Федеральное государственное бюджетное образовательное учреждение
высшего образования «Санкт—Петербургский государственный университет»

На правах рукописи

Елаев Евгений Валерьевич

АВТОМАТИЗАЦИЯ ТЕСТОВОГО КОНТРОЛЯ ЦИФРОВЫХ РАДИОЭЛЕКТРОННЫХ УСТРОЙСТВ

Специальность 2.3.3. — «Автоматизация и управление технологическими процессами и производствами»

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель:
доктор физико-математических наук, профессор
Овсянников Дмитрий Александрович

Санкт-Петербург – 2025

Оглавление

Введение

Актуальность работы. Современные экономические условия и указания правительства Российской Федерации к развитию отечественной промышленности в целом и электроники в частности, выдвигают высокие требования к качеству выпускаемой продукции при организации высокотехнологичных производств радиоэлектронной аппаратуры. Это порождает высокие требования к автоматизированным системам технического контроля работоспособности устройств, как в рамках автоматизированных систем управления производством, так и в контексте задач обслуживания и ремонта дорогостоящих и сложных систем цифровых радиоэлектронных устройств (ЦРЭУ). Эффективная система технического контроля и диагностики должна обеспечивать многоступенчатую стратегию поиска неисправностей в радиоэлектронных системах с максимальной глубиной поиска, вплоть до компонентов.

Это привело к необходимости разработки новых способов создания программных и поведенческих моделей цифровых устройств, методов и алгоритмов, которые позволили бы автоматизировать процесс определения работоспособности, поиска неисправностей радиоэлектронных систем, функционал и предназначение которых неизвестны. Указанные задачи тестового контроля носят фундаментальный характер, их решение обеспечит научный подход к вопросам автоматизации построения тестовых и диагностических программ для радиоэлектронных систем. Это существенно позволяет не только значительно повысить качество серийно выпускаемой продукции, но и увеличить эксплуатационные характеристики радиоэлектронных изделий как гражданского, так и военного назначения.

Актуальность задач тестирования для промышленного комплекса. Особое внимание хотелось бы уделить промышленным комплексам технического контроля качества и диагностики в рамках автоматизированных систем управления производствами, выпускающими радиоэлектронную продукцию,

в основе этих комплексов лежат методы и принципы структурного тестирования. Это класс задач касается контроля качества продукции, выпускаемой уже по заданной технологии. Особенность построения таких систем заключается в реализации задач реинжиниринга при построении и организации современных предприятий. В результате данного процесса возникла новая концепция организации производства, известная как интеллектуальные распределённые производственные системы. Она включает в себя интеграцию информационных потоков предприятия, децентрализованное управление, способность к взаимодействию в различных условиях, интеграцию человеческого фактора с программными и аппаратными ресурсами, а также возможность масштабирования и устойчивость к сбоям.

Базовой основой таких методов проектирования является система менеджмента качества в соответствии со стандартами международной организации по стандартизации ISO 9000. Кроме того, современные тенденции требуют широкого внедрения в производство CALS-идеологии, и таких стандартов как ISO 10303, ISO 9000, ERP.

Эффективность системы контроля качества в значительной степени обусловлена её непредвзятостью и способностью оперативно вмешиваться в контролируемый процесс. Это, в свою очередь, зависит от того, насколько точно и быстро действующая система тестирования, технического контроля и диагностики отражает реальное положение дел.

Для создания и эффективного использования подобных систем требуется инновационный подход к разработке аппаратно-программных комплексов тестирования и диагностики радиоэлектронных систем, организованных по иерархическому принципу и адаптированных для использования в составе распределенных интеллектуальных производственных систем.

Организационная структура систем технического контроля и диагностики может состоять из нескольких подсистем, некоторые из которых должны взаимодействовать друг с другом и с внешним оборудованием. В случае слож-

ных, многокомпонентных систем приходится расширять тестирование, проверяя синхронность работы как встроенных функций, так и взаимодействие подсистем друг с другом. Для такого тестирования нужны средства, позволяющие планировать и оценивать поведение не только отдельных элементов, но и всей системы в целом.

В последние годы помимо производства промышленностью на постоянной основе широкой номенклатуры ЦРЭУ различной степени интеграции, нуждающейся как в тестовом контроле, так и в ремонте в ходе эксплуатации, наблюдается устойчивая тенденция к усложнению радиоэлектронных систем за счёт внедрения новых, более совершенных компонентов. Это ведёт к повышению функциональности и эффективности конечного продукта. Современным радиоэлектронным системам характерны:

- корреляционные связи (в процессе работы между аналоговыми и цифровыми компонентами, входящими в состав единого модуля или системы, возникают корреляционные связи, они обеспечивают согласованное функционирование различных частей системы и повышают её надёжность и стабильность);

- интерфейсы для взаимодействия (для обеспечения взаимодействия между различными блоками системы и подключения к вычислительным и контрольным устройствам современные радиоэлектронные системы оснащаются множеством разнообразных интерфейсов,что позволяет интегрировать систему в более широкие сети и обеспечивает гибкость её настройки и эксплуатации);

- гетерогенный характер (одной из ключевых особенностей современных радиоэлектронных систем является их гетерогенный характер, который проявляется в использовании как цифровых, так и аналоговых компонентов в рамках одной системы,что позволяет создавать более гибкие

и адаптивные системы, способные эффективно решать широкий спектр задач).

Исходя из вышеизложенного, можно сделать вывод о том, что многообразие структурных решений предполагает наличие широкого спектра методов тестирования, что, в свою очередь, обеспечивает гибкость при разработке аппаратно — программных комплексов для технического контроля качества и диагностики, оснащёных интегрированной программной средой. Комплексность и неоднородность таких систем существенно усложняет как тестирование, диагностику и отладку объектов контроля, так и технологию подготовки, и программирование тестовых средств.

Степень разработанности темы. Вопросу тестирования цифровых электронных устройств, посвящены работы таких ученых, как Lala P. K., Bergeron J., Thoulath B., Yin X. H., Xu C. F., Bushnell M. L., Панков Д. А., Увайсов С. У., Сулейманов С. П., Золоторевич Л. А., Мосин С. Г., Скобцов Ю. А., Скобцов В. Ю., Иванов И. А. Овсянников Д. А., Гришкин В. М., Михайлов А. Н., Лопаткин Г. С.. Однако, вопросам моделирования, методам и алгоритмам автоматизированного структурного тестирования уже произведенных устройств уделено недостаточно внимания.

Целью диссертационного исследования является разработка методов, алгоритмического и программного обеспечения автоматизированного тестового контроля сложных ЦРЭУ, оценивающих их работоспособность путем автоматизированной генерации входной тестовой последовательности сигналов и моделирование реакции объекта контроля (ОК) на них.

Для достижения поставленной цели, сформулированы следующие задачи.

Задачи диссертационной работы:

- выполнить анализ существующих методов и средств тестирования ЦРЭУ;

- разработать методы моделирования цифровых устройств, детализированные вплоть до уровня компонентов на основе математических моделей;
- разработать алгоритм автоматизированного создания проверяющей тестовой последовательности для цифрового ОК;
- разработать метод интеграции аналоговых узлов в имитационную модель ОК, используя их математическую модель;
- создать специализированное программное обеспечение, позволяющее автоматизировать процесс тестирования ЦРЭУ.

Научная новизна работы заключается в следующем:

- разработаны методы создания программных моделей компонентов радиоэлектронных систем на основе их технических описаний и математических моделей;
- разработан метод внедрения аналоговых узлов в имитационную модель ОК на основе их математических моделей;
- разработана и реализована в виде программного обеспечения технология формирования моделей входных воздействий, обеспечивающих реакцию всех компонентов системы и активизацию связей между ними;
- создан и реализован в виде программного модуля алгоритм автоматизированного тестирования ЦРЭУ, позволяющий определить работоспособность ОК;
- разработаны принципы моделирования цифровых систем, в состав которых входят элементы программируемой логики, при отсутствии доступа к конфигурирующей программе («прошивке»).

Теоретическая значимость работы состоит в разработке математического аппарата, алгоритма и методов, позволяющих автоматизировать тестирование цифровых устройств.

Практическая значимость результатов работы состоит в создании программного комплекса CRIT, позволяющего автоматизированно формировать адекватные тесты контроля качества для различных радиоэлектронных систем. Эти тесты предназначены для работы в комплексе с установками тестового контроля, выпускаемыми отечественной промышленностью такими как УТК-512 и Скат-Ц. Указанные установки используются на ряде отечественных промышленных предприятий.

Методы исследования: в работе используются численные методы, методы автоматизации технологических процессов, информационные технологии, методы нейронных сетей, структурный анализ, декомпозиция. **Методологической основой** являются компьютерные технологии и математическое моделирование.

Основные положения, выносимые на защиту:

- метод формирования программных моделей цифровых микросхем в контексте задач тестирования;
- метод интегрирования нецифровых аналоговых компонентов в объект контроля, путем программной реализации их математических моделей функционирования;
- интерфейсный метод и алгоритм автоматизированного построения тест—программ;
- компьютерная технология интерфейсного метода, интегрированная в программный комплекс CRIT, автоматизирующая процесс построения тест—программ.

Достоверность полученных результатов подтверждается практическим путем при тестировании цифрового объекта контроля на основе предложенных подходов, алгоритма и методов, с использованием программного комплекса CRIT.

Апробация материалов диссертации в работах. Результаты, полученные автором, докладывались на научных конференциях: 2014 International conference on computer technologies in physical and engineering applications (ICCTPEA); Четвертой конференции «Информационные технологии на службе оборонно-промышленного комплекса России 2015»; “Huawei” Company Seminar “Open Day” 2019; MWENT-2018(Moscow Workshop on Electronic and Networking Technologies).

Результаты диссертационной работы, подтвержденные соответствующими **актами внедрения**, используются в производственной деятельности АО «Производственная компания «Специальные Инновационные Технологии»» (Приложение Б), а также в учебном процессе факультета ПМ-ПУ СПбГУ (Приложение В).

Результаты опубликованы соискателем, в том числе в рецензируемых научных изданиях из «Перечня ВАК». По материалам диссертации опубликовано в общей сложности 13 работ [1–13], в том числе получено 1 свидетельство о регистрации программы на ЭВМ [13]. В журналах ВАК, соответствующей специальности – 4 работы [1–4]; в изданиях, индексируемых в международных базах данных (Scopus, Web of Science) – 3 работы [5–7].

Соответствие паспорту научной специальности. Диссертационное исследование проведено в соответствии с п.2 «Автоматизация контроля и испытаний» и п.15 «Теоретические основы, методы и алгоритмы диагностирования (определения работоспособности, поиск неисправностей и прогнозирования) АСУТП, АСУП, АСТПП и др.», паспорта научной специальности **2.3.3.- «Автоматизация и управление технологическими процессами и производствами».**

Структура работы. Диссертационная работа состоит из введения, трех глав, заключения и списка литературы, содержащего 124 наименований, списка сокращений, списка иллюстративного материала, 6 приложений. Общий объем диссертации составляет 128 страниц, включая 36 рисунков и 8 таблиц.

Глава 1

Единая методология создания поведенческих моделей цифровых радиоэлектронных устройств в контексте задач тестового контроля

1.1 Анализ существующих методов и средств тестирования цифровых радиоэлектронных устройств

Среди всего многообразия методов, способов и средств тестирования цифровых радиоэлектронных устройств можно выделить два глобальных и фундаментальных направления.

Первое направление — это тестирование устройства на этапе его разработки и отладки, перед тем как запустить изделие в серию. Цель тестирования — обнаружить потенциальные проблемы и несоответствия с проектным требованиям. Эти проверки охватывают оценку функциональности, производительности, безопасности.

Тестирование разработанной функциональности применяется для оценки работоспособности всех функций и возможностей проектируемого устройства.

Тестирование производительности измеряет скорость работы устройства, его способность выдерживать высокие нагрузки. Оно включает нагрузочное тестирование, стресс-тестирование и тестирование стабильности.

Безопасность устройства также подвергается тщательному тестированию и анализу надежности и уязвимостей системы к внешним угрозам. Кроме того, на этапе разработки и отладки устройства проводятся испытания на соответствие стандартам и нормативным требованиям.

Так же на этапе проектирования часто пытаются спрогнозировать и осуществить имитацию возможных ошибок, дать оценку вероятности возникновения таких неисправностей, что способствует сужению области поиска дефектов [14–37]. При этом стоит отметить список таких ошибок не может обеспечить всю необходимую полноту проверки неисправностей.

Второе направление и цель данной работы — структурное тестирование, т.е. тестирование, которое применяется для оценки работоспособности финального, т.е. уже разработанного изделия. Его отличительная черта заключается в том, что проверяется корректность работы в соответствии со схемой устройства, которая реализует запланированный и разработанный функционал. Предполагается, что этап проектирования и разработки цифровой схемы устройства реализован без ошибок (например, в нем отсутствуют «гонки фронтов»), и в случае его реализации, изделие исправно выполняет возложенную на него функцию. Таким образом, тестовая программа не проверяет соответствие готового изделия задумке проектировщика и корректность выполнения устройством предусмотренных в нем алгоритмов.

В рамках проверки технического состояния устройства осуществляется оценка функционирования его составных элементов, а также анализируется целостность связей между этими элементами и отсутствие непредусмотренных «паразитных» соединений внутри устройства.

В контексте данного исследования разработка теста устройства предполагает поиск таких последовательностей входных воздействий, которые, проходя через тестируемые компоненты, вызывают генерацию внутренних сигналов, достигающих выходных разъемов [38–49].

В настоящее время при решении подобных задач наибольшее применение на практике получили такие средства диагностирования, как автоматизированное тестовое оборудование (Automated Test Equipment – ATE). Применение ATE позволяет значительно ускорить процесс определения работоспособности радиоэлектронных систем и обнаружения неисправностей. Интеграция

АТЕ в качестве средств диагностирования на всех уровнях жизненного цикла производства и эксплуатации радиоэлектронных систем — есть ключевое направление развития современных методов контроля и диагностики. По используемому способу диагностирования различают следующие типы АТЕ:

- визуальный автоматизированный контроль (Automated Optical Inspection – AOI);
- внутрисхемное тестирование (In-Circuit Test – ICT);
- периферийное/граничное сканирование (Boundary Scan);
- функциональное тестирование (Functional Test – FCT).

Автоматизированный визуальный контроль (AOI) [50–55], является одним из широко распространенных методов диагностирования, используемых в изготовлении и эксплуатации радиоэлектронных систем. Для проведения автоматизированного визуального контроля (AOI) используются специализированные осветительные приборы, камеры, системы компьютерного наблюдения. Это позволяет добиться точной высокоскоростной оценки качества широкого спектра радиоэлектронных устройств. Системы «машинного видения» или AOI системы могут оценивать миллионы контрольных точек, используемых для визуального контроля и точных измерений в доли секунды. AOI системы используют визуальные методы контроля печатных плат на наличие дефектов. Они способны обнаруживать различные дефекты поверхности печатных плат, в том числе царапины, пятна, а также более распространенные, такие как обрывы, короткие замыкания, истончения припоя и т.д. AOI также могут обнаружить несоответствующие, отсутствующие или неправильно установленные компоненты. Иными словами, они могут выполнять все визуальные проверки, которые ранее осуществлялись вручную.

Использование систем автоматизированного визуального контроля зачастую требует сложного и дорогостоящего оборудования. Кроме того, для обес-

печения наилучших результатов, требуются статистические данные. Получение таких данных сопряжено со значительными временными затратами.

В процессе использования технологии **внутрисхемного тестирования (ICT)** [56–64] тестер получает доступ к внутренним цепям печатной платы с помощью контактных иголок, которые адаптер тестера прижимает к поверхности платы. Это позволяет проводить тестирование электронных компонентов и цепей без необходимости физического вмешательства в устройство.

Игольчатый адаптер — это устройство, которое обеспечивает электрический контакт между различными компонентами электронных схем. Он состоит из набора игл, которые могут контактировать с поверхностями различных компонентов. Игольчатые адаптеры широко используются в производстве электроники для тестирования, ремонта и модификации электронных устройств.

Кроме того, широкое применение нашли системы «летающих щупов» (англ. flying probe test) [65–71], которые отличаются большей универсаль-ностью. Системы «летающих щупов» — это автоматизированные системы тестирования печатных плат, которые широко применяются в современной электронной промышленности. Они используются для проверки целостности электрических цепей на печатных платах и выявления возможных дефектов.

Системы «летающих щупов» работают следующим образом: специальный манипулятор с зондом перемещается над поверхностью печатной платы, автоматически проверяя соединения между элементами схемы. Если обнаруживается дефект, система сигнализирует об этом.

Одним из основных недостатков является необходимость тщательной подготовки тестовых образцов. Перед началом тестирования необходимо обеспечить чистоту печатной платы, отсутствие загрязнений и посторонних предметов, таких как пыль, мельчайшие остатки стружки или кусочков материалов. В противном случае результаты тестирования могут быть неточными или ошибочными.

Ещё одним недостатком является возможность повреждения чувствительных элементов печатной платы. При контакте с летающими щупами могут возникнуть механические повреждения, особенно если тестируемая плата имеет хрупкие или чувствительные компоненты.

Также стоит отметить, что системы «летающих щупов» требуют квалифицированного персонала для настройки и обслуживания. Неправильное использование или настройка системы может привести к ошибкам в результатах тестирования и повреждениям объекта контроля.

Для выполнения процедуры **граничного сканирования (Boundary Scan)** требуется, чтобы интегральные схемы, установленные на печатной плате тестируемого устройства, включали в себя структуру граничного сканирования. Суть метода заключается в тестировании платы через специализированный компактный разъём с использованием 4-проводного JTAG-интерфейса [72–82], который был стандартизирован IEEE 1149.1. Стандарт был разработан группой Joint Test Action Group (JTAG) и впервые опубликован в 1990 году. Он определяет набор команд и протоколов для тестирования печатных плат и электронного оборудования. Стандарт обеспечивает возможность тестирования широкого спектра устройств, включая микропроцессоры, микроконтроллеры, программируемые логические интегральные схемы (ПЛИС) и другие цифровые компоненты.

При активации режима граничного сканирования компоненты, оснащенные интерфейсом JTAG, временно приостанавливают выполнение своей основной функции и переходят в тестовый режим, предоставляя внешнему оборудованию контроль над своими выводами, что позволяет эффективно анализировать состояние электрических цепей.

Процесс диагностики интегральной схемы с применением метода граничного сканирования включает в себя запись определённых данных в регистры и сравнение полученного результата с эталонным.

Инструменты, предназначенные для тестирования по JTAG-интерфейсу, также позволяют выполнять внутрисистемное программирование Flash-памяти и программируемых логических интегральных схем (ПЛИС).

JTAG-тестирование, будучи эффективным инструментом диагностики, предлагает возможность выявления дефектов пайки в соединениях цифровых интегральных микросхем с различными типами корпусов, включая BGA, позволяет определять наличие коротких замыканий и обрывов, а также идентифицировать неисправные микросхемы, оборудованные цифровыми интерфейсами.

Элементы печатной платы, которые не содержат структуру граничного сканирования, могут тестироваться косвенно. Однако достичь максимального покрытия тестом печатной платы в таких случаях гораздо труднее, а нередко и совсем неосуществимо. Невозможно обнаружить дефекты монтажа, связанные с цифровыми или аналоговыми элементами, которые не имеют JTAG-поддержки, также недоступна диагностика дефектов связей между ними.

Технология граничного сканирования (Boundary Scan) имеет ряд недостатков, которые следует учитывать при её применении:

- ограничения по тестированию сложных устройств (хотя технология позволяет тестировать широкий спектр устройств, она может не подходить для тестирования сложных систем, где требуется более глубокое и детальное исследование);

- необходимость поддержки стандарта IEEE 1149.1 (для использования технологии граничного сканирования все тестируемые устройства должны поддерживать стандарт IEEE 1149.1, это может ограничить применение технологии для устаревших или нестандартных устройств);

- сложность настройки и обслуживания тестового оборудования (настройка и обслуживание тестового оборудования, особенно если оно исполь-

зуется для тестирования сложных устройств, может потребовать значительных усилий и времени);

- возможность ложных срабатываний (в некоторых случаях технология граничного сканирования может давать ложные срабатывания, что может привести к дополнительным затратам времени и ресурсов на устранение неполадок);

- ограничения по глубине тестирования (технология граничного сканирования позволяет тестировать только определённые аспекты устройства, что может не дать полной картины о его работе);

- зависимость от качества тестовых программ (качество тестирования зависит от качества тестовых программ, которые используются для управления процессом тестирования, ошибки в тестовых программах могут привести к неправильным результатам тестирования);

- необходимость обучения персонала (персонал, занимающийся тестированием с использованием технологии граничного сканирования, должен пройти обучение и иметь соответствующие навыки работы с оборудованием и программами тестирования).

Функциональное тестирование (FCT) представляет собой проверку радиоэлектронных систем на выполнение заданной функциональности и на соответствие параметрам, которые заложены в спецификации [36, 44–49, 62]. Составление тестов для проведения функционального тестирования является трудоёмкой задачей, требующей привлечения квалифицированных специалистов. Для написания функциональных тестов предлагается использовать системы моделирования и автоматизированные системы построения тестов, которые представляют собой комплекс программных средств, обеспечивающих создание модели функционирования радиоэлектронных систем и её поведения в различных ситуациях, в том числе и неисправности элементов неисправной

радиоэлектронной системы. Моделирование радиоэлектронных систем позволяет спрогнозировать поведение исправного и неисправного устройства и на основе моделирования выдвинуть гипотезу о возможной неисправности. По результатам моделирования гипотеза проверяется измерениями на реальном радиоэлектронном устройстве.

Анализ современных средств и методов тестирования и диагностики радиоэлектронных систем позволяет сделать вывод о том, что наиболее перспективным является разработка средств функционального тестирования с использованием функций моделирования состояния и поведения радиоэлектронных систем, так как они обладают следующими достоинствами:

- универсальность применения (в отличие от граничного сканирования, не требуют, чтобы компоненты соответствовали каким либо стандартам; не требует контактных площадок как внутрисхемное тестирование);
- независимость от технологий производства радиоэлектронного устройства, включая количество слоёв и размеры компонентов (это делает его универсальным методом проверки, который может применяться к широкому спектру устройств без необходимости адаптации под конкретные технологические особенности производства);
- относительная дешевизна (по сравнению с AOI и ICT системами);
- возможность оценки не только внешнего состояния радиоэлектронного устройства и качества соединений, но и соответствия параметрам, которые заложены в спецификации;
- не требует наличия физического радиоэлектронного устройства на этапе создания тестовой программы.

Таким образом, актуальное значение приобретают средства моделирования поведения радиоэлектронных устройств с учётом подаваемых стимулирующих воздействий. В настоящее время вопросам создания математиче-

ских моделей и средств их разработки уделяется достаточно большое внимание, так как физическое моделирование сложных радиоэлектронных систем связано с большими материальными затратами. Основные исследования и разработки в этой области ведутся исследовательскими центрами компаний ALTERA, National Instruments, Cadence Design Systems, Mentor Graphics, Spectrum Software и Protel International. Этими компаниями разработаны программные продукты для моделирования и анализа аналоговых и цифровых радиоэлектронных систем: QuartusII (Altera), Multisim (National Instruments), Circuit Maker (Protel International), Micro-Cap (Spectrum Software), Pspice (Cadence Design Systems), ModelSim (Mentor Graphics).

Программный пакет Micro-Cap действительно предлагает удобный графический редактор, который позволяет пользователям создавать модели аналоговых и цифровых систем, а также анализировать их параметры. Программа содержит обширную библиотеку компонентов от ведущих производителей из США, Европы и Японии, что делает её мощным инструментом для проектирования и анализа электронных схем [83, 84].

Особенностью программного продукта Multisim является наличие виртуальных измерительных приборов, имитирующих реальные аналоги. Программа хорошо усваивается начинающими пользователями. Этим объясняется ее популярность в учебных заведениях. Пакет программ включает в себя все современные методы математического моделирования и анализа схем. В то же время следует отметить, что лицензионные версии весьма дороги [85–90].

Circuit Maker представляет собой программное обеспечение, предназначенное для имитационного моделирования функционирования аналоговых, цифровых и комбинированных аналого-цифровых систем. Программное обеспечение оснащено интуитивно понятным графическим интерфейсом, который значительно упрощает процесс оперативного и эффективного создания макетов аналоговых и цифровых систем. Результаты моделирования отображаются в графическом виде в форме осциллограмм и графиков частотных характери-

стик. Эта версия программы предназначена для использования на персональных компьютерах.

Тем не менее, Circuit Maker может иметь ограниченную поддержку определенных типов компонентов или функций, что может затруднить разработку сложных проектов. Также программа может требовать значительных вычислительных ресурсов для выполнения некоторых задач, что может замедлить работу на менее мощных устройствах [91, 92].

Pspice — это модифицированная версия программы анализа электронных цепей SPICE. Модели электронных компонентов в формате SPICE используются в большинстве других программ для моделирования и анализа. Продукт имеет удобный и понятный графический интерфейс. Для представления результатов моделирования в удобной форме используется графический постпроцессор Probe. Он отображает графики результатов моделирования на экране и выполняет их математическую обработку.

Тем не менее, у Pspice есть и некоторые недостатки. Например, программа может требовать значительных вычислительных ресурсов, особенно при работе с большими моделями [93–95].

Корпорация ALTERA разработала среду программирования Altera Quartus II и Quartus Prime для своих микросхем. Эти программа представляет собой мощный инструмент для проектирования логики работы микросхем, который позволяет разработчикам создавать сложные электронные устройства. Altera Quartus II и Quartus Prime поддерживают несколько языков программирования, включая HDL, VHDL и Verilog, а также предоставляет возможность проектирования схем на уровне транзисторов. Это позволяет разработчикам выбирать наиболее подходящий метод в зависимости от требований проекта.

В состав сред программирования Altera Quartus II и Quartus Prime входят инструменты для симуляции проектов и программирования микросхем, что позволяет обеспечить высокую точность моделирования и минимизировать ошибки при разработке. Наличие обширной библиотеки моделей элемен-

тов от ведущих американских производителей существенно упрощает процесс проектирования и сокращает время разработки [96–100].

Эти средства дают возможность анализировать поведение устройства при подаче конкретных стимулов. Однако они не позволяют автоматизировать процесс формирования моделей таких тестовых воздействий, которые бы обеспечивали адекватный контроль и диагностику сложных радиоэлектронных систем.

Решение этой задачи влечет за собой разработку автоматизированных методов формирования моделей систем и поиска моделей тестовых воздействий, необходимых для построения систем тестирования и диагностики радиоэлектронных систем, отсутствующих в настоящее время.

