ПРОГРАММНЫЙ МОДУЛЬ АНАЛИЗА РЕЗУЛЬТАТОВ ИМИТАЦИОННЫХ ИСПЫТАНИЙ МИКРОЭЛЕКТРОННЫХ СИСТЕМ УПРАВЛЕНИЯ

Сущинский Д. В.
Кафедра "Микропроцессорная техника и информационно-управляющие системы"
Белорусский Государственный Университет Транспорта
Гомель, Республика Беларусь
E-mail: {dsuschinsky}@gmail.com

В ходе работы были проанализированы методы проведения имитационных испытаний на функциональную безопасность микроэлектронных систем управления. Было создано программное обеспечение позволяющее сократить временные затраты на процесс анализа результатов моделирования. Для подтверждения работоспособности данного приложения было проведено тестирование на примере импульсного декодера, подтвердившее все поставленные задачи.

Введение

Создание микропроцессорных и компьютерных систем железнодорожной автоматики и телемеханики (СЖАТ), неразрывно связано с совершенствованием и развитием методов обеспечения требуемого уровня их безопасности и надежности. Основным методом проверки работоспособности и оценки надежности и безопасности таких систем являются испытания на безопасность функционирования.

I. Содержание документа

Целью испытаний на безопасность функционирования является подтверждение того, что испытываемое устройство или система при возникновении заданного класса неисправностей аппаратных и программных средств, отказах внешних датчиков и неправильных действиях человека-оператора не формирует сигналы управления, нарушающие условия безопасности движения поездов. В виду значительных материальных и временных затрат на имитацию отказов и их устранение выполнить такой анализ можно только программными средствами. Однако современные программные пакеты хотя и позволяют проводить большое количество имитационных испытаний за малый промежуток времени, но не одна из них не предоставляет возможность классификации полученных результатов по видам последствий отказов. Как следствие это приходится делать человеку. Для одной или нескольких схем такой подход является вполне допустимым, но когда число различных отказов в схеме достигает сотен или тысяч, то такой подход становится не приемлемым. Была поставлена задача написать программу, удовлетворяющую следующим требованиям:

1. Уметь анализировать файлы типа *.csd с результатами моделирования схем в Pspice;
2. Дать возможность пользователю самостоятельно задавать критерии отказов;
3. По заданным критериям отказов программа должна классифицировать последствия отказов в схемах на опасные, защитные и маскируемые;
4. В конце работы формировать протокол о результатах, полученных при проведении имитационных испытаний.

II. Разработка программного обеспечения

Для хранения различного рода информации, необходимой непосредственно для работы программного продукта использована локальная база данных. При разработке данного приложения использовался объектно-ориентированный подход, согласно которому ядро программы составляют 5 классов. Объекты класса Schema содержат всю информации о схеме, которая достается из базы данных и хранится в оперативной памяти. Объекты класса Interval содержат всю информации об интервалах, которая достается из базы данных и хранится в оперативной памяти. Объекты класса Failure содержат всю информации об отказах схемы, которая достается из базы данных и хранится в оперативной памяти. Объекты класса Condition содержат всю информации о критериях отказов, которая достается из базы данных и хранится в оперативной памяти. В данных классах есть все поля, эквивалентные полям соответствующих таблиц базы данных, а также методы по работе с данными таблицами (добавление, редактирование и удаление). Объекты класса Report содержат информацию об отказах, найденных в схеме, и информацию, необходимую для создания протоколов испытаний. Программа в автоматическом режиме сканирует все каталоги с отказами и находит в них файлы с *.csd расширением. После нахождения файла программа начинает анализировать, выгружает файл в скрытый от пользователя объект Memo (пары значений «Время» – «Значение сигнала в этот момент времени»). После чего программа пробегает по всем временным отсчетам

