минск, апрель-май 2020 года / Белорусский государственный университет информатики и радиоэлектроники. - Минск: БГУИР, 2020. - С. 58-60

UDC 004.33.54

SOFTWARE TOOL FOR SIMULATION OF LINKED FAULTS BASED ON COUPLING FAULTS

Tyushev T.A., Demenkovets D.V.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Levantsevich V.A. – senior lecturer

Annotation. This paper discusses the architecture of the memory faults simulator focused on coupling faults of three or more memory cells. Memory faults model and notation for their description are described. The algorithm for generating faults and inserting them to the memory model is presented. A software that simulates the process of testing the memory model using march tests is described.

Keywords. RAM, storage devices, RAM testing, march testing, coupling fault, software.