1.2 Области представления и уровни проектирования

Подходы и методы, предложенные в работе к построению программного комплекса для автоматизации процессов построения тестовых программ цифровых радиоэлектронных устройств, основаны на математическом моделировании этих устройств. Однако существует огромное количество научных школ и научно-производственных объединений, занимающихся проектированием, разработкой и изготовлением цифровых радиоэлектронных устройств. Практически каждая из подобных организаций имеет свой определенный подход к моделированию, свою концепцию отличную от остальных, свой взгляд на проектирование.

В контексте комплексного подхода к решению задач автоматизированного тестирования цифровых устройств, которые уже были спроектированы и произведены без учёта методов автоматической проверки (самопроверки), становится очевидной необходимость разработки унифицированной методологии создания поведенческих математических моделей этих устройств в рамках систем автоматизированного проектирования (САПР) [35].

Важно отметить, что при разработке цифровых устройств необходимо проводить различие между моделями устройств и их спецификациями. Спецификация представляет собой документ, который содержит исчерпывающую информацию об устройстве, полученную на основе завершающих этапов проектирования изделия. Этот документ включает в себя детальное описание характеристик устройства, выраженных через временные диаграммы, технические параметры и другие специализированные термины. Спецификация служит основой для формирования полного представления об устройстве, позволяя оценить его функциональность, надёжность и эффективность

Модели устройств находят широкое применение в процессе проектирования, предоставляя возможность осуществлять эмуляцию на различных уровнях абстракции. Это позволяет контролировать на сколько параметры устройства совпадают с заданными спецификациями [35].

Всё многообразие методов и способов создания цифровых устройств и их моделей можно классифицировать в зависимости от подходов к пониманию сути цифрового радиоэлектронного устройства и его организации, вследствие этого, можно отметить три основных области, в которых происходит их моделирование и проектирование (рис. 1.1) [101–103]:

- физическая область, которая отражает непосредственно электронную схему, реальный кристалл(chip) (в рамках данной области проектируется топология кристалла, определяются физические характеристики объекта и т.п.);
- структурная область, которая описывает блоки архитектуры цифрового устройства (блоки, регистры, логические вентили, транзисторы);
- поведенческая область, где осуществляется функциональное представление цифрового устройства.

При этом для каждой из представленных областей выделяют различные уровни моделирования [35]:

- схемный,
- логический,
- языков регистровых передач (ЯРП),
- системный.

Рис. 1.1. Диаграмма уровней абстракции Гайского-Кана

Для наглядности рассмотрим логический уровень моделирования в каждой из области (рис. 1.2).

Каждый тип и способ несет в себе свои особенности реализации, которые могут повлиять на модель и которые нужно учитывать. При разработке, проектировании и синтезе цифрового устройства работы ведутся на различных уровнях абстракции и в областях проектирования, переходя из одного в другой [101].

В процессе разработки компьютерной модели цифрового устройства с целью её последующего тестирования в рамках предложенной концепции, особое внимание будет уделено работе в области поведенческого и струк-

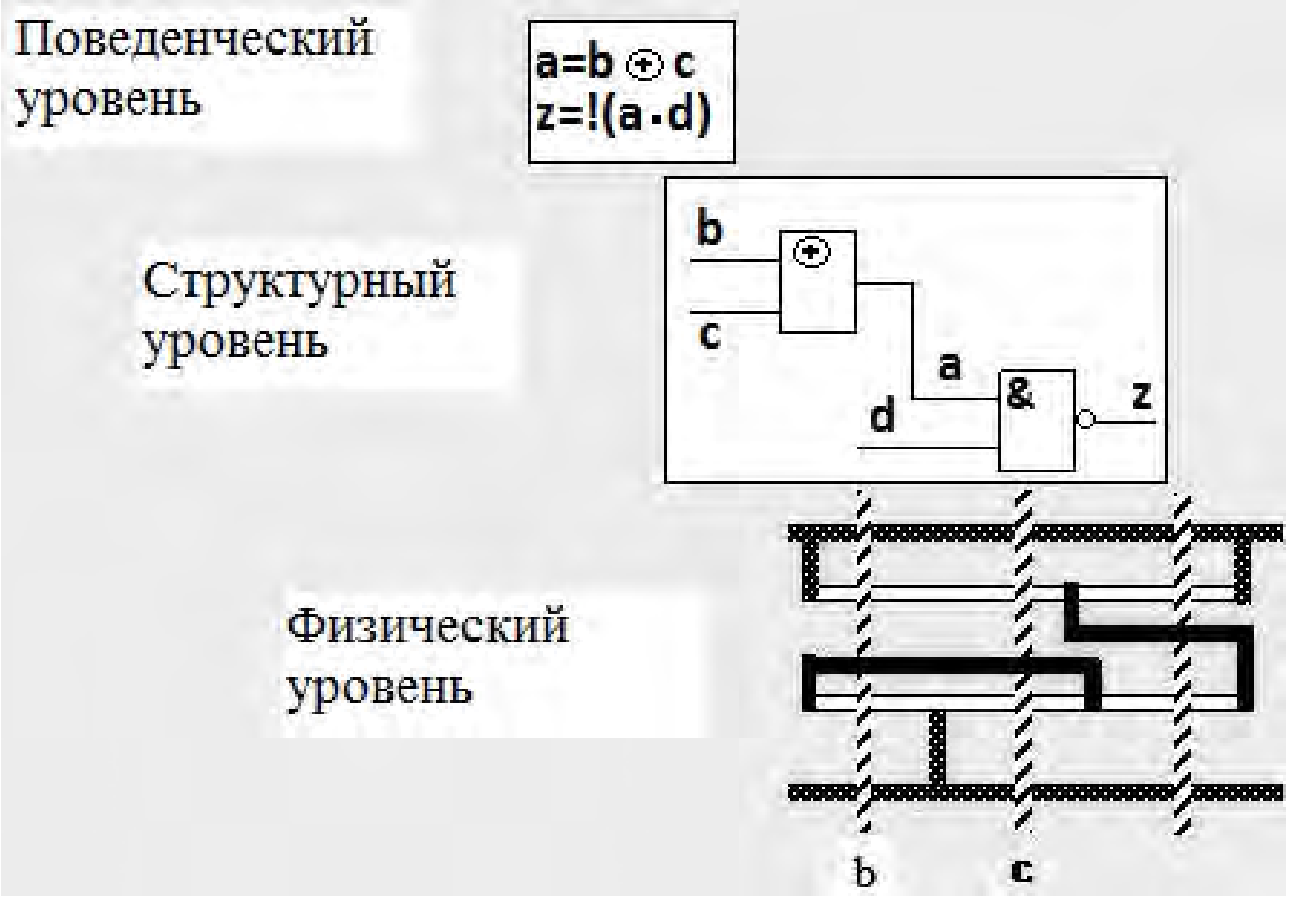

Рис. 1.2. Области представления на логическом уровне

турного представления и на логическом этапе проектирования. На одном и том же конкретном уровне, в любой выбранной области, существует несколько вариантов описания одного и того же цифрового устройства. Например, цифровое устройство на логическом уровне в поведенческой области может быть описано с помощью булевых выражений, таблиц истинности, различных языков программирования или языков описания аппаратуры [35].

1.3 Методы формирования моделей цифровых микросхем, в контексте задач тестирования

С учетом специфики задач тестирования цифровых радиоэлектронных устройств выдвигаются определенные требования к составлению поведенческих моделей цифровых микросхем и их адекватности.

Модель цифрового объекта контроля создается на основе анализа уже известной структуры устройства, т.е. принципиальной схемы, которая отражает способы взаимосвязи между микросхемами (компонентами устройства) и их типа (это могут быть как логические или последовательные микросхемы, так микросхемы памяти или программируемой логики). В рамках рассматриваемой задачи, уровень детализации поведенческой модели вплоть до уровня компонентов достаточен, ввиду того, что ремонт тестируемых изделий осуществляется путем замены неисправных компонентов и/или восстановления связей между ними. Становится очевидно, что для эффективного моделирования цифрового устройства в контексте задач тестового контроля, необходимо разрабатывать модели объекта тестирования, детализированные вплоть до уровня компонентов цифровых радиоэлектронных устройств.

Таким образом, разрабатывается математическая модель компонентов устройства, затем она реализуется программным способом с использованием языков описания и проектирования аппаратных средств (hardware definition language), таких как HDL Verilog или VHDL [104–109]. Эти языки позволяют точно описать структуру и поведение цифровых схем, что необходимо для создания точных и эффективных программных моделей.

Использование языков HDL для создания программных моделей цифровых компонентов имеет ряд преимуществ перед другими методами. Во-первых, это позволяет обеспечить высокую степень соответствия между моделью и реальным устройством, что критически важно для многих задач, таких как моделирование электронных схем, разработка встроенного программного обеспечения и тестирование оборудования. Во-вторых, HDL-языки предоставляют мощные инструменты для верификации и отладки моделей, что способствует повышению качества и надёжности разрабатываемых систем устройств.

Анализируя опыт работы в сфере тестирования, обучения и подготовки персонала для осуществления деятельности тестового контроля (в Приложе-

нии Г можно ознакомиться с копией благодарственного письма за соответствующую деятельность), были выявлены определенные особенности составления моделей в контексте задач тестирования цифровых устройств, на основе чего предложены два равноправных метода разработки цифровых программных моделей:

- интеграционный метод;
- функциональный метод.

Функциональный метод. В соответствии с этим подходом, на основе анализа информации об алгоритмах функционирования элемента, о его предназначении и типовых операциях, присущих данной микросхеме, составляется математическая модель элемента, которую уже реализуют посредствам языков VHDL или Verilog HDL [96, 100], получая программную модель (Verilog design file).

Одним из неоспоримых преимуществ данных языков является синтезабельность HDL описаний, которая им присуща и является одним из важнейших аспектов моделирования. Суть заключается в том, что синтезируемые с помощью HDL модели компонентов объекта контроля способны минимизировать неопределенности, которые могут возникать во время переключения сигналов. Это предоставляет преимущество перед несинтезируемыми моделями, у которых такая возможность отсутствует.

Синтезируемые модели обладают преимуществом в удобстве проверки: если система синтеза способна создать структуру на основе модели, это уже свидетельствует о том, что в реализации отсутствуют ошибки (такие как присвоение значения одного сигнала разным процессам). Однако, следует учитывать, что успешный синтез сам по себе необходим, но не дает гарантии корректности функционального описания.

Таким образом, при моделировании цифровой микросхемы на языке Verilog для задач тестирования (но не проектирования), помимо синтезабельности программной модели, существует еще ряд требований к самой модели:

- при моделировании элемента не реализовывать задержки, отражающие время срабатывания физического элемента, однако учитывать задержки, вызванные особенностью проектирования элемента и спецификой выполнения некоторых функций (процедур) элементом;

- не производить предустановку элемента конкретным значением сигнала (не указывать начальные значения), используя оператор $initial$, если в техническом описании нет соответствующих указаний, иначе это может привести к тому, что при работе тест-программы с реальным физическим объектом, могут появляться ошибки, вызванные несоответствием начальных данных;

- при наличии запрещенных состояний работы элемента, описывать их отдельным циклом или командой, но при тестировании объекта контроля, стараться избежать подачу сигнала, вызывающую работу в запрещенном режиме;

- при выборе способов выполнения операции присваивания приоритет отдается оператору непрерывного присваивания ($assign$), а не циклам с оператором $always$;

- каждую функциональную операцию, выполняемую элементом (запись, хранением, счет и т.п.), описывать отдельным циклом или командой;

- не использовать переменные типа reg, с одним названием в разных циклах работы синхронной логики;

- выходные переменные ($output$) элемента не объединять в шину данных (если количество выходов физического объекта равно N, то количество

выходов у программной модели элемента тоже должно быть N, а не один выход, с N-разрядностью).

Продемонстрируем функциональный подход на примере триггера *1533TM2* [110–112] (рис.1.3).

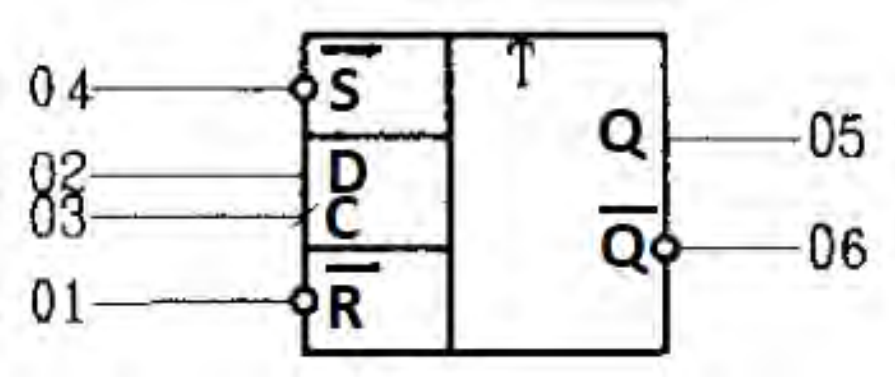

Рис. 1.3. Условно-графическое представление элемента *1533TM2*

Назначение выводов микросхемы:

- вывод 04 — вход установки S микросхемы;
- вывод 01 — вход сброса R триггера;
- вывод 03 — вход тактовый C;
- вывод 02 — вход данных D;
- вывод 05, 06 — выход данных $Q, \overline{Q}$.

Стоит отметить, что низкий уровень сигнала (логический ноль) обозначают как «0» или "L" высокий уровень сигнала (логическая единица) обозначают как «1» или "H" , "X" — произвольный уровень сигнала.

Низкий уровень напряжения на входах установки S (Set) или сброса R (Reset) устанавливает выходы триггера в соответствующие состояния вне зависимости от состояния на других входах. При наличии на входах установки или сброса напряжения высокого уровня для правильной работы триггера требуется предварительная установка информации по входу данных D (Data) относительно положительного фронта тактового сигнала C (Clock), а так же

соответствующая выдержка сигнала информации после подачи положительного фронта синхросигнала [112, 113].

Математическая модель работы триггера выглядит следующим образом:

$$\begin{aligned} &Q = \overline{Q}, \\ &Q = 1, \quad \text{при} \quad S = 0, \quad R = 1, \\ &Q = 0, \quad \text{при} \quad S = 1, \quad R = 0, \\ &Q = D, \quad \text{при} \quad S = R = 1, \quad C = 0 \to 1, \\ &Q = Q_0, \quad \text{при} \quad S = R = 1, \quad C = 0. \end{aligned} \tag{1.1}$$

Здесь Q_0 — предыдущее состояние триггера, $C = 0 \to 1$ — фронт сигнала C, т.е. момент перехода его от 0 к 1. Исходя из анализа данной информации, об алгоритмах работы, была разработана модель элемента, с программным кодом, который можно подробно изучить в приложении Д.

К преимуществам данного метода, можно отнести:

- краткость и лаконичность программного кода (не надо учитывать структурную декомпозицию элемента и взаимосвязь его компонентов);
- отсутствие потребности в базе данных компонентов для составления программной модели элементов;
- снижение вероятности ошибки и времени создания модели.

Недостатками являются:

- невозможность создания модели, при отсутствии описания работы микросхемы (зачастую информация о микросхеме ограничивается ее функциональной схемой);
- невозможность учесть недокументированные возможности;
- необходимость продвинутого владения Verilog;
- необходимость наличия высоких аналитических навыков;

- вариативность в описании программной модели (программный код одной и той же модели может отличаться, в зависимости от стиля программирования на языке Verilog.

Интеграционный метод. Суть интеграционного метода (или метода интеграции) заключается в реализации подходов восходящего проектирования, в соответствии с которым сначала решаются задачи более низкого уровня иерархии, а затем осуществляется переход к задачам последующего (более высокого) уровня [96].

Всё многообразие цифровых устройств имеет свою иерархическую структуру, элементы более высокого уровня иерархии состоят из элементов более низкого уровня. Так, например, объекты класса цифровых устройств »счетчик» в соответствии со своим схемным решением состоят из объектов классов более низкого уровня иерархии, таких как «триггер» и «логический элемент» объект класса «триггер» в свою очередь состоит из объектов класса «логический элемент» и т.п.

К микросхемам высокого уровня иерархии относятся:

- микросхемы запоминающих устройств (ОЗУ, ПЗУ);
- комбинаторные микросхемы – микросхемы, выходные сигналы которых однозначно определяются входными сигналами в данный момент времени (сумматор, шифратор, дешифратор, мультиплексор, цифровой компаратор и т.п.);
- последовательные микросхемы – микросхемы, выходные сигналы которых определяются не только сигналами на входах в данный момент, но и значениями входных сигналов на предыдущих тактах, то есть элементы с внутренней памятью (триггер, счетчик, регистр и т.п.).

К низшему уровню иерархии цифровых устройств относятся цифровые логические элементы, которые реализуют простейшую логическую операцию (бу-

левую функцию) над входными сигналами. К таким логическим операциям относятся:

- повторение;
- логическое отрицание (НЕ);
- логическое сложение или дизъюнкция (ИЛИ), описываемое функцией $F(x_1, x_2) = x_1 + x_2$;
- логическое умножение или конъюнкция (И), описываемое функцией $F(x_1, x_2) = x_1 * x_2$;
- импликация;
- равнозначность или эквивалентность (исключающее ИЛИ-НЕ);
- неравнозначность или сложение по модулю 2 (исключающее ИЛИ);
- операция Пирса (ИЛИ-НЕ);
- операция Шеффера (И-НЕ).

Основная идея метода состоит в том, чтобы, используя оцифрованные микросхемы более низкого уровня иерархии, которые описываются простейшими математическими моделями, и программные возможности сред проектирования, создавать программные модели элементов более высокого уровня иерархии, на основе их структурной декомпозиции и взаимосвязей простейших элементов в них. При этом элементы высокого уровня иерархии описываются значительно более сложными математическими моделями.

Рассмотрим интеграционный способ моделирования на примере триггера *1533ТМ2*, на рисунке (рис. 1.4) представлено схемное решение этого элемента.

Как видно, рассматриваемый элемент (рис. 1.4) состоит из шести объектов класса «логический элемент», представляющих собой низший уровень

иерархии. Данные компоненты с тремя входами и одним выходом, реализуют операцию Шеффера (И-НЕ). Стоить отметить, что данные компоненты по структуре и функциональности идентичны микросхеме *1533ЛА4.* Математическая модель, этих элементов представляет собой булеву функцию

$$o = \overline{i_1 * i_2 * i_3}. \tag{1.2}$$

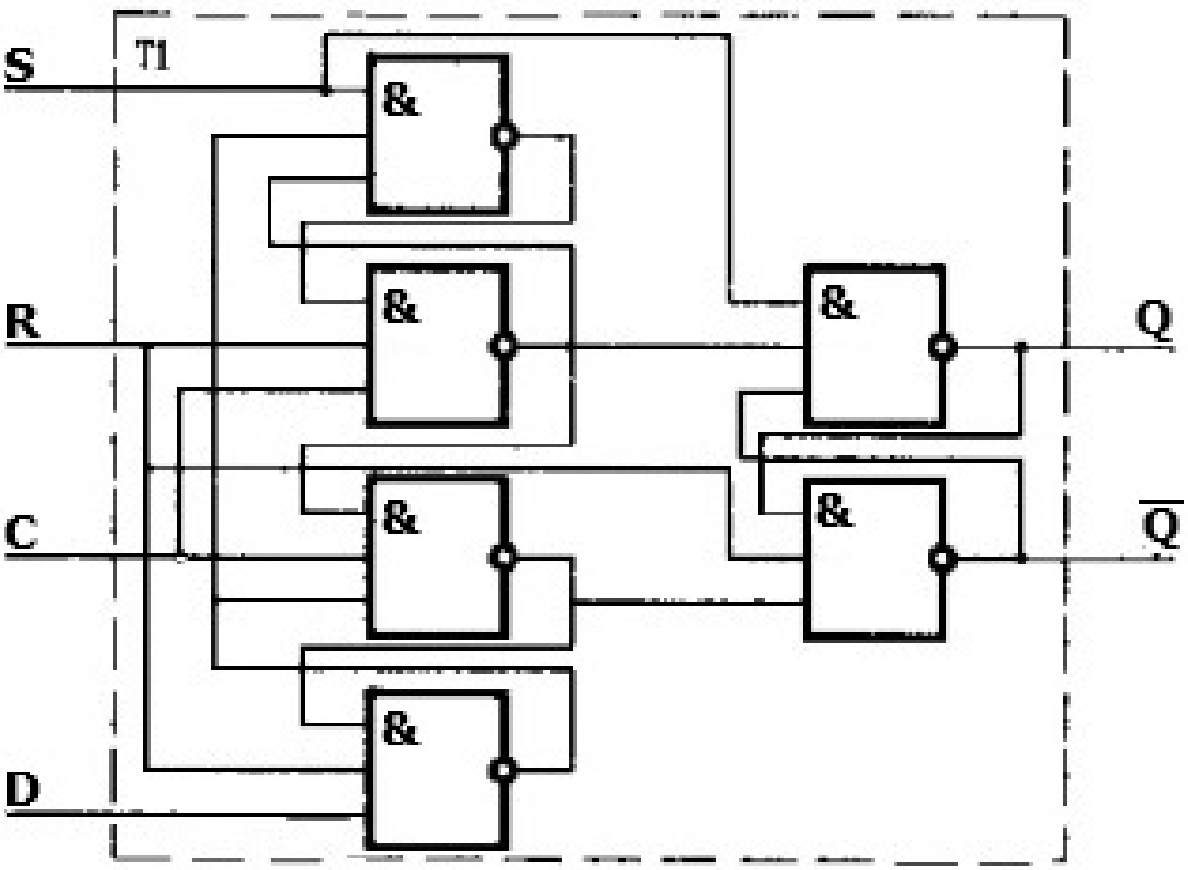

Рис. 1.4. Функциональная схема триггера *1533ТМ2*

Программно реализуем данную математическую модель и получаем программную модель (Verilog design file) и ее файл графической интерпретации bsf-файл (рис. 1.5) в среде Altera Quartus II.

Используя программную модель этого элемента, файл его графической интерпретации (bsf-файл) (рис. 1.5) и готовые примитивы среды Altera Quartus II, инициализирующие входные и выходные разъемы микросхемы, в директории Block Diagram/Schematic File в соответствии с архитектурой схемного решения располагаем модели компонентов и организуем их взаимосвязь. На

рисунке (рис. 1.6) представлен цифровой вид функциональной схемы элемента $1533ТМ2$.

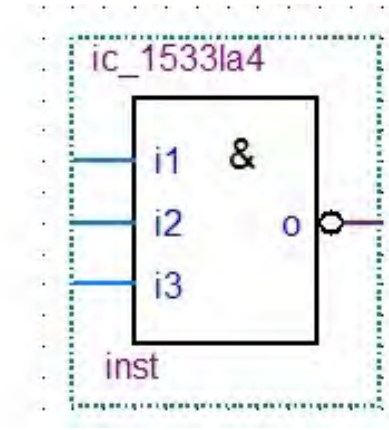

Рис. 1.5. Визуальное представление программной модели логического элемента цифровой микросхемы *1533ЛА4*

С помощью программных средств среды Altera Quartus II, осуществляем компиляцию «оцифрованного» схемного решения и последующую генерацию программной модели (Verilog design file). Стоит отметить, что программная модель (модуль) не может включать описание других моделей, но может содержать ссылки на другие модули как на свои структурные компоненты, что показано в программном коде представленном в приложении Д.

К преимуществам данного метода можно отнести:

- гарантированное получение синтезабельной модели (при поддержке строгой иерархии и реализации моделей низших уровней вероятность появления логической ошибки минимальна);
- использование простейших математических моделей, гарантирующих простоту реализации;
- отсутствие потребности в сборе информации об алгоритмах работы микросхемы и ее функциональных особенностях, достаточно наличие функциональной схемы элемента;
- отсутствие необходимости в высоких компетенциях в схемотехнике и навыков владения языком Verilog;

- возможность учесть особенности алгоритма работы элемента, не указанные в описании, но отраженные в схемном решении.

Недостатками являются:

- необходимость наличия базы реализованных компонентов более низкого уровня иерархии, для моделирования микросхем более высокого уровня;
- сложно воспринимаемая структура программного кода;
- времязатратность метода.

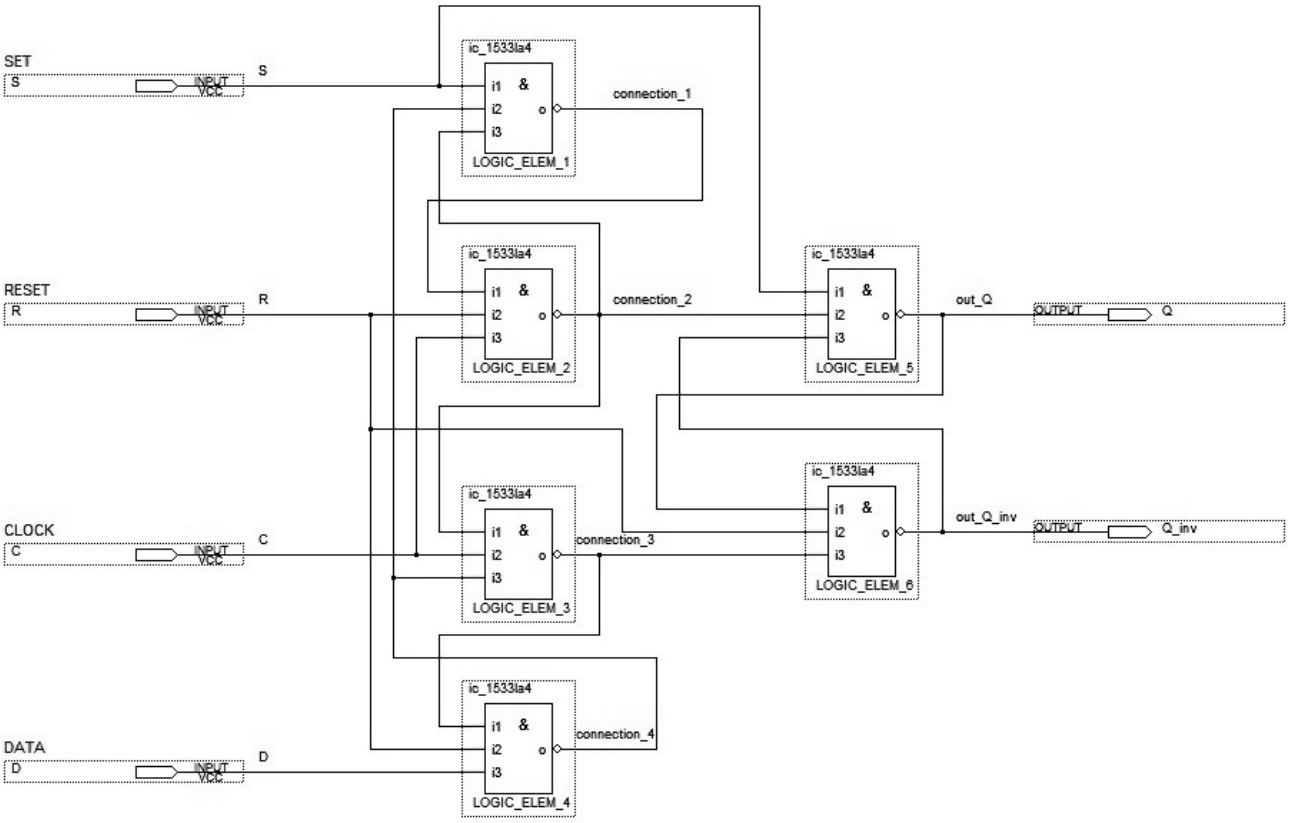

Рис. 1.6. Визуальное представление, функциональной схемы элемента *1533ТМ2* в цифровом виде.