в первом из указанных в программе интервалов и сравнивает значения сигнала в данный момент времени со значением переменной, если при задании условия была задана переменная, или со значением второго сигнала в этот же моменте времени, если при задании критериев был сигнал в качестве второй переменной. Если после анализа интервала не было выполнено условие для обнаружения отказа, то берется следующий критерий и анализируется таким же образом. В случае выполнения всех условий в критерии делается заключение о выполнении заданного критерия. Анализ отказов происходит в определенной последовательности: первыми проверяются критерии опасного отказа, затем защитного отказа и в конце – необнаруживаемого (маскируемого) отказа. Таким образом, в случае, если был обнаружен опасный отказ, то проверка критериев остальных отказов уже не нужна, поскольку система уже не прошла испытания на безопасность на заданном интервале. По ходу анализа критериев отказов на заданном интервале, программа создает протокол с информацией о том, какие критерии были проверены и какие отказы были обнаружены. После проверки всех критериев на всех интервалах программ создает протокол испытаний, который заполняется всей информацией, для генерации которой не нужно участие пользователя.

III. Тестирование

Рассмотрим тестовый пример на базе импульсного Кодера. Создадим в базе схему Кодер, и заполним стартовыми значениями как показано на рисунке 1.

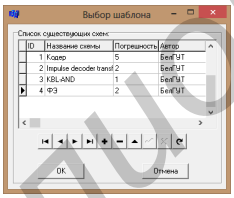

Рис. 1 – Создание схемы Кодер

Как показано на этом рисунке создана схема с названием Кодер, погрешностью 5 процентов, и автором «УО БелГУТ». Следующим шагом будет заполнение основной формы критериями отказов на определенных интервалах, как показано на рисунке 2.

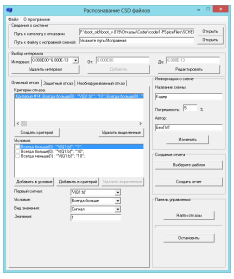

Рис. 2 – Заполнение основной формы

На этом рисунке показано заполнение критериев отказов схемы Кодер, на интервале от 0 до 6E-13 с для опасного, защитного и необнаруживаемого отказа. В результате анализа отказов этой схемы был создан протокол испытаний.

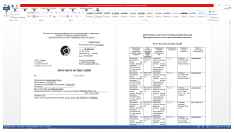

Рис. 3 – Шапка протокола испытаний

На рисунке 3 показана шапка протокола испытаний, заполненная в автоматическом режиме. На рисунке 4 показано окончание протокола испытаний.

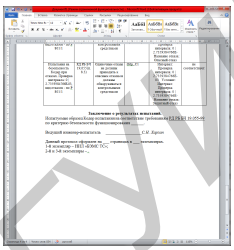

Рис. 4 – Окончание протокола испытаний

В конце протокола было сделано заключение о том, что данная схема не выдержала испытания на безопасность. Приложение удачно прошло фазу тестирования и справилось с поставленными перед ними задачами.

IV. Заключение

Приложение в автоматическом режиме просканировало указанную директорию, распознала все *.csd файлы, находящиеся в ней и классифицировала все значения записанные в них. Можно сказать, что все поставленные задачи были успешно решены.

V. Список литературы

1. Жаднов, В.В. Современные проблемы автоматизации расчетов надежности. / В.В. Жаднов, И.В. Жаднов, С.Н. Полесский // Надежность, №2 (21), 2007. С. 3-12.
2. Викторова В.С., Степанянц А.С. Анализ программного обеспечения моделирования надежности и безопасности систем. // Надежность, №4 (19), 2006. С. 46-57.
3. РТМ 32 ЦШ 1115842.01-94. Безопасность железнодорожной автоматики и телемеханики. Методы и принципы обеспечения безопасности микроэлектронных СЖАТ.
4. Микропроцессоры в 3 кн. Кн.2: Средства сопряжения. Контролирующие и информационно-управляющие системы. Учебник для вузов. / – М.: Высшая школа, 1987.
5. ГОСТ Р 51901.5-2005. Менеджмент риска. Руководство по применению ме-тодов анализа надежности.
6. Харлап С.Н., Комплекс для проведения имитационных испытаний микропро-цессорных систем железнодорожной автоматики на функциональную безопасность // Ресурсосберегающие технологии на железнодорожном транспорте: Материалы Всероссийской научно-технической конференции с международным участием. – Красно-ярск, 2005. – С. 188-193