1.4 Правила наименования компонентов и сигнальных цепей

Наименование типа компонентов не может начинаться с цифры и не должно содержать буквы русского алфавита и знаки препинания, поскольку

такие имена запрещены языком Verilog. Поэтому наименование типа компоненты должно начинаться с символьного префикса, за которым следует оригинальное имя компоненты. Для работы с системой рекомендуется использовать префиксы "*ic_*" и "*icp_*". Правила перевода русских символов в латиницу определяются пользователем.

Префикс "*ic_*" означает, что схема устройства создана с отображением в каждой микросхеме только внутренних, независимых друг от друга, функциональных частей. При этом модель микросхемы не поддерживает распределение сигналов по ножкам микросхемы, а описывает функциональность независимой части микросхемы. Иными словами, каждая независимая часть микросхемы представляется на схеме как отдельный компонент.

Префикс "*icp_*" означает, что схема устройства создана с отображением микросхемы как таковой. В этом случае модель микросхемы поддерживает распределение сигналов по ножкам микросхемы. Например, микросхема *133LA3* состоит из четырех независимых двухвходовых схем *И-НЕ*. На рисунке (рис. 1.7) приведена часть схемы, использующая обычное отображение составляющих частей микросхемы, характерное для САПР PCAD.

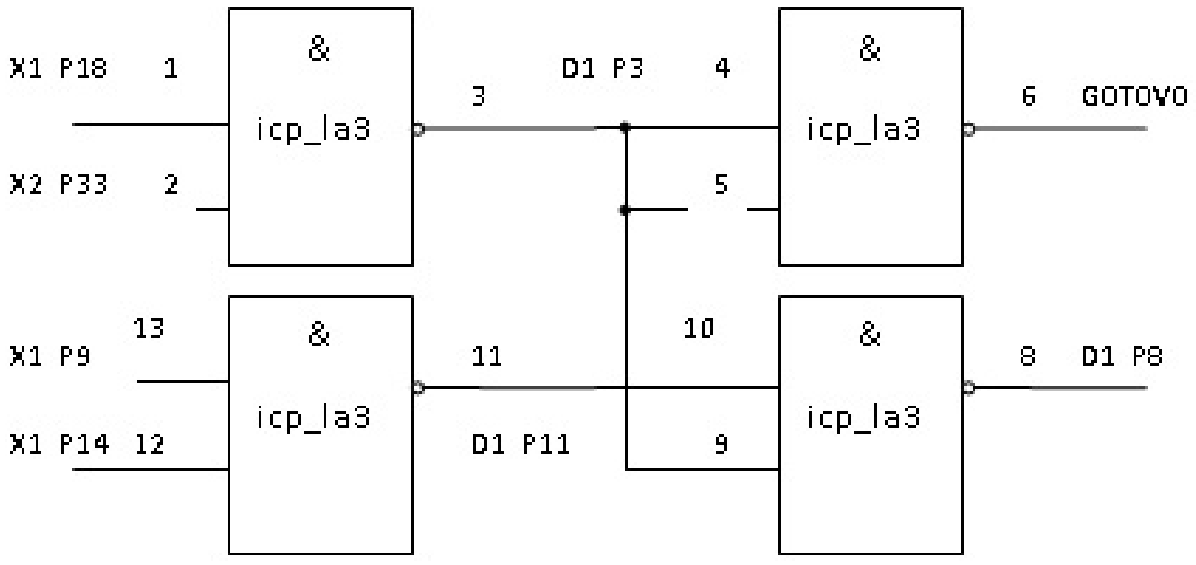

Рис. 1.7. Пример схемы с использованием префикса *icp_* в наименовании типа компонента

В этом САПР принадлежность функциональных частей микросхемы определяется по перечислимым именам, т.е. если имеется обозначение $D1.3$, это означает, что это третья независимая функциональная часть микросхемы с обозначением $D1$, а тип ее взят из библиотеки примитивов САПР. Расстановка номеров контактов частей микросхемы, в этом случае, происходит автоматически и берется из библиотеки примитивов. Verilog модель микросхемы *133LA3* в этом случае выглядит так:

```
module icp_1331a3(P1, P2, P3, P4, P5, P6, P8,
            P9, P10, P11, P12, P13);
input P1, P2, P4, P5, P9, P10, P12, P13;
output P3, P6, P8, P11;

assign P3 = ~(P1 & P2);
assign P6 = ~(P4 & P5);
assign P8 = ~(P9 & P10);
assign P11 = ~(P12 & P13);
endmodule
```

Модель описывает всю микросхему со всеми ее функциональными частями, а сигналы привязаны к контактам микросхемы. Контакты микросхемы в этой модели имеют символические имена, начинающиеся на символ *"P"*. Ниже (рис. 1.8) приведена та же часть схемы, но использующая отображение только функциональных частей микросхемы, каждая из которых рассматривается как отдельный компонент. Такое представление характерно для САПР Quartus. В этой САПР однотипные функциональные части микросхем рассматриваются как отдельный компонент, имеющий уникальное в рамках всей схемы устройства перечислимое имя. Это имя не должно содержать символов знаков препинания. Поэтому перечислимое имя рекомендуется составлять также как и ранее, заменяя символ точки на символ нижнего подчеркивания.

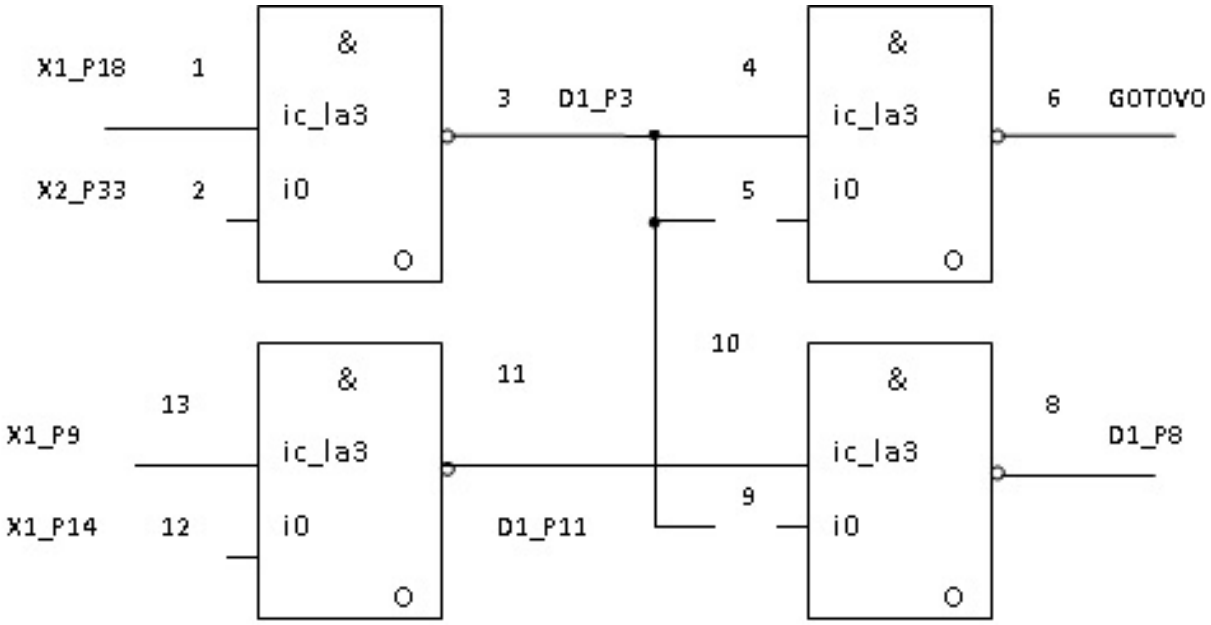

Рис. 1.8. Пример схемы с использованием префикса $ic\_$ в наименовании типа компонента

Следует отметить, что номера контактов микросхемы задаются здесь как комментарии (текстовые надписи) и не оказывают влияние на последующее моделирование. Вместо номеров контактов при моделировании используются функциональные названия сигналов такого компонента. В этом примере это имена $i0$, $i1$ и O. Verilog модель микросхемы *133LA3* в этом случае выглядит так:

```
module ic_133la3(i0, i1, O);
input i0, i1;
output O;
assign O = ~(i0 & i1);
endmodule
```

Контакты микросхемы в этой модели не присутствуют. Система поддерживает оба способа наименования компонентов. Для этого необходимо, чтобы используемые модели компонентов были включены в соответствующую базу данных системы. Смешанное наименование компонентов в рамках схемы одного устройства не допускается.

В примерах (рис. 1.7 и рис. 1.8) показаны также названия сигнальных цепей схемы, сформированных по правилам, о которых будет сказано ниже. Как уже отмечалось, все сигнальные цепи должны быть поименованы. В противном случае САПР разработки схем будет именовать цепи автоматически, по своим правилам. САПР PCAD именует все неподписанные цепи в своем порядке в виде $NET00$, $NET001$ и т.д., а САПР Quartus присваивает наименования вида $SYNTHESIZED\_WIRE\_0$, $SYNTHESIZED\_WIRE\_1$ и т.д. При формировании топ модели устройства наименования всех цепей сохранятся. Очевидно, что с такими именами цепей разбираться при отладке, как теста, так и при просмотре временных диаграмм на реальном устройстве крайне неудобно. Для работы с системой проектирования тестов рекомендуются следующие правила именования цепей.

- Имя должно содержать только символы латинского алфавита, символ нижнего подчеркивания и не должно начинаться с цифры. Это требование обязательно.

- Цепь, имеющая символьное имя в исходной схеме, должна иметь по возможности такое же имя и в скорректированной. В случае, если имя написано кириллицей, то его необходимо записать латиницей.

- Все оставшиеся цепи именуются по составному принципу. Первая часть носит перечислимое имя компонента, с выхода которого она берет свое начало, затем следует символ нижнего подчеркивания и латинская буква "P", а затем номер контакта компонента от которого она отходит. Т.е. формат имени имеет следующий вид «Перечислимое имя компонента»$\_P$ «номер выходного контакта компонента». Цепи именуются только по выходам соответствующих компонентов.

На рисунках (рис. 1.7 и рис. 1.8) имена цепей сформированы по этим правилам. Например, имя цепи $X1\_P33$ означает, что она начинается на 33 контакте компонента $X1$, возможно это цепь идущая с краевого разъёма $X1$.

Цепь с именем $D1\_P3$ - это выход микросхемы $D1$ контакт 3. Как видно из изображения, эта цепь соединяет выход $D1$, её третий контакт со входными контактами 4, 5, 9, этой же микросхемы. На схеме также присутствует цепь с именем "GOTOVO" выходом микросхемы $D1$ контакт 6. Если бы она не была поименована на исходной схеме, то ей следовало бы дать имя "$D1\_P6$".

Если придерживаться этих правил наименования цепей, то достаточно быстро можно найти интересующие точки на схеме устройства, на визуализированных временных диаграммах и в тексте моделей устройства, в ходе работ по наладке теста и устройства.

1.5 Метод интегрирования аналоговых узлов в модель цифрового объекта контроля при его моделировании

Довольно часто в цифровых объектах контроля встречаются различные аналоговые компоненты. В большинстве случаев их необходимо интегрировать в программную модель объекта контроля. Для этого необходимо, учитывая функциональность аналоговых элементов, их предназначение и роль [114–121], создать математическую модель этих компонентов, описывающую их влияние на изменение характеристик сигнала при прохождении через эти аналоговые компоненты. Затем, в ходе моделирования цифрового устройства, программно создаются «мнимые (виртуальные)» элементы, заменяющие функциональный узел аналоговых компонентов, исходя из их математической модели.

Наиболее часто приходиться сталкиваться со следующими аналоговыми устройствами:

- фильтры по питанию;
- стабилизаторы напряжения;

- привязывающие сопротивления и согласующие делители на сопротивлениях;
- задерживающие емкости и RC цепочки;
- монтажные И / ИЛИ;
- реализованные с помощью аналоговых компонентов функциональные цифровые узлы;
- аналого-цифровые или цифро-аналоговые узлы.

При корректировке схемы фильтры по питанию и стабилизаторы напряжения просто исключаются из представления схемы (рис. 1.9), поскольку они не оказывают влияния на поведенческую модель устройства.

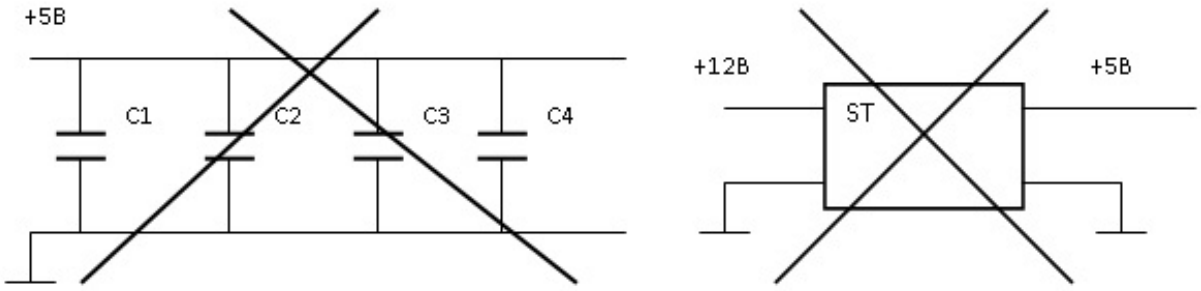

Рис. 1.9. Удаление фильтра по питанию и стабилизатора напряжения

Сопротивления, подключенные к питанию для раздачи в схеме уровня логической единицы, необходимо заменить на непосредственное подключение к линии логической единицы VCC (рис. 1.10).

Сопротивления, подключенные к питанию обеспечивающие привязку одиночного выхода компоненты с открытым коллектором необходимо исключить (рис. 1.11), а в моделях этих компонентов предусмотреть выход в виде обычного логического сигнала.

Согласующие делители на сопротивлениях, подключенные к выходам компонентов с открытым коллектором или к выходам компонентов с третьим состоянием, необходимо исключить (рис. 1.12).

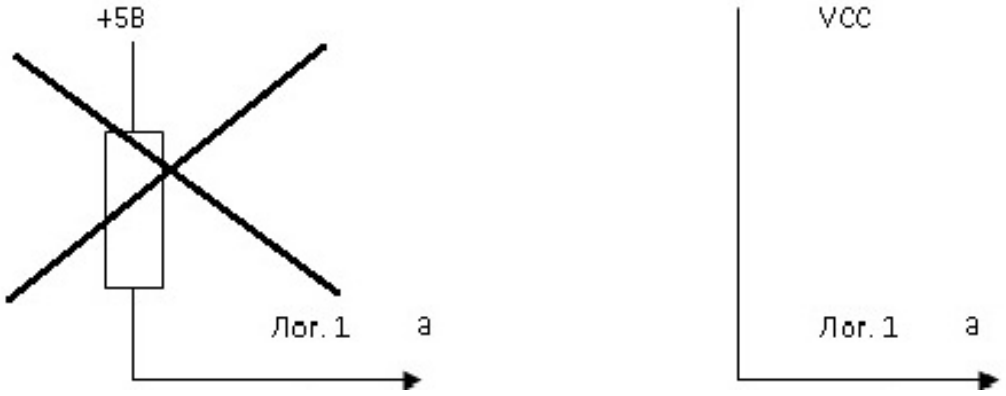

Рис. 1.10. Замена привязывающего сопротивления на прямое подключение

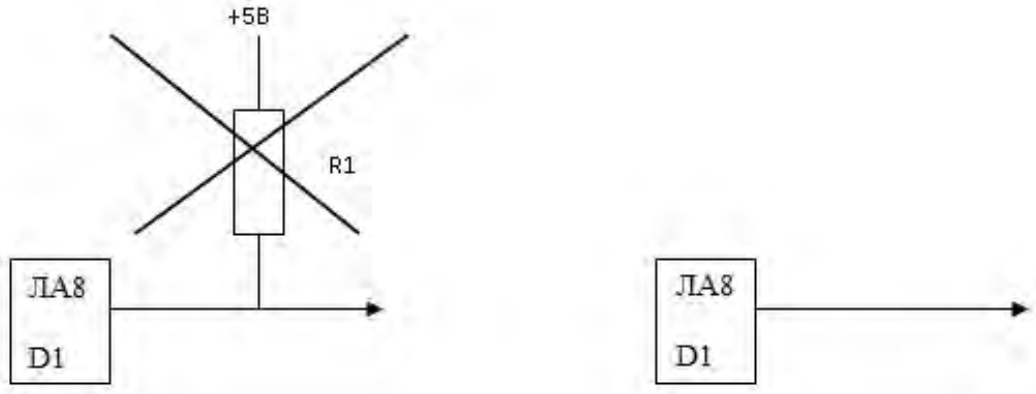

Рис. 1.11. Исключение привязывающего сопротивления для схем с открытым коллектором

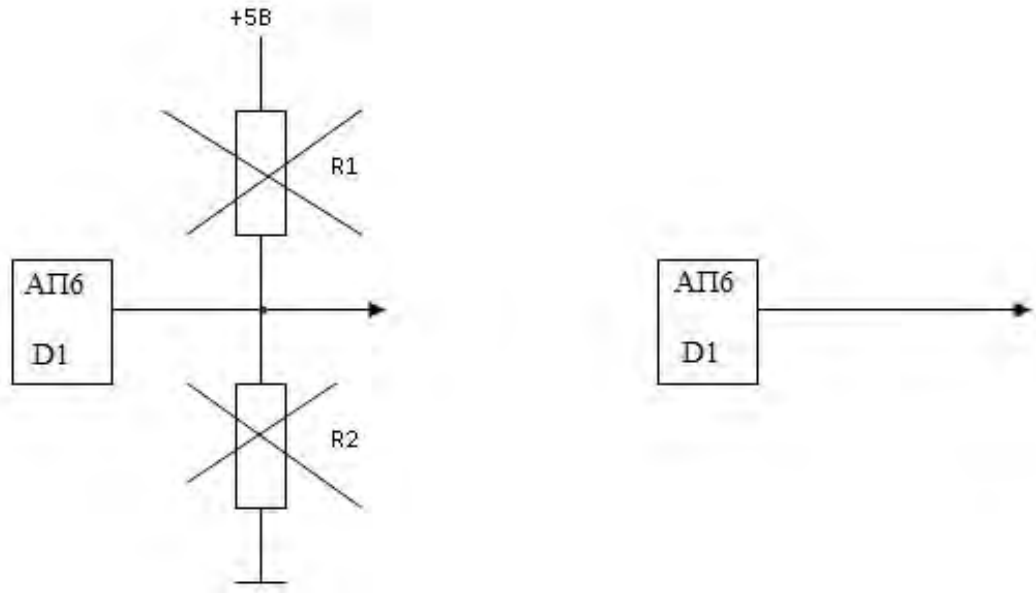

Рис. 1.12. Исключение согласующего делителя для компонентов с открытым коллектором или схем с третьим состоянием

Цепи, в которых встречаются задерживающие емкости и задерживающие RC цепочки, необходимо заменить на эквивалентные этим задержкам

фиктивные элементы "Delay" (рис. 1.13). При этом необходимо добавить в библиотеку компонентов соответствующую модель этого элемента. Величина задержки определяется расчетным путем.

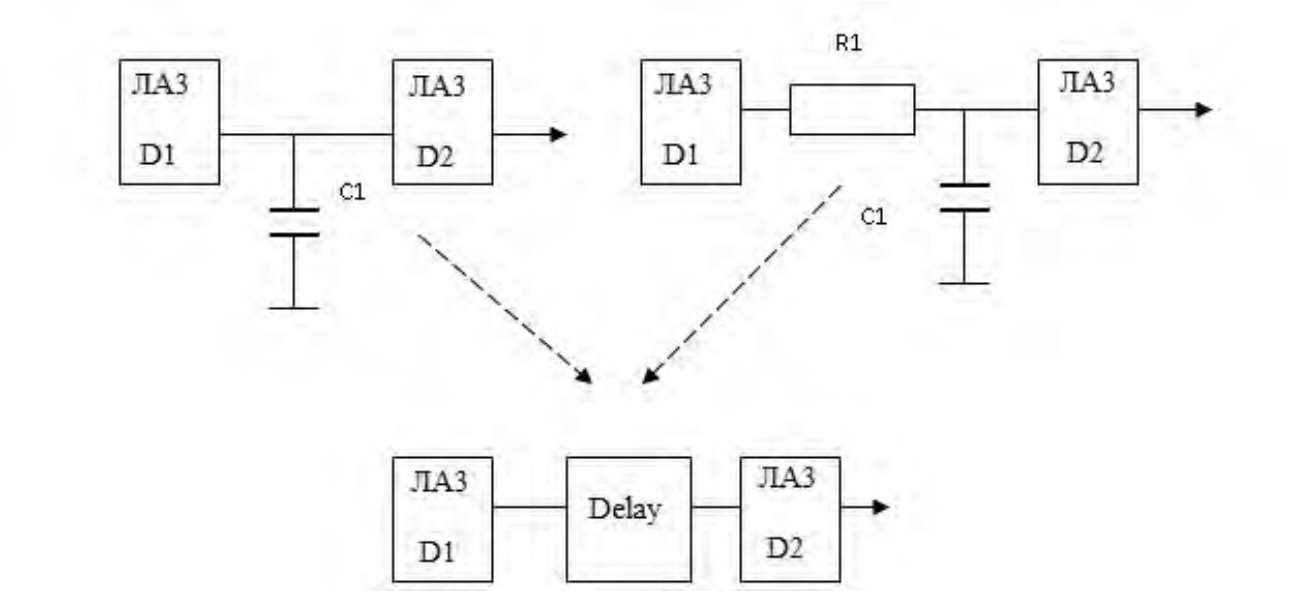

Рис. 1.13. Замена задерживающих цепочек на фиктивные (виртуальные) элементы "Delay"

Монтажные И/ИЛИ как правило объединяют выходы компонентов с открытым коллектором. Их необходимо заменить на соответствующий фиктивный элемент, осуществляющий эту функцию (рис. 1.14).

Реализованные с помощью аналоговых компонентов функциональные цифровые узлы отличаются большим разнообразием, поэтому далее приводятся только некоторые примеры таких замен. В каждом конкретном случае необходимо изучить функциональность реализованного узла и разработать его математическую модель, а также его графическое представление. В схеме этот функциональный узел заменяется с помощью используемой САПР разработки схем на его графическое представление.

Рассмотрим в качестве примера управляемый одноразрядный повторитель цифрового сигнала с третьим состоянием. На рисунке (рис. 1.15) приведена его схемотехническая реализация, взятая из принципиальной схемы устройства. Этот управляемый повторитель реализован на базе шинного формирователя *1102АП8*($D50$), схемы И-НЕ с открытым коллектором *539ЛА9*

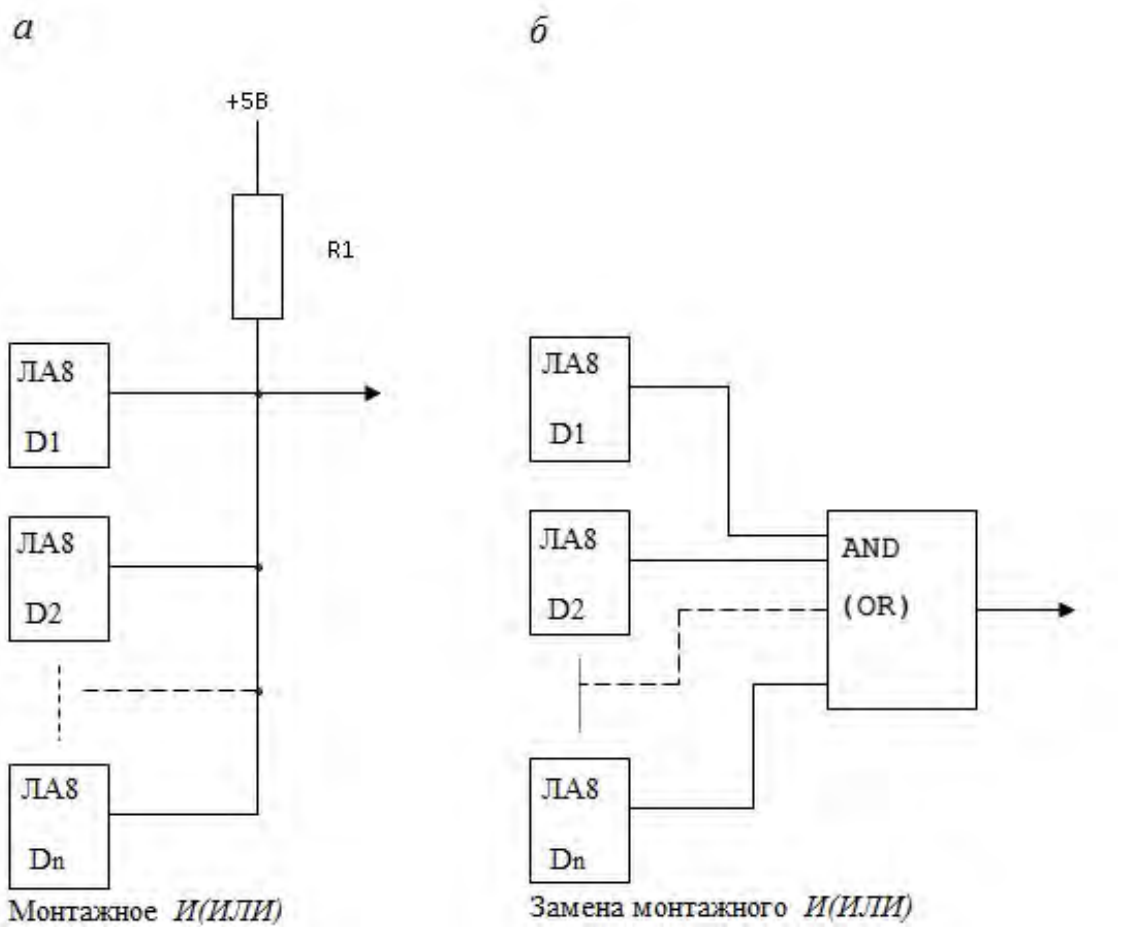

Рис. 1.14. Замена монтажного И (ИЛИ) на фиктивный элемент $AND(OR)$

$(D51)$, транзистора, трех резисторов и фильтрующего конденсатора. Из анализа этого узла можно получить следующую математическую модель:

$$\begin{aligned} s\_output &= s\_input, \quad \text{при} \quad s\_control = 0, \\ s\_output &= 1'bz, \quad \text{при} \quad s\_control = 1. \end{aligned} \tag{1.3}$$

Здесь $s\_control, s\_input$ — соответственно управляющий и информационный входные сигналы фиктивного элемента, $1'bz$ — состояние высокого импеданса, s_output — выход фиктивного элемента. При коррекции исходной схемы устройства рассмотренный узел заменяется на фиктивный элемент FVT, выполняющий те же функции. Вид этого элемента также приведен на рисунке (рис. 1.15). Для осуществления коррекции необходимо создать соответствующий компонент в САПР для разработки электронных схем и заместить им реальные узлы в схеме устройства.

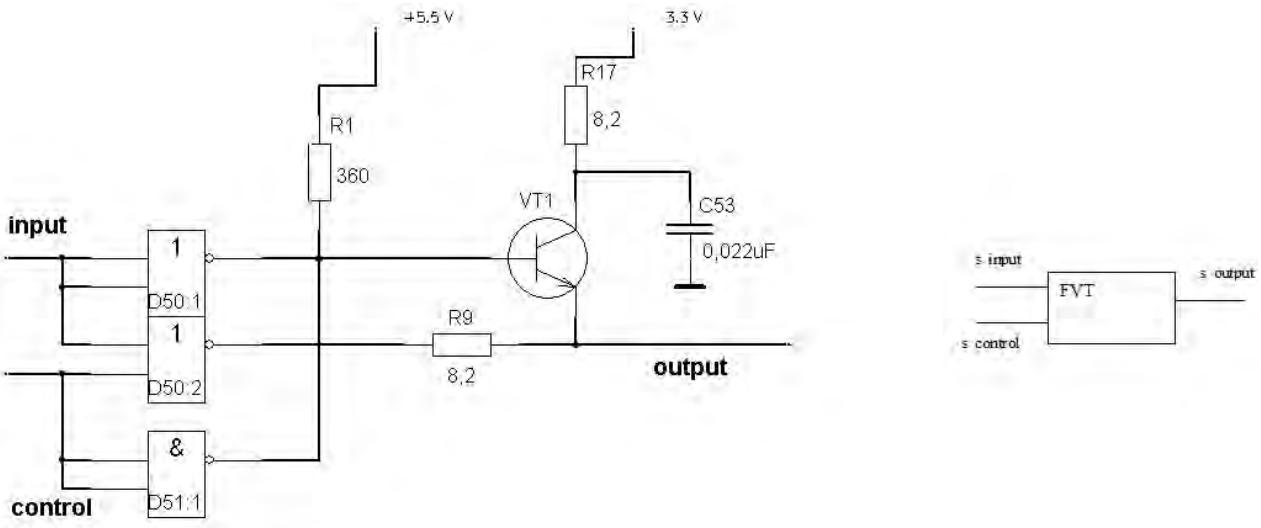

Рис. 1.15. Замена управляемого повторителя реализованного на нецифровых компонентах на фиктивный элемент FVT

В другом примере в схеме устройства используется компаратор $554CA4$ (рис. 1.16), который сравнивает величину уровня логического сигнала с опорой и вырабатывает напряжение $+5B$, если уровень сигнала больше опоры или $0B$ в противном случае. Полностью смоделировать такую функциональность в цифровом представлении невозможно, однако, для случаев, когда входной сигнал удовлетворяет требованиям на величину логических уровней «0» и «1», компаратор можно рассматривать как устройство просто как повторитель логического сигнала. В этом случае математическая модель данного аналогового узла будет выглядеть следующим образом:

$$P11 = P3. \tag{1.4}$$

В старых устройствах, которые до сих широко используются, цифровые переключатели реализовывались на транзисторах. В качестве еще одного примера рассмотрим переключатель с третьим состоянием, реализованный на двух транзисторах, на базы которых подаются цифровые сигналы. Этот переключатель и его функциональная замена FDR показана рисунке (рис. 1.17).

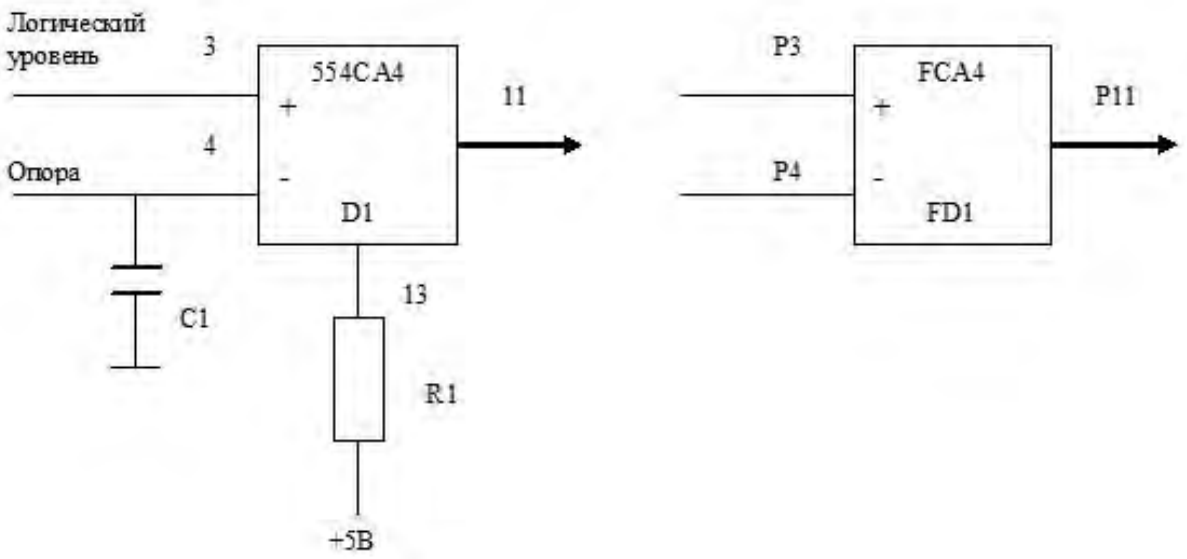

Рис. 1.16. Замена компаратора $554CA4$ на фиктивный элемент $FCA4$

Математическая модель, описывающая функционал данного блока выглядит следующим образом:

$$\begin{aligned} out &= 1, && \text{при} \quad in1 = 1, in2 = 0, \\ out &= 1'bz, && \text{при} \quad in1 = 0, in2 = 0, \\ out &= 0, && \text{при} \quad in1 = 0, in2 = 1, \\ out &= 0, && \text{при} \quad in1 = 1, in2 = 1. \end{aligned} \tag{1.5}$$

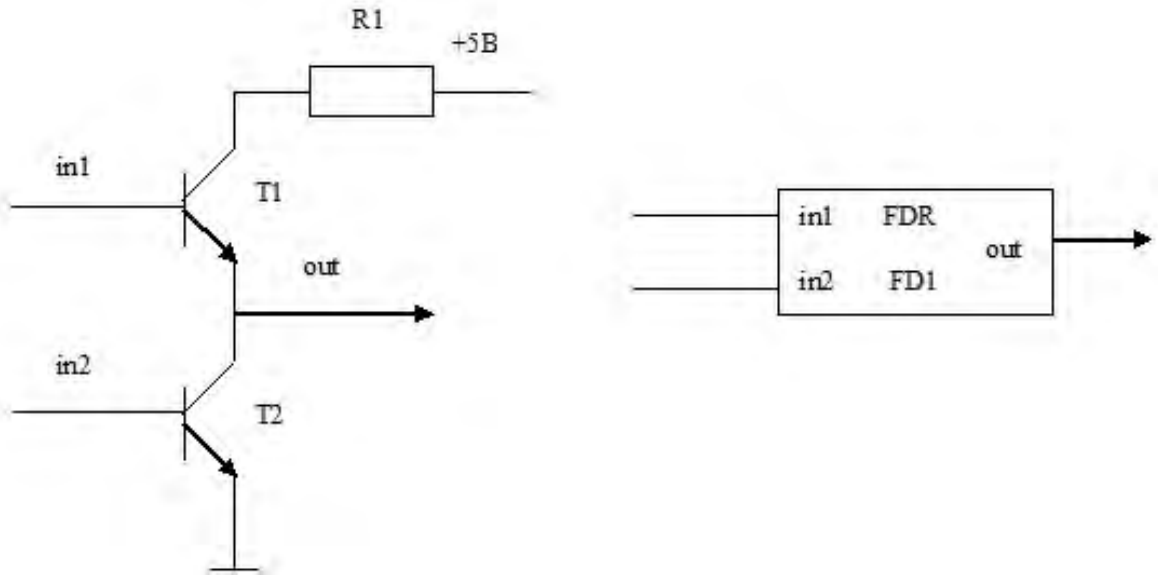

Рис. 1.17. Замена транзисторного переключателя на фиктивный элемент FDR

К сожалению, аналого-цифровые (АЦП) и цифро-аналоговые (ЦАП) преобразователи, присутствующие в устройствах, не могут быть заменены на фиктивные компоненты. Поэтому при коррекции схемы устройства для ЦАП необходимо предусмотреть дополнительные выходы, соединенные с цифровыми входами ЦАП с тем, чтобы можно было проконтролировать сигнал, подаваемый на ЦАП. Сам тест надо строить так, чтобы при подаче сигнала на ЦАП можно было измерить вырабатываемые им напряжения.

Для АЦП необходимо предусмотреть дополнительные выходы, соединенные с цифровыми выходами АЦП с тем, чтобы можно было проконтролировать сигнал, вырабатываемые АЦП при подаче на его аналоговый вход различных напряжений. Сам тест надо строить так, чтобы при подаче на его вход последовательности различных напряжений можно было проконтролировать вырабатываемую последовательность и промоделировать поведение цифровой части схемы с этой последовательностью.

1.6 Выводы по главе 1

В главе 1 проведен анализ основных методов и способов тестирования цифровых радиоэлектронных устройств, описаны ключевые особенности каждого из них. Также произведен сравнительный анализ готовых программных продуктов для моделирования аналоговых и цифровых радиоэлектронных систем, которые используются при осуществлении тестового контроля. Сделан вывод о том, что данные средства дают возможность анализировать поведение устройства при подаче конкретных стимулов. Однако, они не позволяют автоматизировать процесс формирования моделей таких тестовых воздействий, которые бы обеспечивали адекватный контроль и диагностику сложных цифровых радиоэлектронных систем.

Сделан вывод о необходимости автоматизации процесса формирования моделей тестовых воздействий, которые бы обеспечивали адекватный контроль и диагностику сложных цифровых радиоэлектронных устройств.

Сделан вывод о том, что с учетом специфики задач ремонта и тестового контроля цифровых радиоэлектронных устройств, уровень детализации поведенческой модели вплоть до компонентов достаточен для эффективного решения рассматриваемых задач. Созданы методы формирования моделей цифровых устройств (функциональный и интеграционный) с учетом особенностей задачи тестового контроля, для каждого метода разобраны преимущества и недостатки.

Разработан метод внедрения аналоговых узлов в имитационную модель объекта контроля на основе математических моделей. Детально разобраны наиболее часто встречающиеся случаи, такие как компараторы, транзисторные переключатели и т.п.

Глава 2

Разработка автоматизированной генерации проверяющей тестовой последовательности для задач тестового контроля цифровых устройств

2.1 Технология тестового контроля цифровых радиоэлектронных устройств

Тестом цифрового устройства будем называть совокупность заранее определённых воздействий на его входы и анализ соответствующих ответов устройства на выходах. Существует несколько способов формирования такого набора входных и выходных воздействий [44–49, 101].

В настоящее время как для тестового контроля качества производимых цифровых радиоэлектронных устройств, так и для их ремонта широко применяются специализированные установки тестового контроля. Данное оборудование, используя краевые разъемы, взаимодействуют с цифровыми объектами контроля. При этом на входные контакты устройства подаются соответствующие воздействия и фиксируется отклик устройства на эти воздействия. Совокупность поданных с помощью установки входных сигналов и результат реакций устройства на них является тестом устройства. Получить такую совокупность можно различными путями.

Самый распространенный вариант – формирование совокупности на основе проверки работоспособности основных и главных функциональных узлов устройства. Входные воздействия при этом задаются, как правило, непосредственно самими разработчиками устройства, таким образом, чтобы наиболее полно проверить функционал устройства.

В случае же, если устройство проектировалось и разрабатывалось с использованием системы автоматизации проектирования электронных устройств (САПР), то реакции устройства в таком случае могут быть получены с использованием этой же САПР.

Также широкое распространение получил способ формирования проверяющей последовательности путем снятия с краевых разъемов изделия реакций на входные воздействия в процессе функционирования устройства в штатном режиме («слепой тест»). Этот способ также достаточно трудоемок, и при этом редко применим, в связи с отсутствием возможности подключиться к разъемам устройства, которое функционирует в составе более крупного модуля.

В основе третьего способа создания теста лежит математическое компьютерное моделирование. Благодаря тому, что устройство известно и имеет четкую структуру, модель представляет собой схему взаимосвязей всех ее компонентов, которые входят в ее состав (Netlist) и соответственно программных моделей самих компонентов. При этом входные воздействия могут быть представлены как соответствующие программные модели. Данные модели реализуются с использованием специализированных языков, таких как Verilog и VHDL. Эти языки предоставляют инструментарий для детального описания функционирования проектируемых устройств. После завершения этапа моделирования наступает фаза тестирования устройства, в ходе которой проводится проверка его поведения при различных входных воздействиях [1–12,39–43].

Следует отметить, что при проведении тестирования важно учитывать, что в большинстве случаев этот процесс осуществляется вручную. Разработанная система автоматизированного тестирования цифровых устройств CRIT [13], основана на использовании математических методов моделирования радиоэлектронных систем и синтеза входных воздействий (входных сигналов). Она автоматизирует процесс поиска оптимальной последовательности входных сигналов, считая удачной тестовую комбинацию, которая активизирует

компоненты и все возможные взаимосвязи устройства, вызывая соответствующие реакции на выходе устройства в процессе тестирования.

В качестве числовой характеристики, определяющей успешность проверяющей последовательности, выступает тестовое покрытие P. Под ним понимается процентное отношение активированных сигнальных линий N_{act}, в процессе проведения теста, к полному количеству всех сигнальных линий N, которые содержатся в устройстве.

$$P = (N_{act}/N)100. \tag{2.1}$$

Важно отметить, что в системе критерием покрытия является покрытие по переключению. Это означает необходимость осуществления последовательного переключения сигнальной линии хотя бы один раз, что влечет за собой смену значения с логического «1» на логический «0».

В конечном итоге, при решении задачи тестирования, необходимо стремиться к тому, чтобы тестовое покрытие было максимально обширным.

Следует обратить внимание, что при тестировании комбинаторных схем с элементами реализующими булевы формулы, проверка по переключению недостаточна, так как простая проверка по переключению не исключит неисправности, вызванные ошибками монтажа или эксплуатации, а именно, слипание ножек, отрыв ножки, замыкание и т.п. Для этого в проверяющую тестовую последовательность должны быть включены следующие комбинации:

- «бегущий ноль»;
- «бегущая единица»;
- абсолютный ноль (подача по всем входным каналам микросхемы сигнала «0»);
- абсолютная единица (подача по всем входным каналам микросхемы сигнала «1»);

В частных случаях, когда изделие целиком и полностью состоит из микросхем, реализующих булеву логику, для решения задач тестирования можно использовать SAT-решатели, данный метод подробно был описан в статье Федюковича П. А., Елаева Е. В., Мащинского Н. С., Гришкина В. М.: «Формирование тестовых последовательностей с помощью SAT –решателя» [11].

Таким образом, автоматизация процедуры поиска тестирующей последовательности входных воздействий, способной обеспечить максимальное тестовое покрытие по всем сигнальным линиям, является главной задачей системы CRIT.

На рисунке (рис. 2.1) представлена технология разработки теста. Важно подчеркнуть, что рассматриваемая система охватывает все стадии создания теста, за исключением физического этапа отладки теста с применением инструментов тестового контроля.

Методология, описанная в первой главе, и тщательный анализ функциональности каждого элемента являются основой процесса разработки программных моделей устройств с использованием языка Verilog. Также для этого могут быть использованы как специальные программные средства, предназначенные для моделирования, так и сама разрабатываемая система [38].

В базе данных находятся модели компонентов устройства, которые были созданы. Она включает в себя множество различных устройств, при этом является частью интегрированной системы. Это позволяет при составлении тестовых сценариев для различных устройств обращаться к базе данных и использовать уже разработанные модели компонентов, обеспечивая эффективность и ускорение процесса тестирования.

Информация о цифровом устройстве часто ограничивается перечнем элементов, входящих в устройство, его принципиальной схемой, представленной в бумажном виде, и файлами «прошивки» элементов памяти. Для создания программной модели объекта контроля, как уже было сказано ранее, может быть использована любая специализированная среда проектирования

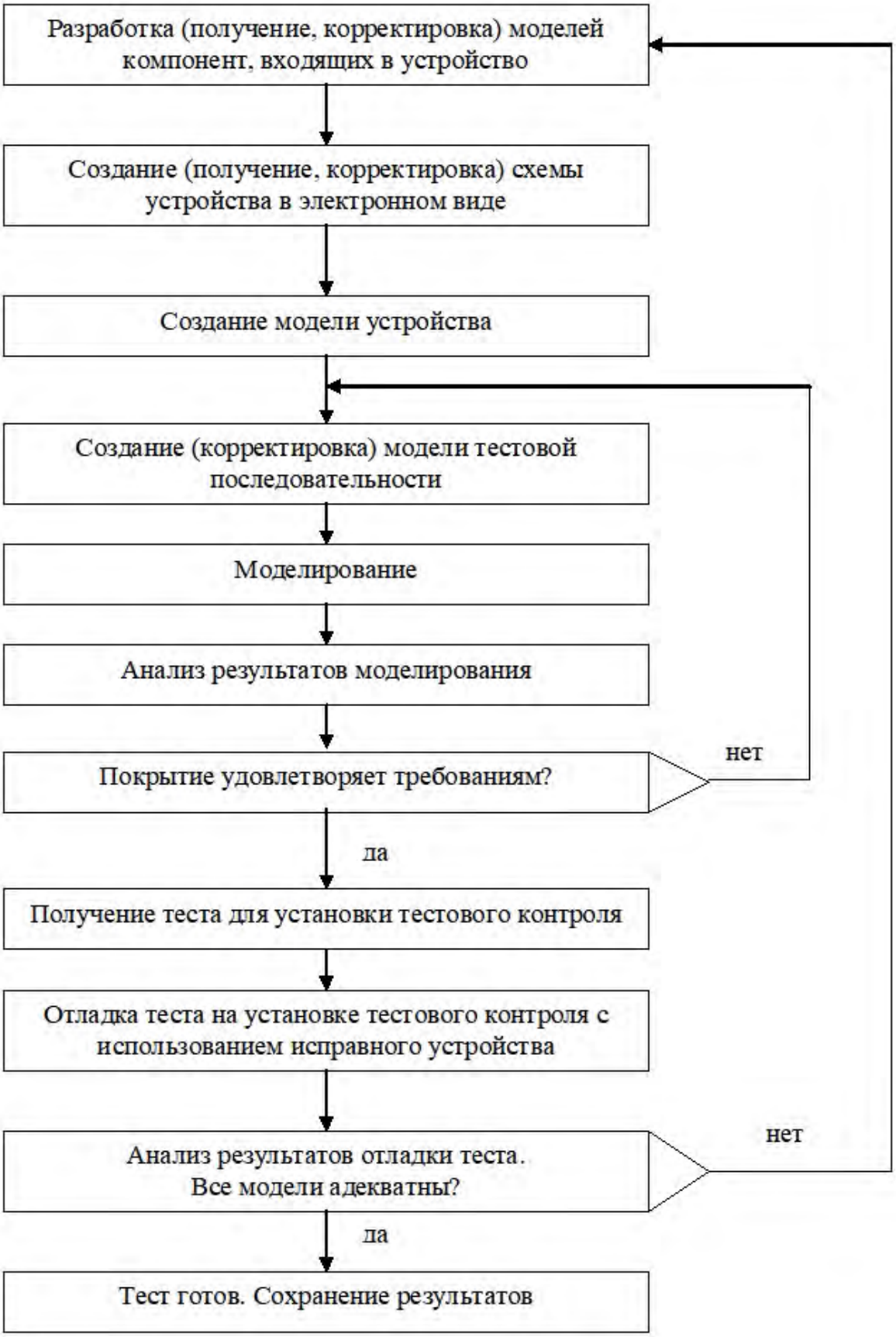

Рис. 2.1. Этапы построения тестов

ЦРЭУ, в данной работе используется среда проектирования Altera QuartusII, как наиболее удобная, развитая и обладающая всем необходимым инструментарием. Ключевым понятием, которым будем оперировать в работе, говоря о создании программной модели цифрового объекта контроля в рамках Altera QuartusII, является логический файл (Design File) — файл, который описывает алгоритм работы моделируемого устройства. Создание компьютерных моделей устройств, представляющих собой оцифровку принципиальной схемы объекта контроля, с использованием языка Verilog для описания моделей компонентов микросхем включает несколько ключевых этапов. Первоначально создаются математические модели компонентов микросхемы. Затем, используя специализированные программные средства, эти модели описываются на языке Verilog, что позволяет получить логические файлы модулей нижнего Low-Level Design (LLD) уровня.

Детализация на уровне LLD играет ключевую роль, позволяя создать подробное описание электронных модулей, являющихся составными частями цифрового объекта контроля. Эти файлы содержат исчерпывающую информацию, необходимую для понимания структуры и функций каждого модуля, что критически важно для обеспечения корректной работы всего устройства.

Последующим шагом является соединение этих модулей, организация их взаимосвязей, затем в результате процесса моделирования формируется модуль верхнего уровня иерархии (Top-level) [122]. В зависимости от сложности проекта и выбранного метода моделирования количество уровней может быть различным (то есть уже в их состав могут входить модули).

При переводе схемы устройства в цифровой вид мы решаем следующие задачи:

- создание математических моделей компонентов (эти модели служат основой для структурного описания устройства и позволяют провести детальный анализ его работы, при построении математических моделей компонентов устройства проводится тщательный анализ алгоритмов ра-

боты и типовых операций, характерных для используемых микросхем, такой подход обеспечивает высокую точность и достоверность результатов моделирования, что критически важно для выявления потенциальных проблем устройства);

- создание на основе разработанных математических моделей программных поведенческих модулей;
- отладка и коррекция созданных моделей в случае необходимости;
- сборка созданных моделей в единый программный модуль и его компиляция;
- отладка и коррекция программной модели объекта тестирования.

В результате, используя программные средства среды Altera QuartusII, получаем Verilog Design File (стандартное расширение - .v), файл (Netlist) , содержащий текстовое описание модуля на языке Verilog HDL.

При моделировании определяется реакция модели системы на заданные входные воздействия, т.е. моделируется поведение реальной системы, работающей с реальными сигналами. Если модели системы и модели входных воздействий адекватны, то результаты эмуляции будут адекватно отражать поведение реальной системы. При этом совокупность входных воздействий и реакций на них представляет поведение реальной исправной системы и может быть использована в качестве эталона для реализации соответствующего теста.

Этот тест может быть загружен в универсальную или специализированную аппаратуру тестового контроля, с помощью которой и проводится тестирование реальных цифровых радиоэлектронных систем. Таким образом, при данном подходе для создания контрольно–диагностических тестов нет необходимости изучать функциональность системы и её составляющих - достаточно сформировать «удачную» последовательность входных воздействий, на

которую реагировали бы все компоненты моделируемой системы, и активировались бы все связи между этими компонентами.

Система автоматизированно создает модель проверяющей тестовой последовательности, основываясь на интерфейсном методе (подробно этот метод описан в следующем разделе 2.3) и информации, которую получает пользователь в ходе работы с инструментами, входящими в состав системы. Система производит моделирование с использованием компилятора Icarus и при этом использует все доступные модели, извлеченные из базы данных.

Система анализирует файл результатов моделирования и вычисляет покрытие для всех уровней схемы. Решение о достаточности покрытия принимает пользователь. В случае достижения необходимого уровня покрытия система формирует тест устройства в формате пригодном для загрузки в установку тестового контроля. Последующие этапы работы над тест- программой осуществляются с использованием аппаратно-программного автоматизированного комплекса тестирования и диагностики цифровых электронных систем УТК-512 (рис. 2.2).

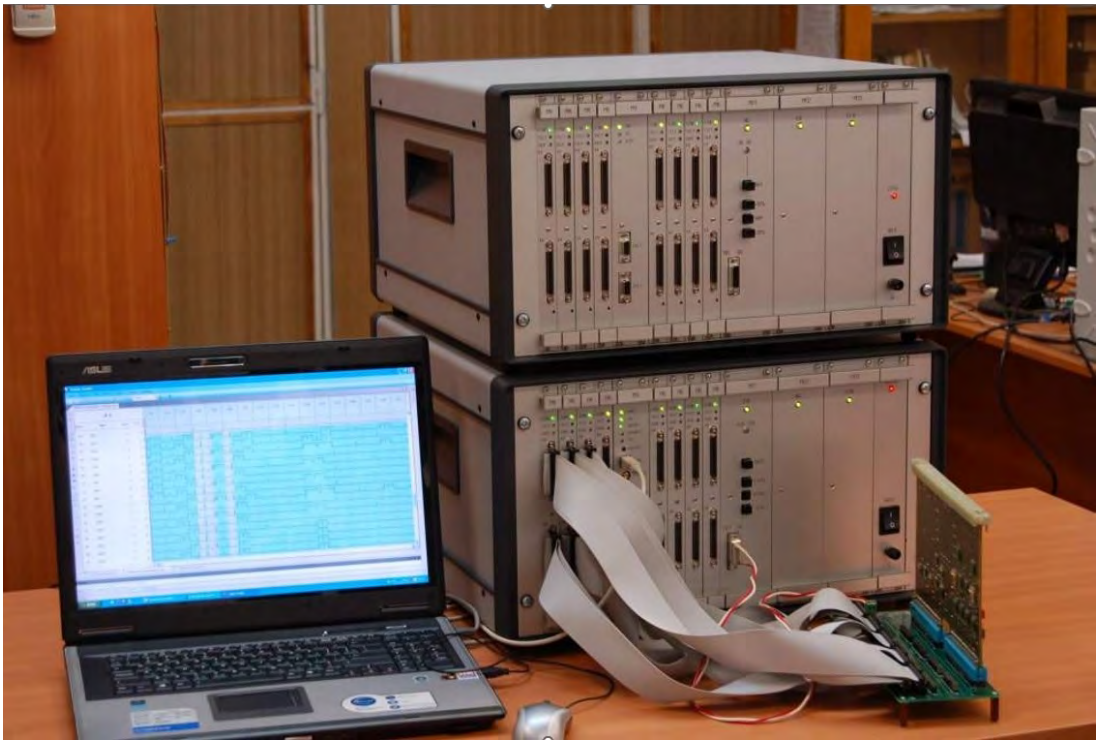

Рис. 2.2. Установка тестового контроля

На следующем этапе производят отладочные работы тестовой программы, которая будет проверена и видоизменена с помощью аппаратно-программного комплекса тестирования и диагностики цифровых электронных систем УТК-512 (рис. 2.2).

С помощью интегрированной среды ЯСТЕК осуществляется взаимодействие с УТК-512 и последующие отладочные работы. [8, 123]. При работе с тестами в данной среде существует возможность получать, передавать и принимать различные данные, проводить сравнение с эталонными значениями, а также отображать результаты в понятной и удобной для восприятия форме.

Схемное представление тестирования с применением комплекса УТК-512 показано на рисунке (рис. 2.3).

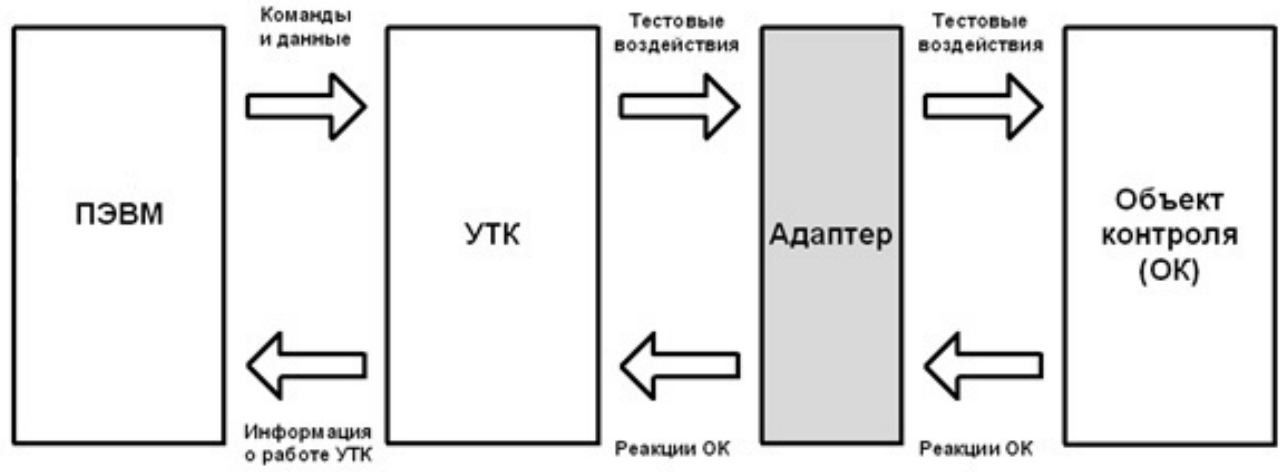

Рис. 2.3. Визуализация алгоритма тестирования с использованием комплекса УТК-512

Таким образом, САПР CRIT является основным средством для получения тест-программы, среда ЯСТЕК является необходимым дополнением, позволяющим провести отладку и дополнить тестовую программу с учетом отклика объекта контроля [8].

2.2 Подходы к моделированию объекта контроля с компонентами программируемой логики

Особое внимание следует уделит тому, что процессоры и другие компоненты программируемой логики (такие как ПЗУ или ПЛИС), которые не имеют конфигурирующей программы (так называемой «прошивки»), являются отдельным случаем при создании математических и программных моделей в рамках задачи тестового контроля [110, 112, 113]. В итоге системы, которые содержат данные элементы, не отвечают общим принципам тестирования и их невозможно проверить в рамках данной парадигмы. Следующие особенности рассматриваемых элементов, объясняют данный факт:

- большое количество подсистем (вентилей, триггеров и т.д.);
- наличие в памяти программы, которая подвергается изменениям и довольно часто недоступна для анализа;
- наличие у процессора многоуровневого автомата состояний.

В текущий момент существует несколько методов и подходов, направленных на разработку программных моделей процессоров или элементов с оперативной памятью (программируемой логикой) в контексте моделирования всего объекта контроля для формирования эффективных проверочных тестов. Ниже мы рассмотрим основные способы и подходы к созданию программных моделей процессоров [12].

1. Для процессора или ПЛИС элемента создается полная функциональная математическая и, как следствие, программная модель, для этого необходима полная доступная документации.

Плюсы:

- данный способ разрешает использование абсолютно любой «прошивки»;

- позволяет моделировать реакцию системы на все возможные входные сигналы.

Минусы:

- нужны огромные трудозатраты, чтобы создать и отладить модель, это сопоставимо с разработкой нового процессора;
- требуется доступ к полной документация.

2. Создание так называемой «усечённой» модели. В контексте осуществления тестирования всего изделия данный подход к моделированию предполагает анализ функциональности описанных выше элементов системы на предмет избыточности тех или иных функций и команд, которые входят в состав «прошивки». Это даёт возможность моделировать лишь ту часть функционала, которая может быть использована исключительно для тестирования.

Плюсы:

- устанавливается оптимальный баланс между полнотой и достаточностью модели, а также трудозатратами, необходимыми для ее создания;
- при сокращении избыточности функциональности, время моделирования уменьшается.

Минусы:

- подача, не совпадающих с задуманными, данных на входы модели может привести к неверным результатам;
- необходимо получить документы и информацию о назначении файла «прошивки», чтобы ограничить функционал в допустимых пределах.

3. Создание модели — «заглушки». Подобная модель представляет собой эмуляцию устройства, которое при любых получаемых на входе воздействиях последовательно (потактно) подает на выход строго определенную последовательность. В процессе моделирования данным способом, создается массив данных, которые будут подаваться на выход при подаче сигнала. Но при этом

не учитывается действительный алгоритм функционирования данного компонента.

Плюсы:

- значительно сокращается время, необходимое на создание модели и изучение документации, благодаря отказу от использования логики работы элемента;
- появляется возможность проверить выходы элемента на наличие «монтажных дефектов», таких как обрыв или замыкание.

Минусы:

- невозможно проверить исправность самого элемента (обычно у таких элементов есть встроенные возможности самодиагностики, поэтому предполагаем, что элемент можно протестировать отдельно с использованием таких технологий как JTAG);
- необходимо такое конструктивное решение , которое позволяло бы заменить программу «прошивки».

Отдельно следует отметить, что современные программируемые СБИС (Xlinx, Altera и им подобные) используемые для реализации того или иного функционала, программируются с помощью соответствующих функциональных и поведенческих моделей на HDL языках. Если разработчик открывает доступ к этим моделям или подробно описывает их функциональность, то изделия, содержащие такие компоненты, также можно протестировать в рамках предлагаемой технологии.

2.3 Интерфейсный метод автоматизированной генерации тестовых воздействий

При создании модели входных воздействий основной проблемой является поиск таких временных последовательностей входных сигналов устрой-

ства, которые приводили бы к активации всех компонентов объекта тестирования при моделировании его реакции на поданные воздействия и изменяли бы состояние выходных сигналов [4–11]. В работе предлагается интерфейсный метод автоматизированной генерации тестовых воздействий для ЦРЭУ [3, 5], в основе интерфейсного метода лежит дифференцированный подход к рассмотрению объекта контроля и составлению тестовой последовательности для него.

Для создания теста в рамках предложенного подхода на первом этапе необходимо провести изучение и анализ принципиальной схемы устройства. Затем в этом схемном решении, исходя из иерархии функциональной сложности элементов, логики их работы и способов взаимодействия с соседними элементами, выделяются типовые функциональные (структурные) «блоки», называемые так же логическими интерфейсами.

Элементы класса цифровых микросхем высокого уровня иерархии с наиболее сложным функционалом (преимущественно запоминающие устройства и последовательные микросхемы) (см. раздел 1.2) являются «основой» блока или «корневыми» элементами, а связанные с ними в рамках одного структурного блока элементы становятся управляющими (как правило это логические элементы и комбинационные микросхемы). В этом случае входные тестовые воздействия строятся на базе типовых операций присущих корневому элементу интерфейса, и тестовая последовательность подбирается таким образом, чтобы проверить работоспособность и функциональность элемента находящегося в основе блока. Следует отметить, что границы структурного блока (интерфейса) выбираются таким образом, чтобы выходы корневых элементов были выходами интерфейса. Каждый выделенный интерфейс в зависимости от класса корневого элемента имеет свои типовые тестовые наборы.

Каждый интерфейс тесно связан с физическими микросхемами, находящимися на верхнем уровне иерархии компонентной части системы (корневыми элементами). Эта связь проявляется как в особенностях функциониро-

вания интерфейса, так и в процессе создания тестовых последовательностей для проверки структурных блоков устройства. Разработка эффективных тестов требует глубокого понимания типовых операций и функций, выполняемых корневыми элементами системы.

Функциональные блоки (логические интерфейсы) , которые формируются из тех компонентов устройства, непосредственно связанных с входными краевыми сигналами устройства и которые можно объединить в функциональный блок – являются интерфейсами внешнего уровня (рис. 2.4). Управляющими сигналами интерфейса в таком случае и будут являться сигналы, полученные с входов напрямую, что довольно удобно.

Интерфейсами внутреннего уровня (рис. 2.4) называются блоки, которые непосредственно не связаны с входными сигналами и взаимодействует с ними через выходы выделенных предыдущих интерфейсов, через них и будет осуществляться подача тестового сигнала, т.е. опосредованно по отношению к входным разъемам.

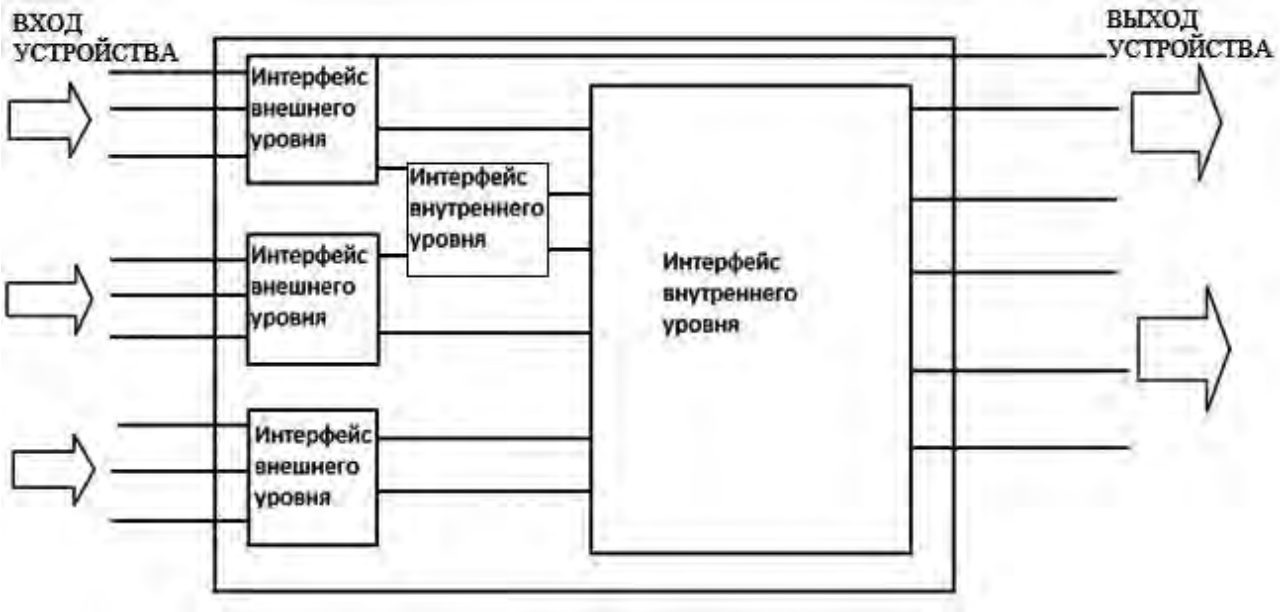

Рис. 2.4. Интерфейсное представление объекта контроля

Рассмотрим устройство с точки зрения структурных интерфейсов. Структура такого представления устройства приведена на рисунке (рис. 2.4). На нем отображена общая структура некоторого устройства. Разъемы «Вход»

и «Выход» показывают, куда подаются входные воздействия и считываются выходные реакции, соответственно.

Можно выделить огромное количество вариаций логических интерфейсов, при этом все они взаимодействуют, обмениваются информацией друг с другом, принимают внешние воздействия благодаря следующим типам функциональных сигналов:

- сигналы адреса (AddressWires),
- сигналы данных (DataWires),
- сигналы управления (ControlWires).

Для наглядности сигналы обмена с абстрактным логическим интерфейсом показаны на рисунке (рис. 2.5).

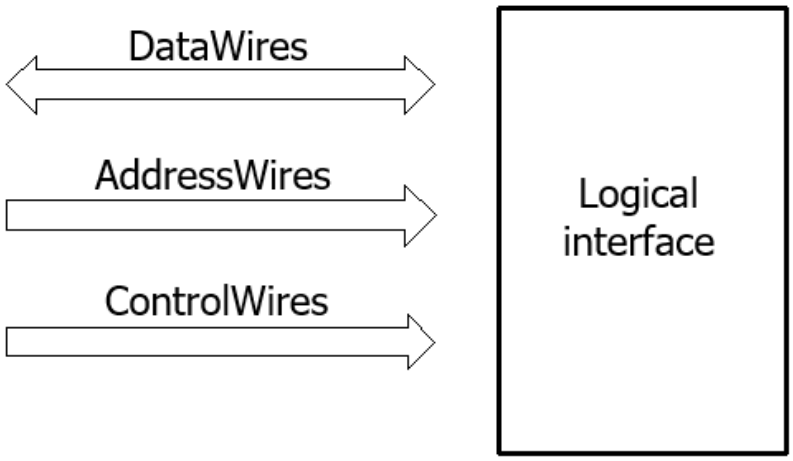

Рис. 2.5. Сигналы обмена с абстрактным логическим интерфейсом

Определенные функциональные сигналы могут быть не представлены в конкретной реализации логического интерфейса, если они не соответствуют требованиям к типу интерфейса. Например, логический интерфейс, описывающий память (MemoryInterface) имеет все указанные типы сигналов, а логический интерфейс называемый «Счетчик» (CountInterface) имеет только данные и управление, а интерфейс «Генератор тактовых сигналов» (ClockInterface) имеет только однобитовый сигнал данных [5].

Пользователь определяет количество сигналов каждого интерфейса, руководствуясь особенностями конкретной реализации устройства. Функциональное наименование сигналов интерфейса определяется типом корневого элемента интерфейса и реализуется при конструировании конкретного экземпляра логического интерфейса данного типа.

Рассмотрим соответствие функциональных сигналов абстрактного логического интерфейса на примере интерфейсов типа «Память» и «Счетчик». Сигналы адреса логического интерфейса «Память» отражают сигналы AddressWires абстрактного логического интерфейса, сигналы данных соответствуют сигналам DataWires, а сигналы Запись/Чтение и Выборка относятся к типу ControlWires. Сигналы типа DataWires могут быть как двунаправленными, так и только однонаправленными. Сигналы обмена с логическим интерфейсом «Память» показаны на рисунке (рис. 2.6).

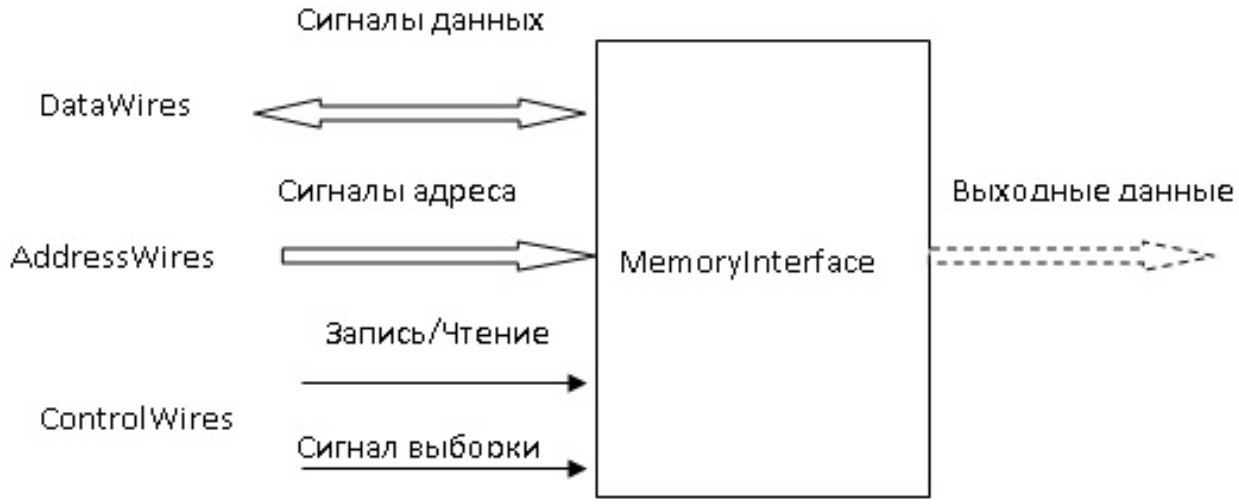

Рис. 2.6. Сигналы обмена с логическим интерфейсом «Память»

В логическом интерфейсе «Счетчик» адресные сигналы (AddressWires) абстрактного логического интерфейса отсутствуют, а сигналы Load, Reset, Increment и Decrement относятся к управляющим (ControlWires). Выходные данные, а также выходные сигналы переноса и заимствования могут присутствовать или отсутствовать в объекте этого интерфейса в зависимости от их

использования в схеме устройства. Пример обмена сигналами с логическим интерфейсом счетчика представлен на рисунке (рис. 2.7).

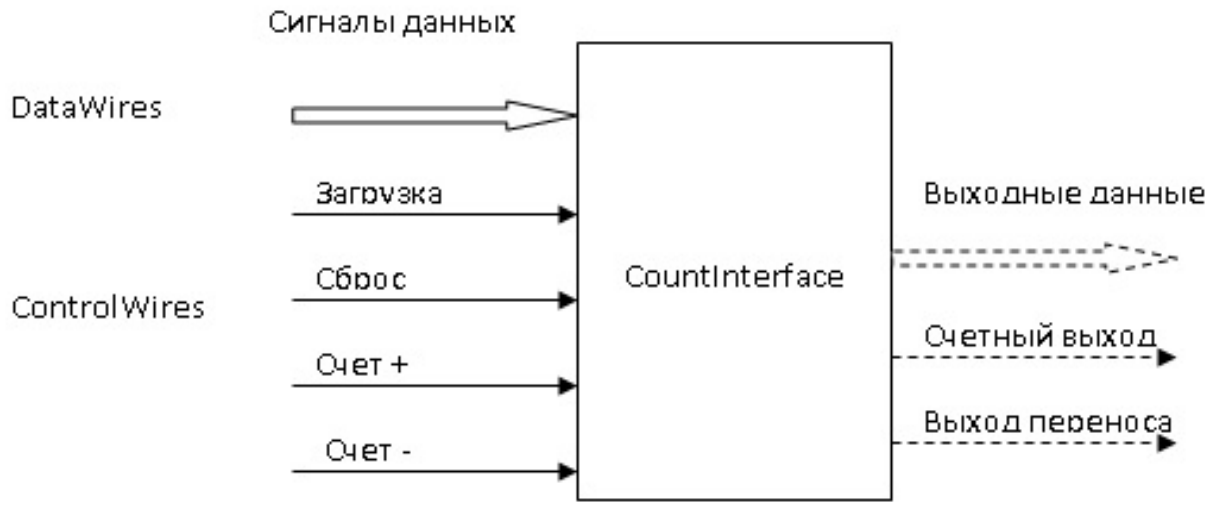

Рис. 2.7. Сигналы обмена с логическим интерфейсом «Счетчик»

Для того чтобы реализовать конкретный вариант интерфейса, соответствующий определенному классу логического интерфейса, необходимо провести анализ с целью определения какие типы функциональных сигналов будут использоваться, в каком количестве, и к каким сигнальным линиям устройства они будут относится.

Любой сигнал имеет два свойства в рамках модели логического интерфейса: «активность» и «задержка». Первая из характеристик, как следует из названия, описывает степень активности сигнала воспринимаемой интерфейсом. Этот параметр может определяться как числовым значением (уровнем) сигнала («логический 0» и «логическая 1»), так и изменением состояния сигнала, («фронтом» (переходом сигнала от значения 0 к 1) или «спадом» (соответственно переход от 1 к 0)). В случае если активность сигнала не определена (Undefined), то атрибут этого сигнала логического интерфейса не используется при реализации объекта данного интерфейса [5]. Однако это не значит, что сигнал не участвует в работе интерфейса. Например, когда мы перебираем некоторые данные во входном интерфейсе, нам все равно, что устройство считает активным - это вопрос интерпретации содержимого этих данных. Вто-

рой параметр – «задержка», позволяет учитывать временные соотношения при подаче сигналов в интерфейс.

Таким образом, функциональные блоки (интерфейсы) представляют собой совокупность цифровых микросхем, взаимодействующих между собой, такой блок можно для наглядности представить как многослойную нейронную сеть (рис. 2.8), где корневой элемент B_n это выходной слой, а непосредственно входы устройства I_1, I_2, I_3 это входной слой. Теперь рассмотрим задачу подбора проверяющей тестовой последовательности для интерфейса, как аналог метода обратного распространения ошибок при обучении нейронной сети.

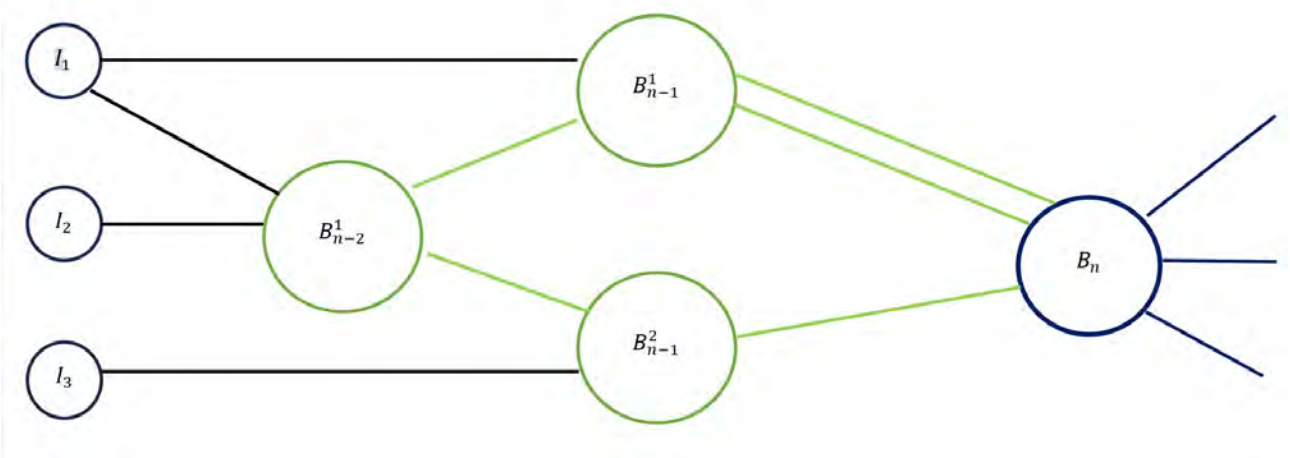

Рис. 2.8. Схематичное представление интерфейса, на подобии нейронной сетки

Таким образом, известна проверяющая последовательность для нашего корневого элемента B_n, то есть известна числовая комбинация из 0 и 1, которая должна быть на выходе элемента и соответственно, какая комбинация, при этом должна быть на входе элемента. Начинаем подбор входных тестовых воздействий от конца к началу от B_n к B^1_{n-1} и от B_n к B^2_{n-1}, то есть от выходного слоя к предыдущему. Исходя из этого, сначала мы подбираем входные воздействия, для компонентов B^1_{n-1} и B^2_{n-1} интерфейса так, чтобы в результате моделирования реакции этих компонентов на заданную последовательность на вход корневого элемента B_n пришла нужная проверяющая последовательность из 0 и 1. Повторяем данную операцию на последующие слои, пока

входная проверяющая последовательность не будет полностью сформирована. Данные манипуляции называются активация интерфейса.

Таким образом, в ходе программной реализации команд интерфейса, сохраняется совокупность сигналов, проверяющих элементы интерфейса, находящиеся раньше корневого элемента. На основе таких сохраненных комбинаций, обеспечивающих на выходе проверяемого элемента значение «0» (логического нуля) или «1» (логической единицы), создается необходимая тестовая последовательность для проверки последующих элементов интерфейса, в том числе и для элемента, лежащего в основе блока.

Более подробно рассмотрим принцип, описанный выше на примере интерфейса «Триггер» (рис. 2.9) в основе которого лежит D-триггер $1533TM2$ (Табл 2.1) [110, 111].

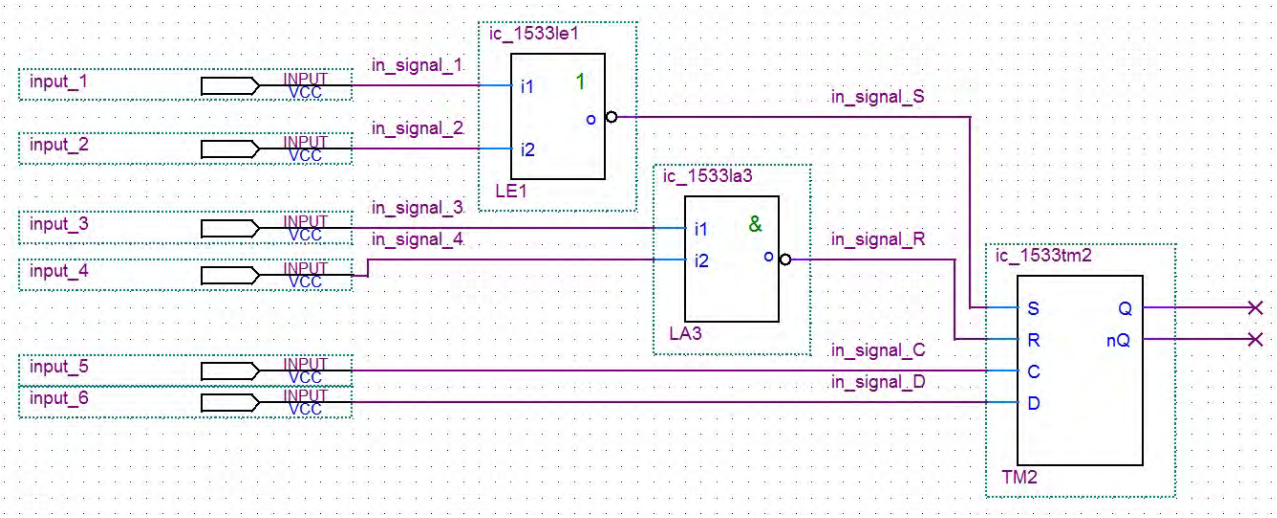

Рис. 2.9. Интерфейс в основе которого лежит D-триггер $1533TM2$

Триггеры являются ключевыми компонентами автоматов (устройств) с памятью. Синхронизация и тактирование играют важную роль в обеспечении корректной работы таких устройств. В результате изучения принципов работы триггеров, процессов синхронизации и тактирования можно получить общее представление о функционировании элементов с памятью, которые принципиально отличаются от комбинационных.

В контексте современных исследований в области теории автоматов с памятью, становится очевидным, что реакция таких устройств на входные

Табл. 2.1. Таблица истинности D-триггера $1533ТМ2$

Входы				**Выходы**	
S	R	C	D	Q	nQ
0	1	X	X	1	0
1	0	X	X	0	1
0	0	X	X	1	1
1	1	$0 \to 1$	1	1	0
1	1	$0 \to 1$	0	0	1
1	1	0	X	Q_0	nQ_0

воздействия существенно зависит от их текущего внутреннего состояния. Это означает, что одни и те же входные данные могут приводить к различным выходным значениям в зависимости от того, в каком состоянии находится автомат. Показатели текущего состояния и выходные данные автомата с памятью напрямую зависят от начальных условий и истории всех предыдущих входных комбинаций сигналов. Этот факт подчёркивает сложность и глубину процессов, происходящих внутри таких устройств.

Термин «последовательностные» был введён для обозначения класса устройств, которые осуществляют последовательное преобразование входных данных в выходные. Это преобразование включает в себя не только обработку текущих входных сигналов, но и учёт предыдущего состояния автомата, что делает процесс обработки информации в таких устройствах особенно интересным для изучения. Базовыми элементами класса последовательностных устройств являются триггеры. Они имеют возможность хранить один бит информации [110].

Очевидно, что низкий уровень сигнала (логический ноль) обозначают как «0», высокий уровень сигнала (логическая единица) обозначают как «1», «X» — произвольный уровень сигнала.

Низкий уровень напряжения на входах установки (S) или сброса (R) устанавливает выходы триггера в соответствующие состояния вне зависимости от состояния на других входах (C и D). При наличии на входах установки или сброса напряжения высокого уровня для правильной работы триггера требуется предварительная установка информации по входу данных относительно положительного фронта тактового сигнала ($0 \rightarrow 1$), а также соответствующая выдержка сигнала информации после подачи положительного фронта синхросигнала.

Таким образом, тестовая комбинация для данного элемента будет выглядеть следующим образом (Табл. 2.2).

Табл. 2.2. Значения входных (R, S, C, D) и выходных (Q, nQ) сигналов проверяющей тестовой комбинации

Название входов	**Потактовое значение сигнальной линии**										
R	0	1	1	1	1	1	0	1	1	1	0
S	1	0	1	1	1	0	1	1	1	1	1
C	0	0	0	1	0	0	0	0	1	0	1
D	0	0	0	0	0	1	1	1	1	1	1
Q	0	1	1	0	0	1	0	0	1	1	0
nQ	1	0	0	1	1	0	1	1	0	0	0

Рассмотрим интерфейс внешнего уровня в основе которого лежит D-триггер $1533TM2$ (рис. 2.9), для удобства будем его называть D-триггер интерфейс.

Входы триггера C и D напрямую соединены с входами объекта контроля $input\_5$ и $input\_6$, входы S и R соединены с входными разъемами опосредованно, через логические элементы $1533LE1$ (выполняет логическую операцию ИЛИ-НЕ) и $15533LA3$ (выполняет логическую операцию И-НЕ)

соответственно. Стоит отметить, что входы триггера S, R, C — управляющие, а вход D — информационный.

Для тестирования части цифрового объекта контроля, объеденной в триггер-интерфейс, программе необходимо подать на элемент $ic\_1533tm2$ тестовую последовательность, представленную в таблице (Табл. 2.2). На входы C, D подача необходимых сигналов происходит напрямую через входы $input\_5$ и $input\_6$ соответственно.

Чтобы понять, как на вход S и R триггера подать необходимую тестовую последовательность (Табл. 2.2), нужно обратиться к элементам, находящимся на предыдущем слое (рис. 2.8).

Элементом, преобразующем входной сигнал, поступающий на вход S D-триггера, является ic_1533LE, выполняющий операцию ИЛИ-НЕ, принцип работы этого элемента показан в таблице (Табл. 2.3).

Табл. 2.3. Таблица истинности элемента 1533LE1

Название входов	**Потактовое значение сигнальной линии**			
Вход "$i1$" элемента	1	0	1	0
Вход "$i2$" элемента	1	1	0	0
Выход o элемента	0	0	0	1

При проверки данного элемента, программа запоминает комбинации, обеспечивающие значение логической единицы (на обоих входах элемента низкий уровень напряжения 0) и логического нуля (на одном из входов высокий уровень напряжения 1) на выходе элемента, сигнал с которого напрямую идет на вход S элемента $ic\_1533tm2$.

Оперируя этими комбинациями, программа обеспечивает подачу таких сигнальных последовательностей на входы $input\_1$, $input\_2$, которые гарантируют на входе S триггера проверяющую тестовую последовательность из таблицы (Табл. 2.2). Один из возможных вариантов проверяющей тестовой

последовательности, сформированной на входах $input\_1$ и $input\_2$ представлен в таблице (Табл. 2.4).

Табл. 2.4. Пример тестовой комбинации, сформированной для входа S

Название входов	**Потактовое значение сигнальной линии**										
$input\_1$	0	1	0	0	0	0	0	0	0	0	0
$input\_2$	0	1	0	0	0	1	0	0	0	0	0
Вход S элемента $ic\_1533tm2$	1	0	1	1	1	0	1	1	1	1	1

Формирование сигнала тестового контроля для входа R происходит аналогичным образом, как описано выше, с учетом логики функционирования элемента $ic\_1533LA3$, находящегося в схемном решении раньше D-триггера (рис. 2.9). Один из возможных вариантов проверяющей тестовой последовательности, сформированной на входах $input\_3$ и $input\_4$ представлен в таблице (Табл. 2.5).

Табл. 2.5. Пример тестовой комбинации, сформированной для входа R

Название входов	**Потактовое значение сигнальной линии**										
$input\_3$	1	0	0	1	0	0	1	0	0	0	1
$input\_4$	1	0	1	0	0	0	1	0	0	0	1
Вход R элемента $ic\_1533tm2$	0	1	1	1	1	1	0	1	1	1	0

Таким образом, формируется одна из возможных тестовых последовательностей проверяющей D-триггер интерфейса (Табл. 2.6), комбинации, приведшие к переключению выходов триггера из состояния логического нуля в состояние логической единицы и наоборот, запоминаются системой и используются для тестирования последующих элементов.

Табл. 2.6. Проверяющая тестовая последовательность для рассматриваемого D-триггер интерфейс

Название входов	**Потактовое значение сигнальной линии**										
$input\_3$	1	0	0	1	0	0	1	0	0	0	1
$input\_4$	1	0	1	0	0	0	1	0	0	0	1
$input\_1$	0	1	0	0	0	0	0	0	0	0	0
$input\_2$	0	1	0	0	0	1	0	0	0	0	0
$input\_5$	0	0	0	1	0	0	0	0	1	0	1
$input\_6$	0	0	0	0	0	1	1	1	1	1	1
Q	0	1	1	0	0	1	0	0	1	1	0
nQ	1	0	0	1	1	0	1	1	0	0	0

Напомним, что о качестве тестового покрытия можно судить по критерию полноты тестового покрытия P (2.1) см. раздел 2.1.

Рассмотрим алгоритм, позволяющий провести тестирование радиоэлектронного устройства в автоматизированном режиме с использованием интерфейсного метода:

1. Разработка программной модели объекта контроля в специализированной САПР, поддерживающей Verilog. Для удобства в работе используется среда Altera Quartus II, как наиболее развитая, обладающая необходимым инструментарием. Данный этап включает в себя реализацию программных моделей компонентов устройства и проверки их на адекватность функционирования (см. раздел 1.3); замену аналоговых компонентов на программные модели, реализующие их функционал (см. раздел 1.5); реализацию взаимосвязи между компонентами в соответствии со схемным решением объекта контроля; проверку соблюдения всех требований к моделированию устройства.

2. На основе анализа схемного решения, функциональности компонентов и их иерархии осуществляется разделение объекта на структурные блоки (логические интерфейсы). Для каждого структурного блока определяется корневой элемент. Определяются вид интерфейса (внутренний или внешний), типы используемых функциональных сигналов , т.е. определяется какие входы на данный корневой элемент являются управляющими (ControlWires), а какие адресными (AddressWires) и информационными (DataWires).

3. Исходя из анализа типовых операций присущих корневому элементу интерфейса, типов используемых функциональных сигналов составляется числовая проверяющая тестовая последовательность (комбинация «0» и «1»), характерная для этого элемента.

   Для выбранного интерфейса в зависимости от управляющих элементов в интерфейсе и их количества, и особенностей функционирования, проверяющая числовая тестовая последовательность может отличаться.

4. Осуществляется активация интерфейса так, чтобы на корневой элемент пришла нужная проверяющая его комбинация из «0» и «1» (т.е. фактически, используя подход рассмотренный выше, выбранный интерфейс тестируется отдельно), устанавливается и запоминается, какая последовательность взаимосвязанных сигналов обеспечивает корневому элементу на каждом его выходе уровень сигнала, соответствующий «логическому нулю» или «логической единице» с целью использования ее в дальнейшем.

5. Рассматривая устройство, как набор интерфейсов взаимодействующих между собой, снова обращаемся к аналогии нейронной сети (рис. 2.8) и подходу разобранному выше, только теперь, оперируя не компонентами устройства, а интерфейсами целиком, аналогично находим тестовую

последовательность для всего объекта контроля. Проверяющие последовательности для каждого интерфейса были найдены на предыдущем этапе.

Важно отметить, что такие последовательности определены неоднозначно и допускают некоторые вариации. Перебираем такие возможные комбинации с целью максимизировать полноту тестового покрытия P (2.1), находим оптимальную, она и будет проверяющей тестовой последовательностью.

2.4 Реализация программного обеспечения комплексной разработки инструментальных тестов цифровых устройств

Программное обеспечение комплексной разработки инструментальных тестов цифровых устройств CRIT [1, 13] реализует указанные выше методы и предполагает использование широкого спектра специализированных инструментов для создания, хранения и модификации моделей компонентов радиоэлектронных систем, а также для разработки комплексных моделей устройств, формирования моделей входных воздействий, последующего моделирования реакций на эти воздействия и анализа результатов. В компьютерную технологию входят также средства формирования контрольно-диагностических тестов, пригодных для загрузки в аппаратуру тестового контроля, средства локализации неисправностей реальных устройств, подвергающихся тестированию с применением данной технологии.

Программный комплекс CRIT [1, 13] был реализован в виде приложения для операционной системы Windows. Приложение реализовано на языке C#. В нем интегрирована среда моделирования, база данных моделей цифровых

компонентов, а также встроены поддержка логических интерфейсов, средства анализа и визуализации результатов моделирования.

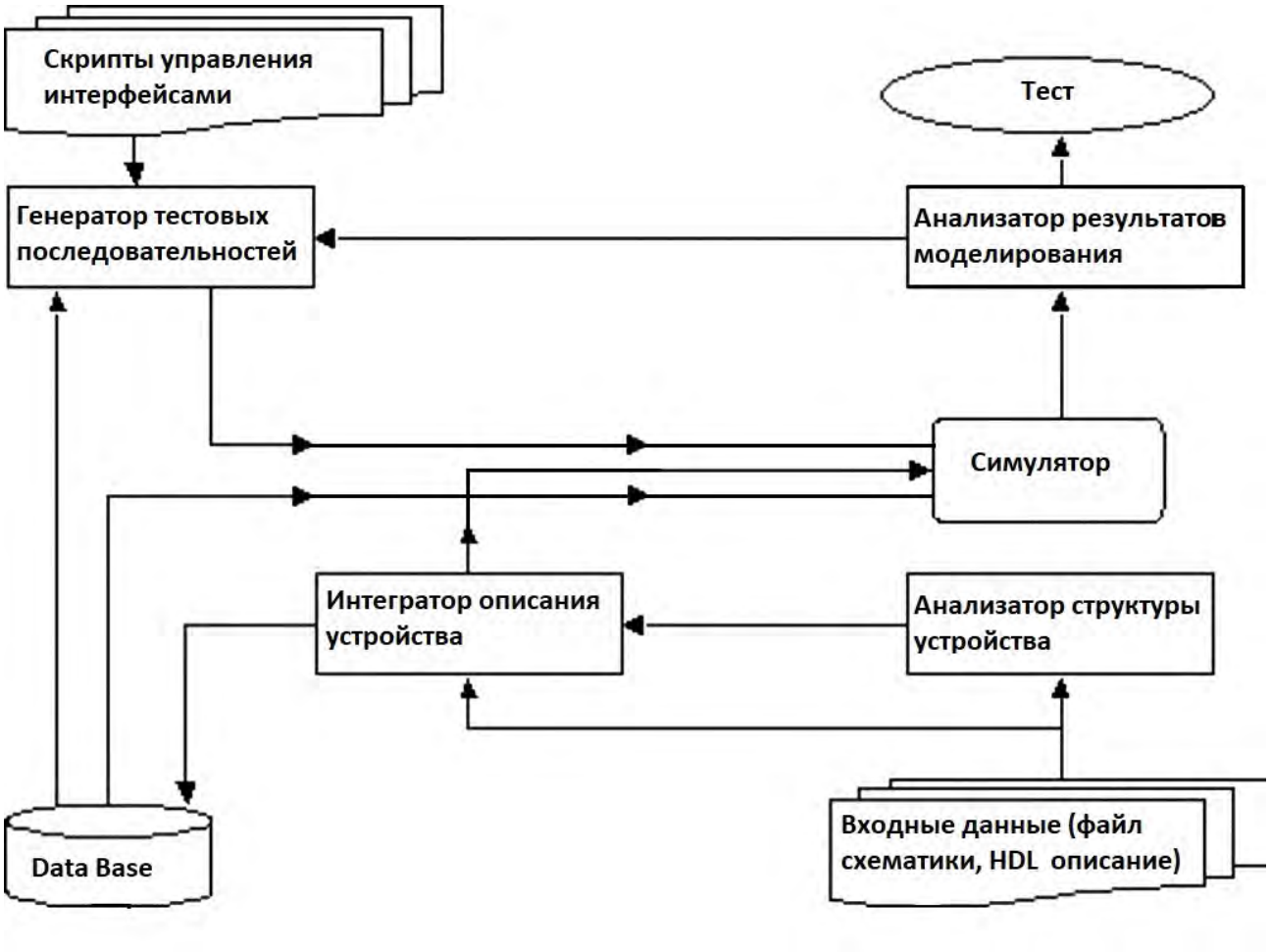

Рис. 2.10. Схемное представление системы CRIT

Создание модели тестовой последовательности автоматически выполняется системой на основе информации, полученной от пользователя, который работает с системными инструментами, поддерживающими логические интерфейсы. По завершению моделирования формируется стандартный файл результатов моделирования, в который записываются временные последовательности состояний всех сигнальных линий устройства. Структура системы приведена на рисунке (рис. 2.10).

Благодаря базе данных, можно синхронизировать работу всех программных средств системы. Данные для моделирования извлекаются и помещаются в нее при помощи генератора запросов, который по мере завершения обра-

ботки каждого этапа формирует запрос, соответствующий тому или иному актуальному действию на текущий период времени.

Система поддерживает 2 типа систем управления базами данных:

- стандартный сервер MS SQL Server (этот сервер должен быть заранее установлен, и он может находится как на локальной машине, так и на машине подключенной к сети, для использования этого сервера требуется лицензия);

- Компактный сервер MS SQL Server Compact (этот сервер должен быть заранее установлен на локальную машину, для него необходима свободная бесплатная лицензия, база данных представляет собой файл с расширением «sdf», находящийся на локальной машине).

Анализатор структуры служит для выделения из входного файла схематики устройства иерархической структуры, описывающей состав, сигналы и связи между ними. Интегратор описания устройства, на основе полученной структуры строит Top–model устройства на языке Verilog.

Функцию моделирования работы электронных устройств выполняет симулятор. Его предназначение заключается в том, чтобы используя Top-model устройства и моделей компонентов, а также модели входных воздействий получить представление о динамике изменения состояния всех сигналов (как внутренних, так и внешних).

В программном комплексе используется симулятор Icarus Verilog, он ориентирован на язык Verilog. Пакет Icarus Verilog должен быть предварительно установлен на локальную машину. Для его использования требуется свободная бесплатная GNU лицензия. Система, используя компилятор Icarus и все модели, относящиеся к устройству, производит автоматическое моделирование. В начале процесса моделирования все модели, которые необходимы для успешной симуляции извлекаются из базы данных.

При использовании пакета Icarus моделирование проходит в 2 этапа. На первом этапе компилируются файлы всех моделей. На втором, в случае успешной компиляции, запускается сам симулятор и происходит моделирование. По окончании моделирования системой формируется файл результатов с временными последовательностями состояний всех сигнальных линий устройства, который и является тест-программой.

Логические интерфейсы реализуются в рамках объектно-ориентированной модели с использованием процедуры наследования классов. Базовым классом, содержащим поля и методы, присущие всем логическим интерфейсам, является класс iLogicalIntreface. Из этого класса порождаются различные типы логических интерфейсов, например iRam, iCounter, iClock, iReset и т.п., каждый из которых описывает соответствующий тип логического интерфейса. Наследником каждого типа логического интерфейса является класс, описывающий функциональность данного типа. Для типов логических интерфейсов упомянутых ранее это классы Ram, Counter, Clock, Reset.

Генератор тестовых последовательностей создает непосредственно первое и последующие приближения модели тестовой последовательности, опираясь на данные интерфейсы и скрипт- программы, которыми осуществляется управление ими, на основе информации о проверяющей тестовой последовательности для корневого элемента интерфейса. Данная модель корректируется на последующих этапах итерации. Когда происходит моделирование реакции объекта контроля на данные воздействия, анализатор результатов моделирования осуществляет обработку информации, собранную в течении одного цикла моделирования и ее последующий анализ, одновременно находится значение тестового покрытия. Команды, созданные анализатором, генератор тестовых последовательностей использует для коррекции модели входной последовательности.

На «Программное обеспечение комплексной разработки инструментальных тестов цифровых устройств» CRIT получено свидетельство о государ-

ственной регистрации программы для ЭВМ 2017612352 [13], с копией данного документа можно ознакомиться в Приложении А.

2.5 Выводы по главе 2

Во второй главе представлена непосредственно технология тестового контроля цифровых радиоэлектронных устройств и подробно разобраны все ее этапы, описан критерий полноты тестового покрытия. Данные подходы основаны на использовании методов математического и компьютерного моделирования как цифровых радиоэлектронных систем, так и представления входных воздействий. Стоит отметить, что при моделировании определяется реакция модели системы на заданные входные воздействия, т.е. моделируется поведение реальной системы, работающей с реальными сигналами. Если модели системы и модели входных воздействий адекватны, то результаты эмуляции будут адекватно отражать поведение реальной системы. При этом модельная совокупность входных воздействий и реакций на них представляет поведение реальной исправной системы и может быть использована в качестве эталона для реализации соответствующего теста.

Разработаны подходы к формированию моделей цифровых радиоэлектронных устройств с элементами программируемой логики (такие как ПЗУ или ПЛИС), при отсутствии доступа к конфигурирующей программе («прошивке»), в контексте задачи тестового контроля. Описаны плюсы и минусы каждого из подходов.

Разработан интерфейсный метод автоматизированного построения тестовых программ, как аналог метода обратного распространения ошибки при обучении нейронной сети, детально разобранный на примере. Разработан алгоритм, позволяющий провести тестирование радиоэлектронного устройства в автоматизированном режиме с использованием интерфейсного метода.

В главе 2 описана структура разработанного «Программного обеспечения комплексной разработки инструментальных тестов цифровых устройств» CRIT, реализующего указанные выше методы и использующего широкий спектр специализированных инструментов для создания, хранения и модифи-

кации моделей компонентов радиоэлектронных систем, а также для разработки комплексных моделей устройств, формирования моделей входных воздействий, последующего моделирования и анализа результатов. В него интегрирована компьютерная технология, реализующая интерфейсный метод тестирования и автоматизирующая процесс построения тест—программ. Система формирует данные тесты в виде пригодном для загрузки в аппаратуру тестового контроля.

На программный комплекс CRIT получено свидетельство о государственной регистрации программы для ЭВМ № 2017612352 [13].

Глава 3

Практическое применения реализованных методов и алгоритма на реальном цифровом объекте контроля

3.1 Моделирование цифрового объекта тестирования

Для демонстрации эффективности предложенных методов, подходов и алгоритма тестирования, которые были реализованы в программном комплексе CRIT, был осуществлен тестовый контроль цифрового модуля, содержащего в себе микросхемы средней степени интеграции. Следует отметить, что широкая номенклатура ЦРЭУ различной степени интеграции в настоящее время выпускается промышленностью и нуждаются как в тестовом контроле непосредственно во время производства, так и в ремонте в ходе эксплуатации.

Информация об устройстве, была предложена в виде его принципиальной схемы в бумажном виде (6 листов формата А3), на рисунке (рис. 3.1) представлен фрагмент схемного решения устройства.

При моделировании цифрового объекта контроля, как было уже описано выше, используется стратегия восходящего проектирования (раздел 2.1), поэтому сначала производиться моделирование микросхем, являющихся компонентами объекта контроля.

1533ЛЕ1 — цифровая интегральная схема транзисторной логики с диодами Шоттки серии ТТЛ. Микросхемы *1533ЛЕ1* представляют собой четыре одинаковых логических элемента 2ИЛИ-НЕ, т.е. реализуют булеву функцию $F(x_1, x_2) = \overline{x_1 + x_2}$. Данные микросхемы предназначены для работы в узлах и блоках радиоэлектронной аппаратуры специального назначения [110,111,124].

1533АП5 — микросхема представляет собой два четырёхразрядных магистральных передатчика, которые не инвертируют входную информацию.

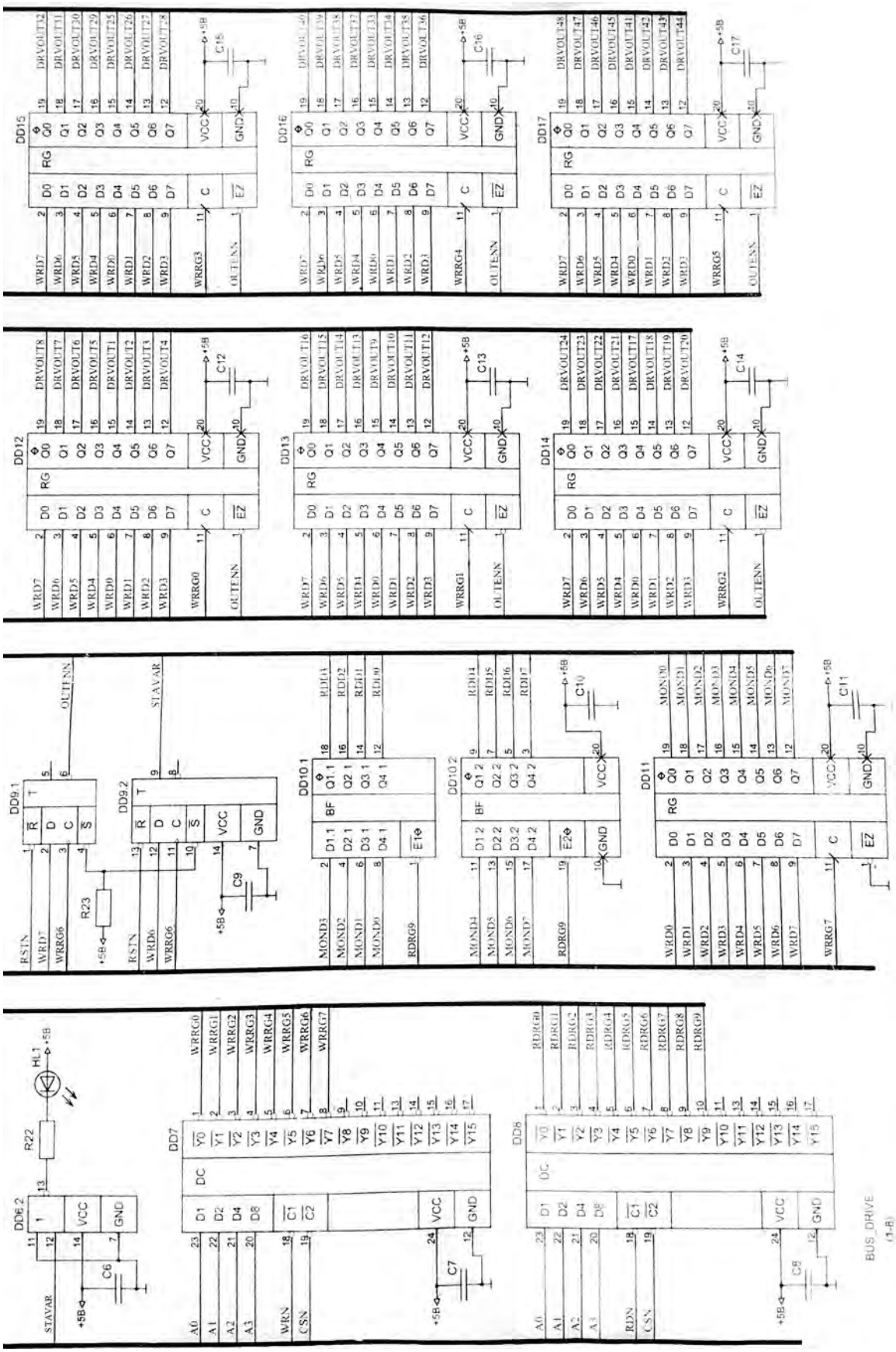

Рис. 3.1. Фрагмент принципиальной схемы устройства

Она имеет три состояния на выходе: логического нуля, логической единицы и Z-состояние (состояние высокого импеданса). Чтобы перевести выходы микросхемы в высокоимпедансное состояние необходимо подать на входы управления напряжение высокого уровня [110, 111, 124].

Высокоимпедансное логическое состояние — состояние выхода логического устройства, при котором он обладает высоким сопротивлением (импедансом), то есть фактически отключен от подсоединённого к нему проводника. Также высокоимпедансным является состояние проводника, при котором все логические выходы, к нему подсоединённые находятся в высокоимпедансном состоянии.

Условно-графическое обозначение этого элемента представлено на рисунке (рис. 3.2).

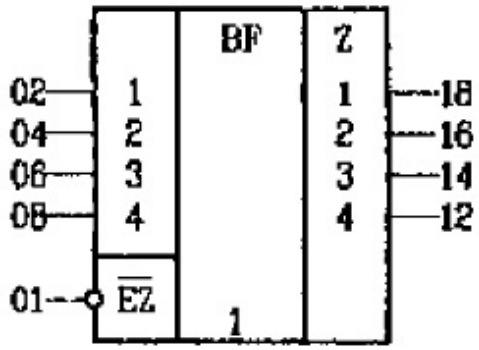

Рис. 3.2. Условно-графическое обозначение *1533АП5*

- вывод 01 ($\overline{EZ}$) — вход разрешения снятия состояния высокого импеданса, т.е. сигнал управления (Control Wires);
- выводы 02, 04, 06, 08 — информационные входы $D0, D1, D2, D2$, т.е. сигналы данных (DataWires);
- выводы 18, 16, 14, 12 — выходы $Q0, Q1, Q2, Q3$ элемента.

Математическая модель работы данного устройства:

$$\begin{aligned} Q(Q0, Q1, Q2, Q3) &= 4'bz, && \text{при} \quad \overline{EZ} = 1, \\ Q(Q0, Q1, Q2, Q3) &= D(D0, D1, D2, D3), && \text{при} \quad \overline{EZ} = 0. \end{aligned} \tag{3.1}$$

Программная модель представлена в приложении Е.

1533ИД3—дешифратор четырехразрядного двоичного кода, задаваемого на входах D_1, D_2, D_3, D_4. Особенностью данного дешифратора является наличие стробирующих входов $\overline{C_1}$, $\overline{C_2}$. Если на любой из этих входов подать напряжение высокого уровня, то на всех 16 выходах устройства также будет высокий уровень независимо от значения, поданного на входы D_1, D_2, D_3, D_4.

Таким образом, при подаче напряжения низкого уровня на стробирующие входы $\overline{C_1}$, $\overline{C_2}$ дешифратор будет работать в штатном режиме, преобразовывая четырёхразрядный двоичный код на входах D_1, D_2, D_3, D_4 в сигнал логического нуль на соответствующем выходе [110, 111, 124].

Визуальное представление программной модели на рисунке (рис. 3.3).

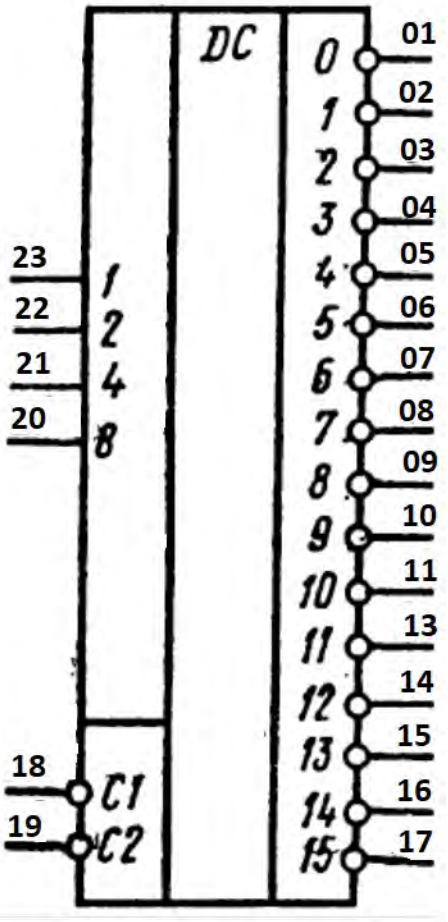

Рис. 3.3. Условно-графическое представление элемента *1533ИД3*

Назначение выводов микросхемы:

- выводы 18, 19 — входы стробирования $\overline{C1}, \overline{C2}$, т.е. сигналы управления (ControlWires);

- выводы 23, 22, 21, 20 — входы информационные $D0, D1, D2, D3$ устройства, т.е. сигналы данных (DataWires);

- выводы 01,02,03,04,05,06,07,08,09,10,11,13,14,15,16,17 — выходы; $\overline{Y0}, \overline{Y1}, \overline{Y2}, \overline{Y3}, \overline{Y4}, \overline{Y5}, \overline{Y6}, \overline{Y7}, \overline{Y8}, \overline{Y9}, \overline{Y10}, \overline{Y11}, \overline{Y12}, \overline{Y13}, \overline{Y14}, \overline{Y15}$ элемента.

Математическую модель работы устройства, наиболее лаконично и понятно выразит, таблица состояний элемента (Табл. 3.1).

Программная модель представлена в приложении Е.

1533СП1 — микросхема предназначена для сравнения четырехразрядных двоичных чисел, представленных в прямом коде. Сравнение производится со старших разрядов, если они различны, то эти разряды определяют результат сравнения, если они равны - проводится сравнение последующих разрядов и т.д. [110, 111, 124].

Условно-графическое представление микросхемы на рисунке (рис. 3.4).

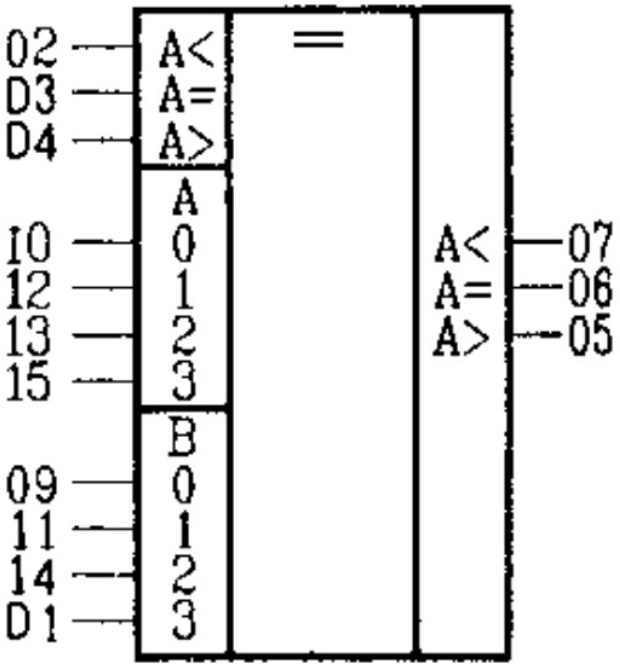

Рис. 3.4. Условно-графическое представление *1533СП1*

Табл. 3.1. Таблица состояний элемента *1533ИД3*

Входы						**Выходы наращивания**															
$\overline{C1}$	$\overline{C2}$	D3	D2	D1	D0	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$	$\overline{Y4}$	$\overline{Y5}$	$\overline{Y6}$	$\overline{Y7}$	$\overline{Y8}$	$\overline{Y9}$	$\overline{Y10}$	$\overline{Y11}$	$\overline{Y12}$	$\overline{Y13}$	$\overline{Y14}$	$\overline{Y15}$
0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1
0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	1	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	x	x	x	x	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Назначение выводов микросхемы:

- выводы 01, 03, 04 — входы сравнения ($A<, A=, A>$) ;
- выводы 10, 12, 13, 15 — входы первого из четырехразрядных сравниваемых чисел A, в дальнейшем будем обозначать эти входы как $A0, A1, A2, A3$ при этом $A3$ – старший разряд числа, а $A0$ – младший разряд;
- выводы 09, 11, 14, 01 — входы второго из четырехразрядных сравниваемых чисел B, в дальнейшем будем обозначать эти входы как $B0, B1, B2, B3$ при этом $B3$ – старший разряд числа, а $B0$ – младший разряд;
- входы 07, 06, 05 — выходы ($A<, A=, A>$) элемента.

Математическую модель данного элемента наиболее лаконично и понятно можно выразить с помощь таблицы состояний (Табл. 3.2).

1533ИР37— восьмиразрядный буферный регистр с тремя состояниями на выходе (логического нуля, логической единицы и z -состояние(состояние высокого импеданса)) [110, 111, 124].

Визуальное представление программной модели на рисунке (рис.3.5).

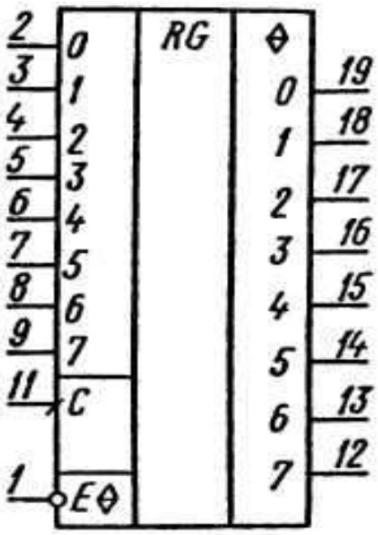

Рис. 3.5. Визуальное представление *1533ИР37*

Табл. 3.2. Таблица состояний элемента *1533СП1*

Входы сравнения				**Входы наращивания**			**Выходы**		
A3;B3	A2;B2	A1;B1	A0;B0	A>B	A<B	A=B	A>B	A<B	A=B
A3>B3	x	x	x	x	x	x	1	0	0
A3<B3	x	x	x	x	x	x	0	1	0
A3=B3	A2>B2	x	x	x	x	x	1	0	0
A3=B3	A2<B2	x	x	x	x	x	0	1	0
A3=B3	A2=B2	A1>B1	x	x	x	x	1	0	0
A3=B3	A2=B2	A1<B1	x	x	x	x	0	1	0
A3=B3	A2=B2	A1=B1	A0>B0	x	x	x	1	0	0
A3=B3	A2=B2	A1=B1	A0<B0	x	x	x	0	1	0
A3=B3	A2=B2	A1=B1	A0=B0	1	0	0	1	0	0
A3=B3	A2=B2	A1=B1	A0=B0	0	1	0	0	1	0
A3=B3	A2=B2	A1=B1	A0=B0	x	x	1	0	0	1
A3=B3	A2=B2	A1=B1	A0=B0	1	1	0	0	0	0
A3=B3	A2=B2	A1=B1	A0=B0	0	0	0	1	1	0

Назначение выводов микросхемы:

- выводы 01 $\overline{EZ}$— вход разрешения снятия состояния высокого импеданса с выхода (ControlWires);
- выводы 02, 03, 04, 05, 06, 07, 08, 09 — информационные входы $D0, D1, D2, D3, D4, D5, D6$ соответственно (DataWires);
- вывод 11— тактовый вход C (ControlWires);
- выводы 12, 13, 14, 15, 16, 17, 18, 19 — выходы $Q7, Q6, Q5, Q4, Q3, Q2, Q1, Q0$ микросхемы соответственно.

Математическая модель работы микросхемы *1533ИР37*:

$$Q(Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7) = 8'bz, \quad \text{при} \quad \overline{EZ} = 1,$$
$$Q(Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7) = D(D0, D1, D2, D3,$$
$$D4, D5, D6, D7),$$
$$\text{при} \quad \overline{EZ} = 0, C = 0 \to 1.$$

Здесь $8'bz$ — восьмиразрядное состояние высокого импеданса z, $C = 0 \to 1$ — фронт сигнала C, т.е. момент перехода сигнала из состояния логического нуля в состояние логической единицы.

Программные модели описанных микросхем представлены в Приложении Е.

После получения и отладки программных моделей всех компонентов, составляющих объект контроля, создается модель цифрового устройства (раздел 2.1). Соединяются все его компоненты в соответствии со структурой, полученной из принципиальной схемы, организовывая тем самым взаимосвязь между этими модулями. Используя программные средства среды Altera Quartus II, создается программная модель устройства (Netlist), из-за большого количество элементов и связей между ними модель получилась довольно объемной, поэтому на рисунке (рис. 3.6) представлен только фрагмент визуального представления программной модели. Стоит пояснить, что сигнальная линия со звездочкой на конце или в начале, это не «порванная» и не «разъединенная» линия, просто средства и возможности QuartusII, позволяют соединять линии, прерванные звездочкой, но имеющие одинаковое название, в одну сигнальную линию во время компиляции.

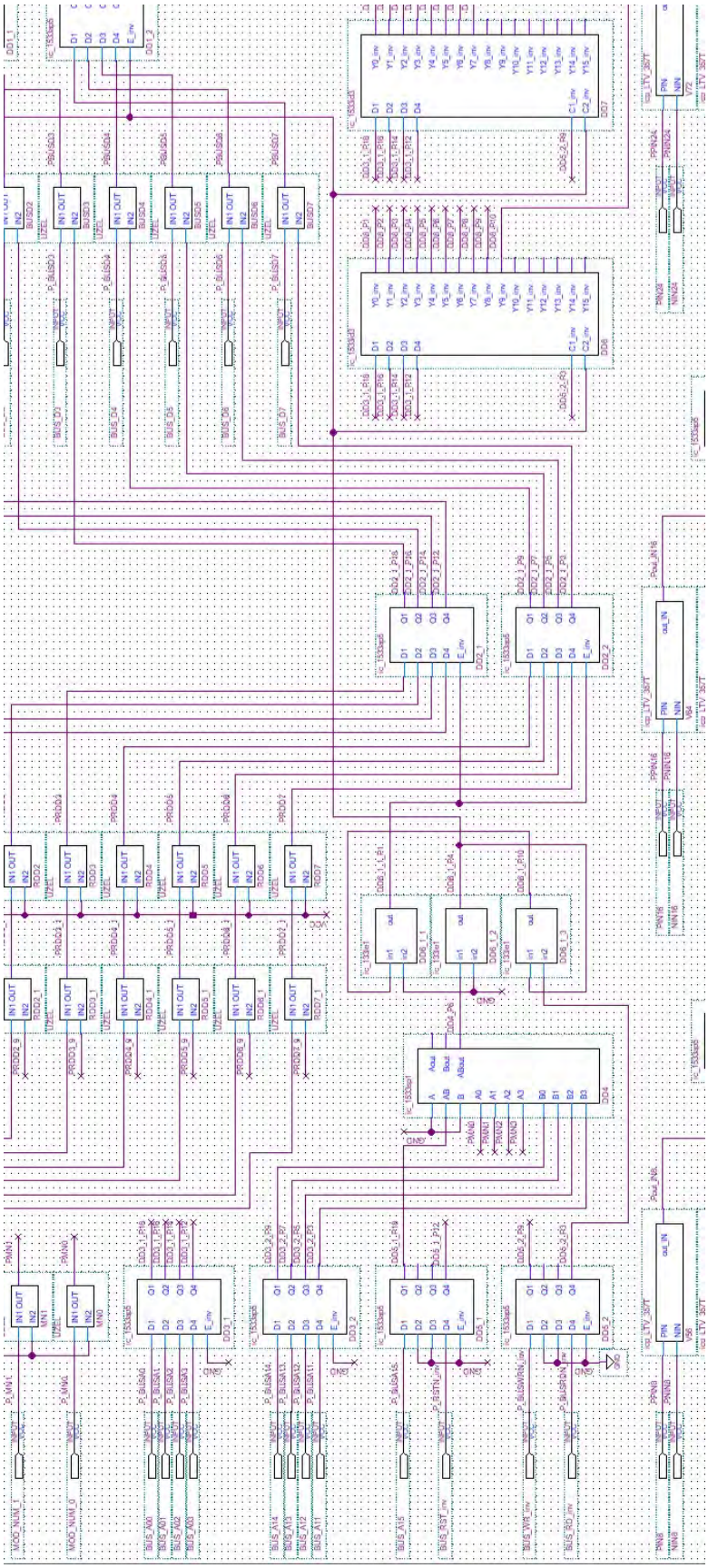

Рис. 3.6. Визуальное представление фрагмента программной модели объекта контроля

3.2 Тестирование объекта контроля

На этапе создания тест-программы используется интерфейсный метод и его программная реализация в рамках САПР CRIT. Интерфейсы были выбраны так, что их корневыми элементами были:*1533СП1* (микросхема предназначена для сравнения 4-разрядных двоичных чисел), *1533ИР37* (буферный регистр), *1533ИД3* (дешифратор четырех разрядного двоичного кода). Проверяющая тестовая последовательность для корневых элементов в виде скрипта (текстового файла формата .tst) загружается в САПР CRIT, куда ранее уже были загружены модели компонентов объекта контроля и его Netlist. Результатом моделирования процесса тестирования является файл результатов, включающий в себя временные последовательности состояний всех сигнальных линий устройства. Этот файл является тестовой программой. Визуализация фрагмента результатов тестирования приведена на рисунке (рис.3.7).

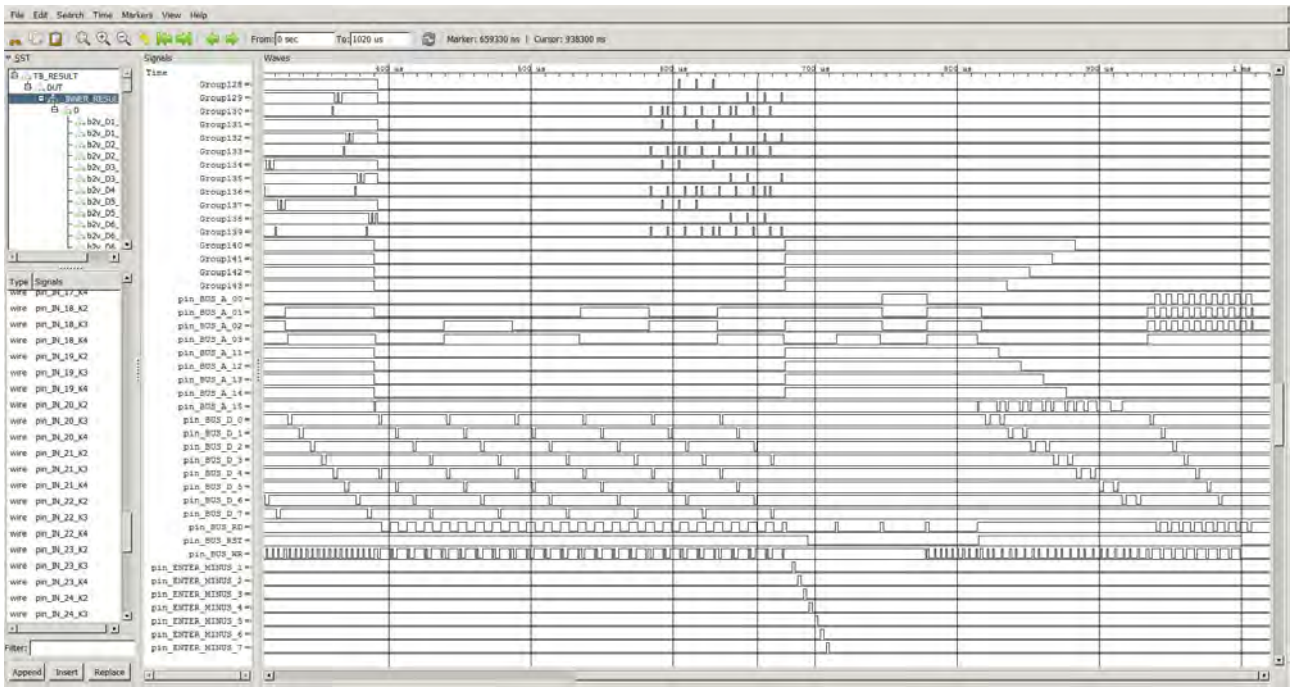

Рис. 3.7. Визуализация фрагмента результатов тестирования рассматриваемого ЦРЭУ

Как видно из анализа результатов тестового покрытия (рис. 3.8) общее покрытие составляет $97,93\%$. Оно удовлетворяет заявленным требованиям и является максимальным, т.к. большей величины покрытия достичь не возмож-

но из-за особенностей схемного решения устройства (на некоторых сигнальных линиях невозможно изменить значение сигнала).

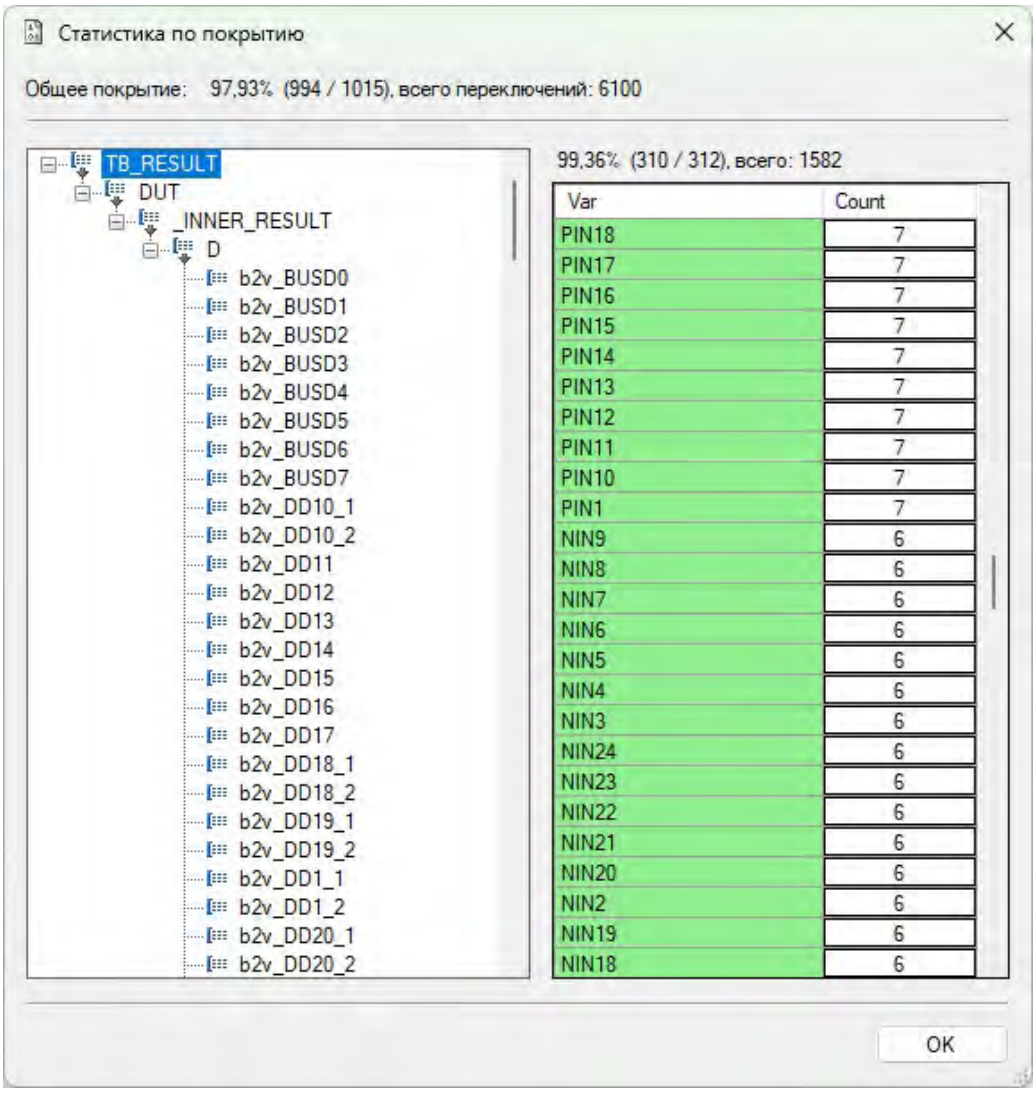

Рис. 3.8. Представление анализа результатов тестирования рассматриваемого объекта контроля

После того как получен тест с удовлетворительным покрытием, осуществляется отладка теста на установке тестового контроля УТК-512 с использованием непосредственно самого теста (тест в пригодном формате загружается в программную среду ЯСТЕК [8, 123], которая осуществляет взаимодействие с УТК-512) и эталонного объекта контроля, подключенного к данной установке.

В ситуации, когда происходит остановка процесса тестирования из-за несоответствия реакции эталонного объекта контроля на тестовое воздействие

(т.е. останов по браку), первым шагом является проверка корректности программной модели объекта контроля. Если модель работает корректно, производится корректировка теста с целью устранения ошибки.

В случае корректного прохождения теста на УТК-512, тестовая последовательность признается корректной для дальнейшего использования.

Прогон написанного теста на УТК-512 (рис. 3.9) показал, его исправность, таким образом, можно считать поставленную задачу выполненной.

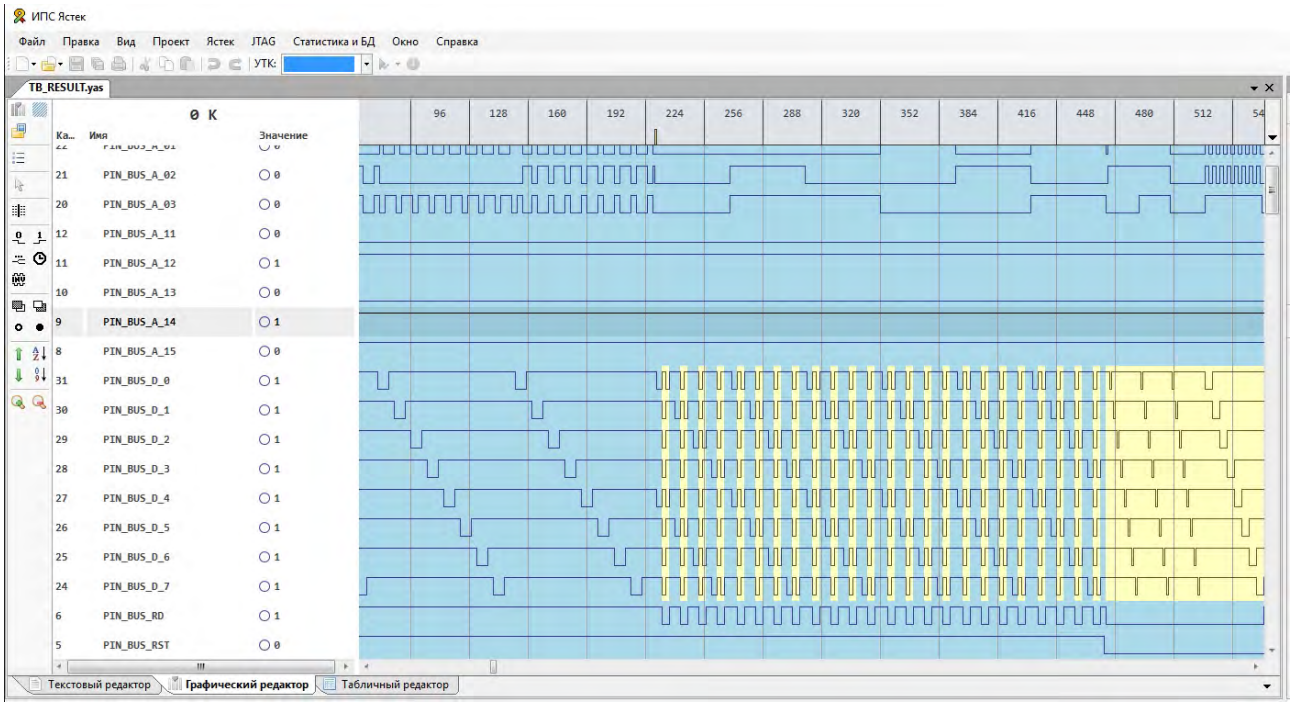

Рис. 3.9. Визуальное представление теста рассматриваемого объекта контроля в среде ЯСТЕК

3.3 Выводы по главе 3

Для демонстрации эффективности разработанных методов, алгоритма и программного обеспечения был реализован полный цикл тестового контроля достаточно сложного цифрового устройства с использованием всех предложенных технологий, методов и алгоритма. В результате была достигнута требуемая полнота теста. Этот результат был подтвержден при тестировании экземпляра указанного устройства с использованием установки тестового контроля УТК-512.

Были разработаны математические модели компонентов устройства, на основе разработанных методов формирования программных моделей цифровых микросхем, в контексте задач тестирования, были программно смоделированы непосредственно эти компоненты. Используя полученные программные модели, данный о их взаимосвязях и специализированные программные средства была сформирована программная модель всего объекта контроля.

Используя интерфейсный метод и алгоритм автоматизированного построения тест-программ был создан проверяющий тест контроля, который показала максимально возможный результаты полноты тестового покрытия.

При тестировании эталонного объекта контроля, на установке УТК-512 тест показал в полной мере свою исправность и адекватность. Тестирование эталонного объекта контроля прошло успешно.

Заключение

В диссертационном исследовании решена актуальная задача по разработке методов, алгоритмического и программного обеспечения автоматизированного тестового контроля сложных ЦРЭУ. Получены следующие результаты:

1. Проведен анализ существующих методов и средств тестирования цифровых устройств. Сделан вывод о необходимости автоматизации процесса формирования моделей тестовых воздействий, которые бы обеспечивали адекватный контроль и диагностику сложных ЦРЭУ.

2. Созданы методы формирования моделей цифровых устройств с учетом особенностей задачи тестового контроля, для каждого метода разобраны преимущества и недостатки.

3. Разработан метод внедрения аналоговых узлов в имитационную модель ОК, на основе математических моделей. Детально разобраны наиболее часто встречающиеся случаи, такие как компараторы, транзисторные переключатели и т.п.

4. Разработаны подходы к формированию моделей ЦРЭУ с элементами программируемой логики, при отсутствии доступа к конфигурирующей программе («прошивке»), описаны плюсы и минусы каждого из них.

5. Разработан интерфейсный метод автоматизированного построения тестовых программ, как аналог метода обратного распространения ошибки при обучении нейронной сети, детально разобранный на примере. Разработан алгоритм, позволяющий провести тестирование радиоэлектронного устройства в автоматизированном режиме с использованием интерфейсного метода.

6. Разработано «Программное обеспечение комплексной разработки инструментальных тестов цифровых устройств CRIT». В него интегрирована компьютерная технология, реализующая интерфейсный метод тестирования и автоматизирующая процесс построения тест- программ.

7. Разработанная компьютерная технология была использована для проведения полного цикла создания теста достаточно сложного цифрового устройства. При этом осуществлено тестирование цифрового устройства с использованием всех предложенных технологий, методов и алгоритма. В результате была достигнута требуемая полнота теста. Результат был подтвержден при тестировании экземпляра указанного устройства с использованием установки тестового контроля (УТК-512).

Список сокращений

Список сокращений на английском языке

AOI — Automated Optical Inspection

ATE — Automated Test Equipment

BGA — Ball Grid Array

CALS — Continuous Acquisition and Life cycle Support

ERP — Enterprise Resource Planning

FCT — Functional Test

HDL — Hardware Description Language

ICT — In-Circuit Test

IEEE — Institute of Electrical and Electronics Engineers

ISO — International Organization for Standardization

JTAG — Joint Test Action Group

LLD — Low-Level Design

RAM — Random Access Memory

VCC — Voltage Common Collector

VHDL — Very high speed integrated circuits Hardware Description Language

Список сокращений на русском языке

АСТПП — Автоматизированная система технологической подготовки производства

АСУП — Автоматизированная система управления предприятием

АСУТП — Автоматизированная система управления технологическим процессом

АЦП — Аналого-Цифровой Преобразователь

ЦРЭУ — Цифровое Радиоэлектронное Устройство

ОЗУ — Оперативное Запоминающее Устройство

ОК — Объект Контроля

ПЗУ — Постоянное Запоминающее Устройство

ПЛИС — Программируемая Логическая Интегральная Схема

САПР — Система Автоматизированного Проектирования

УТК — Установка Тестового Контроля

ЦАП — Цифро-Аналоговый Преобразователь

ЭВМ — Электронная вычислительная машина

Список литературы

1. Елаев, Е. В. Система комплексной разработки тестов цифровых устройств (crit) / Е. В. Елаев // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2023. – № 4. – С. 122-127.
2. Елаев, Е. В. О методике создания поведенческих моделей цифровых объектов тестового контроля / Е. В. Елаев // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2016. – № 3. – С. 17-23.
3. Елаев, Е. В. Интерфейсный метод автоматизированной генерации тестовых воздействий для цифровых радиоэлектронных объектов контроля /Е. В. Елаев // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2015. – № 4. – С. 19-24.
4. Мащинский, Н. С. Генерация тестовых воздействий для диагностики цифровых электронных систем / Н. С. Мащинский, Е. В. Елаев // Вестник Санкт-Петербургского государственного университета технологии и дизайна. Серия 1: Естественные и технические науки. – 2017. – № 4. – С. 41-45.
5. Grishkin, V. M. Automated test development system for digital devices /V. M. Grishkin, D. A. Ovsyannikov, N. S. Maschinskiy, Y. V. Yelaev // Moscow Workshop on Electronic and Networking Technologies, MWENT 2018 - Proceedings : 1, Moscow, 14–16 марта 2018 года. Vol. 2018-March. – Moscow, 2018. – P. 1-4.
6. Grishkin, V. Interface Method of Digital Devices Testing / V. Grishkin, Y. Yelaev, G. Lopatkin [et al.] // Tenth International Vacuum Electron Sources Conference (IVESC) & Second International Conference on Emission Electronics (ICEE), Санкт-Петербург, 30 июня – 04 2014 года. – Санкт-Петербург: Institute of Electrical and Electronics Engineers, 2014. – P. 107-108.

7. Melnik, V. I. Methods of modeling of the test inputs for analysis the digital devices / V. I. Melnik, A. N. Mikhailov, Y. V. Yelaev [et al.] // 2014 International conference on computer technologies in physical and engineering applications (ICCTPEA): Editor: E.I. Veremey, Санкт-Петербург, 30 июня – 04 2014 года / Санкт-Петербургский государственный университет; IEEE (IEEE Catalog number CFP14BDA-USB). – Санкт-Петербург: Издательство Санкт-Петербургского государственного университета, 2014. – P. 112-113.
8. Степанов, Ю. Л. Развитие программной среды "ЯСТЕК" и ее использование при написании тестовых программ для цифровых модулей / Ю. Л. Степанов, В. М. Гришкин, Е. В. Елаев, П. А. Федюкович // Вопросы радиоэлектроники. – 2015. – № 2. – С. 198-205.
9. Мащинский, Н. С. Моделирование сложных цифровых устройств с целью их тестирования / Н. С. Мащинский, Е. В. Елаев, П. А. Федюкович // Процессы управления и устойчивость. – 2015. – Т. 2, № 1. – С. 452-457.
10. Гусев, О. А. Автоматизация генерации тестовых воздействий для комбинационных цифровых схем / О. А. Гусев, Е. В. Елаев, Н. С. Мащинский, А. Нуракунов // Процессы управления и устойчивость. – 2016. – Т. 3, № 1.
11. Федюкович, П. А. Формирование тестовых последовательностей с помощью SAT-решателя / П. А. Федюкович, Е. В. Елаев, Н. С. Мащинский, В. М. Гришкин // Процессы управления и устойчивость. – 2015. – Т. 2, № 1. – С. 521-526.
12. Елаев, Е. В. Подходы к моделированию микропроцессоров для построения контрольно-диагностических тестов / Е. В. Елаев, Ю. Л. Степанов, В. В. Ферсенков // Процессы управления и устойчивость. – 2015. – Т. 2, № 1. – С. 398-403.
13. Свидетельство о государственной регистрации программы для ЭВМ № 2017612352 Российская Федерация. "Программное обеспечение комплексной разработки инструментальных тестов цифровых устройств" (CRIT): № 2016664716: заявл. 28.12.2016 : опубл. 20.02.2017 /

В. М. Гришкин, Д. А. Овсянников, Г. С. Лопаткин, Е. В. Елаев; заявитель Общество с ограниченной ответственностью «Центр информационно-диагностических систем СПбГУ» (ООО «Центр ИНДИС СПбГУ»).

14. Lala, P. K. Digital Circuits Testing and Testability. - New York: Academic Press, 1997. - 199 pp.
15. Bergeron, J. Writing Testbenches. Functional Verification of HDL Models. - Boston: Academic Publishers, 2000. - 354 pp.
16. Панков, Д. А. Автоматизация разработки и тестирования цифровых систем связи с многоуровневой архитектурой / Д. А. Панков, И. А. Панков, Л. А. Денисова // Автоматизация в промышленности. – 2023. – № 1. – С. 31-35.
17. Панков, Д. А. Алгоритм поиска ошибок программ и стенд тестирования для уникального радиотехнического устройства / Д. А. Панков // Техника радиосвязи. – 2022. – № 1(52). – С. 72-82.
18. Панков, Д. А. Контроль и диагностика неисправностей программно-аппаратного комплекса / Д. А. Панков, Л. А. Денисова // Омский научный вестник. – 2018. – № 2(158). – С. 128-133.
19. Иванов, И. А. Информационная модель процесса проектирования контролепригодных радиоэлектронных средств / И. А. Иванов, С. У. Увайсов // Информационные технологии. – 2011. – № 12. – С. 45-47.
20. Увайсов, С. У. Методика обеспечения диагностируемости электронных средств космических аппаратов по ранговому критерию на ранних этапах проектирования / С. У. Увайсов, И. А. Иванов, Н. А. Кошелев // Качество. Инновации. Образование. – 2012. – № 1(80). – С. 60-63.
21. Увайсов, Р. И. Обеспечение контролепригодности радиоэлектронных средств в рамках CALS-технологий / Р. И. Увайсов, С. У. Увайсов, И. А. Иванов // Качество. Инновации. Образование. – 2011. – № 1(68). – С. 43-47.
22. Иджеллиден, С. Б. Вероятностные оценки значений параметров электрорадиоэлементов в задачах диагностирования радиотехнических

устройств / С. Б. Иджеллиден, А. В. Долматов, С. У. Увайсов, А. Э. М. Алкадарский // Труды международного симпозиума "Надежность и качество". – 2005. – Т. 1. – С. 27.

23. Иванов, И. А. Метод контролепригодного проектирования радиоэлектронных средств / И. А. Иванов, Р. И. Увайсов, С. У. Увайсов // Инновации в условиях развития информационно-коммуникационных технологий. – 2007. – № 1. – С. 225-226.
24. Кофанов, Ю. Н. Комплексная электротепломеханическая диагностическая модель электронного средства / Ю. Н. Кофанов, С. Ю. Сотникова, А. Н. Тихонов, С. У. Увайсов // Информационные технологии в проектировании и производстве. – 2014. – № 4(156). – С. 50-55.
25. Слинкин, Д. И. Анализ современных методов тестирования и верификации проектов сверхбольших интегральных схем / Д. И. Слинкин // Программные продукты и системы. – 2017. – № 3. – С. 401-408.
26. Ксенз, С. П. Диагностика и ремонтопригодность радиоэлектронных средств . - М.: Радио и связь, 1989. - 248 с.
27. Иванюк, А. А. Проектирование контролепригодных цифровых устройств / А. А. Иванюк, В. Н. Ярмолик. – Минск : Бестпринт, 2006. – 296 с.
28. Иванюк, А. А. Моделирование функциональных неисправностей цифровых устройств средствами языка VHDL / А. А. Иванюк // Информатика. – 2007. – № 1(13). – С. 30-39.
29. Ярмолик, В. Н. Мера различия для тестовых наборов при генерировании управляемых вероятностных тестов / В. Н. Ярмолик, В. В. Петровская, И. Мрозек // Информатика. – 2022. – Т. 19, № 4. – С. 7-26.
30. Mrozek, I. Multiple Controlled Random Testing / I. Mrozek, V. Yarmolik // Fundamenta Informaticae. – 2016. – Vol. 144, No. 1. – P. 23-43.
31. Mrozek, I. Optimal controlled random tests / I. Mrozek, V. Yarmolik // Lecture Notes in Computer Science. – 2017. – Vol. 10244 LNCS. – P. 27-38.

32. Золоторевич, Л. А. Моделирование неисправностей СБИС на поведенческом уровне на языке VHDL / Л.А. Золоторевич // Информатика. - 2005. - №3. - С. 135-144.
33. Золоторевич, Л. А. Модели неисправностей при верификации проектов и контроле цифровых систем / Л. А. Золоторевич // Компьютерные науки и информационные технологии : Материалы Международной научной конференции, Саратов, 02–03 июля 2018 года. – Саратов: ИЦ "Наука", 2018. – С. 160-163.
34. Ирзаев, Г. Х. Обеспечение и оценка контролепригодности конструкции сложных радиоэлектронных средств на этапах их проектирования / Г. Х. Ирзаев // Контроль. Диагностика. – 2023. – Т. 26, № 5(299). – С. 34-41.
35. Сперанский, Д. В. Моделирование, тестирование и диагностика цифровых устройств / Д. В. Сперанский, Ю. А. Скобцов, В. Ю. Скобцов. – Москва : Интернет-Университет Информационных Технологий (ИНТУИТ), 2012. – 439 с.
36. Быханова, Н. В. Поиск рациональной структуры тестового генератора для подсистем встроенного самотестирования цифровых схем / Н. В. Быханова, С. Г. Мосин // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). – 2020. – № 1. – С. 89-94.
37. Мосин, С. Г. Метод синтеза тестовых программ для аналого-цифровых интегральных схем с применением сети автоматов / С. Г. Мосин // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). – 2016. – № 2. – С. 32-37.
38. Мельник, В. Методика разработки тест-программ контроля и диагностики цифровых устройств с использованием САПР SimTest / В. Мельник, В. Гришкин, А. Михайлов, Д. Овсянников // Электроника: Наука, технология, бизнес. – 2013. – № S(128). – С. 118-124.
39. Лопаткин, Г. С. Подход к автоматизации тестирования электронных цифровых устройств / Г. С. Лопаткин // Вестник Санкт-Петербургского

университета. Прикладная математика. Информатика. Процессы управления. – 2013. – № 4. – С. 90-98.

40. Гришкин, В. М. Интерфейсный метод построения моделей входных воздействий для тестирования электронных цифровых модулей / В. М. Гришкин, Г. С. Лопаткин, А. Н. Михайлов, Д. А. Овсянников // Вопросы радиоэлектроники. – 2013. – Т. 1, № 1. – С. 80-89.
41. Степанов, Ю. Л. Автоматизированное построение тестов цифровых электронных модулей для комплекса тестового контроля и диагностики УТК-512 / Ю. Л. Степанов, В. М. Гришкин, А. А. Большаков [и др.] // Вопросы радиоэлектроники. – 2012. – Т. 1, № 1. – С. 79-89.
42. Гришкин, В. М. Подход к разработке тестов цифровых электронных модулей для автоматического тестового оборудования / В. М. Гришкин, Ю. Л. Степанов, Г. С. Лопаткин, А. А. Большаков // Вопросы радиоэлектроники. – 2013. – Т. 1, № 1. – С. 89-99.
43. Шубарев, В. А. Развитие средств тестового контроля и диагностики радиоэлектронной аппаратуры в ОАО "Авангард" / В. А. Шубарев, А. Н. Михайлов, С. А. Берлик [и др.] // Вопросы радиоэлектроники. – 2014. – Т. 1, № 2. – С. 63-80.
44. Bushnell, M. L. Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits / M. L. Bushnell, V.D. Agrawal. — NY : Springer New York, 2002. — 690 p.
45. Thoulath, B. Compact test set method for high fault coverage test pattern generation / B. Thoulath, V. Baulkani // International Journal of Applied Engineering Research. — 2015. — Vol. 10, № 55. — P. 453–458.
46. Matinnejad, R. Search-based automated testing of continuous controllers: Framework, tool support, and case studies / R. Matinnejad, S. Nejati, L. Briand et al. // Information and Software Technology. — 2015. — Vol. 57, № 1. — P. 705–722.
47. Скобцов, Ю. А. Генетический алгоритм построения функциональных тестов арифметико-логических устройств / Ю. А. Скобцов, Д. Е. Иванов,

В. Ю. Скобцов // Восточно-Европейский журнал передовых технологий. – 2014. – Т. 2, № 9(68). – С. 9-13.

48. Bhowmik, B. Beyond test pattern generation: Coverage analysis / B. Bhowmik, J. Deka, S. Biswas // 2015 International Conference on Industrial Instrumentation and Control (ICIC). — 2015. — P. 1620–1625.
49. Eggersglu, S. Compact test set generation for test compression-based designs / S. Eggersglu // 20th IEEE European Test Symposium (ETS). — 2015. — P. 1620–1625.
50. Рувинова, Э. Автоматизированный оптический контроль печатных узлов / Э. Рувинова // Электроника: Наука, технология, бизнес. – 2002. – № 6(42). – С. 26-35.
51. Самсоненко, А. С. Автоматизация оптического контроля при производстве печатных сборок методом поверхностного монтажа / А. С. Самсоненко, Р. А. Филиппов, Л. Б. Филиппова // Вестник МГТУ "Станкин". – 2017. – № 2(41). – С. 61-65.
52. Аверченков, В. И. Автоматизация управления оптической инспекцией при контроле качества пайки печатных узлов / В. И. Аверченков, А. С. Самсоненко // Вестник Брянского государственного технического университета. – 2016. – № 2(50). – С. 149-155.
53. Tong, X. Improving accuracy of automatic optical inspection with machine learning / X. Tong, Z. Yu, X. Tian et al. // Frontiers of Computer Science. — 2021. — Vol. 1, № 2(50). — P. 1–12.
54. Астрицкий, А. Системы автоматического оптического контроля монтажа печатных плат Tyco Electronics Automation Group / А. Астрицкий // Компоненты и технологии. – 2003. – № 3(29). – С. 172-175.
55. Dom, B. E. Recent advances in the automatic inspection of integrated circuits for pattern defects / B. E. Dom, V. Brecher // Machine Vision and Applications. — 1995. — Vol. 8, № 1. — P. 5–19.
56. Albee, V. J. The evolution of ICT: Pcb technologies, test philosophies, and manufacturing business models are driving in-circuit test evolution and

innovations / V. J. Albee // IPC APEX EXPO Conference and Exhibition 2013. — 2013. — № 1. — P. 381–401.

57. Holtzer, M. In-circuit pin testing: An excellent potential source of value creation / M. Holtzer // SMT Surface Mount Technology Magazine. — 2015. — Vol. 30, № 6. — P. 68–71.
58. Городецкий, А. Снова о внутрисхемном тестировании ICT / А. Городецкий // Компоненты и технологии. – 2011. – № 7(120). – С. 58-59.
59. Городецкий, А. Снова о внутрисхемном тестировании ICT / А. Городецкий // Компоненты и технологии. – 2011. – № 8(121). – С. 44-45.
60. Городецкий, А. Снова о внутрисхемном тестировании ICT / А. Городецкий // Компоненты и технологии. – 2011. – № 9(122). – С. 6-7.
61. Ефремов, И. А. Библиотека компонентов внутрисхемного тестирования смешанных интегральных схем / И. А. Ефремов, С. Г. Мосин, М. А. Кисляков // Программные продукты и системы. – 2014. – № 1. – С. 187-190.
62. Мосин, С .Г. Метод синтеза тестовых программ для аналого-цифровых интегральных схем с применением сети автоматов / С. Г. Мосин // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). – 2016. – № 2. – С. 32-37.
63. Дианов, В. Н. Исследование методов выявления скрытых дефектов печатных плат / В. Н. Дианов, М. Н. Миронов // Труды международного симпозиума «Надежность и качество». – 2010. – Т. 2. – С. 14-15.
64. Хаханов, В. И. Модели и методы диагностирования цифровых систем на кристаллах / В. И. Хаханов, Е. И. Литвинова, О. А. Гузь, С. В. Чумаченко // Проблемы разработки перспективных микро- и наноэлектронных систем (МЭС). – 2012. – № 1. – С. 22-29.
65. Зайцев, А. Полеты наяву. Тестирование печатных плат с использованием метода «летающих щупов» / А. Зайцев // Эксперт. – 2014. – № 3. – С. 8-19.

66. Медведев, А. Scorpion Technologies открывает новые возможности во внутрисхемном тестировании / А. Медведев // Технологии в электронной промышленности. – 2006. – № 2(8). – С. 80-83.
67. Воронин, В. Применение установки SPEA 4060 для входного контроля печатных плат / В. Воронин // Электроника: Наука, технология, бизнес. – 2021. – № 9(210). – С. 100-103.
68. Брылев, М. Тестирование печатных плат: ICT vs FPT / М. Брылев // Технологии в электронной промышленности. – 2021. – № 1(125). – С. 52-53.
69. Марков, И. Автоматическое тестовое оборудование с подвижными пробниками в производстве электронных изделий / И. Марков, И. Рыков // Компоненты и технологии. – 2005. – № 1(45). – С. 168-170.
70. Марков, И. Автоматическое тестовое оборудование с подвижными пробниками в производстве электронных изделий / И. Марков, И. Рыков // Компоненты и технологии. – 2005. – № 2(46). – С. 204-207.
71. Марков, И. Автоматическое тестовое оборудование с подвижными пробниками в производстве электронных изделий / И. Марков, И. Рыков // Компоненты и технологии. – 2005. – № 3(47). – С. 224-227.
72. Jayapradha, V. Test coverage analysis of memory cluster testing using JTAG/ V. Jayapradha, S. Ravi, R. Kamalakkannan, S. Selvakumar // International Journal of Applied Engineering Research. — 2014. — Vol. 9, № 22. — P. 11861–11870.
73. Renbi, A. Contactless testing of circuit interconnects / A. Renbi, J. Delsing // Journal of Electronic Testing: Theory and Applications (JETTA). — 2015. — Vol. 31, № 3. — P. 229–253.
74. Aleksejev, I. Virtual reconfigurable scan-chains on FPGAs for optimized board test / I. Aleksejev, S. Devadze, A. Jutman, K. Shibin // 2015 16th Latin-American Test Symposium (LATS). — P. 1–6.
75. Ren, X. Detection of illegitimate access to JTAG via statistical learning in chip / X. Ren, V. G. Tavares, R. D. S. Blanton // IEEE Transactions on Computer Aided

Design of Integrated Circuits and Systems. — 2015. — Vol. 34, № 1. — P. 136–149.

76. Nelson, R. JTAG and embedded test complement ATE / R. Nelson // EE: Evaluation Engineering. — 2014. — Vol. 3, № 53. — P. 14–17.
77. Shashidhara, H. B. Board level JTAG/boundary scan test solution / H. B. Shashidhara, S. Yellampalii, V. Goudanavar // Proceedings of International Conference on Circuits, Communication, Control and Computing, Bangalore, India. — 2014. — P. 73–76.
78. Yin, X. H. On a method of getting test data for boundary scan interconnection test in multiple scan chains / X. H. Yin, C. F. Xu // Advanced Materials Research. — 2014. — Vol. 986-987. — P. 1531–1535.
79. Кожанов, К. Д. Интерфейс JTAG. Сравнение программаторов JTAG / К. Д. Кожанов // Colloquium-Journal. – 2020. – № 1-2(53). – С. 63-65.
80. Занг, В. Т. Особенность применения технологии JTAG в диагностике печатных узлов / В. Т. Занг, А. К. Дао, Л. К. Х. Фам [и др.] // Инновационные, информационные и коммуникационные технологии : сборник трудов XVII Международной научно-практической конференции, Сочи, 01–10 октября 2020 года / под.ред. С. У. Увайсов. – Москва: Ассоциация выпускников и сотрудников ВВИА имени профессора Н. Е. Жуковского содействия сохранению исторического и научного наследия ВВИА имени профессора Н. Е. Жуковского, 2020. – С. 427-431.
81. Федотов, А. И. Исследование возможностей интерфейса JTAG для диагностики микропроцессорных систем / А. И. Федотов, С. А. Тогузов // Новые технологии высшей школы. Наука, техника, педагогика : Материалы Всероссийской научно-практической конференции, Москва, 26 марта 2021 года. – Москва: Московский Политех, 2021. – С. 413-416.
82. Строгонов, А. Тестер цифровых БИС на базе JTAG, поддерживающих технологию перифирийного сканирования / А. Строгонов, С. Цыбин, А. Быстрицкий // Компоненты и технологии. – 2005. – № 3(47). – С. 138-143.

83. Амелина, М. А. Программа схемотехнического моделирования Micro-Cap. Версии 9, 10 / М. А. Амелина, С. А. Амелин. – 2-е, Исправленное, Дополненное. – Санкт-Петербург : Издательство Лань, 2014. – 632 с.
84. Micro-Cap и схемотехническое моделирование [Электронный ресурс] – Режим доступа: https://microcap-model.narod.ru/ (дата обращения: 20.06.2018).
85. Тестирование электронных устройств и измерительного оборудования / Multisim [Электронный ресурс] – Режим доступа: https://labview.izmeril.ru/multisim (дата обращения: 21.06.2018)
86. Колесникова, Т. Работа с виртуальными приборами в программной среде NI Circuit Design Suite - Multisim 12.0. Часть 1 / Т. Колесникова // Компоненты и технологии. – 2014. – № 1(150). – С. 158-161.
87. Колесникова, Т. Работа с виртуальными приборами в программной среде NI Circuit Design Suite - Multisim 12.0. Часть 2 / Т. Колесникова // Компоненты и технологии. – 2014. – № 2(151). – С. 129-132.
88. Колесникова, Т. Работа с виртуальными приборами в программной среде NI Circuit Design Suite - Multisim 12.0. Часть 3 / Т. Колесникова // Компоненты и технологии. – 2014. – № 3(152). – С. 162-166.
89. Колесникова, Т. Работа с виртуальными приборами в программной среде NI Circuit Design Suite - Multisim 12.0. Часть 4 / Т. Колесникова // Компоненты и технологии. – 2014. – № 4(153). – С. 158-162.
90. Бордиян, Р. Н. Перспектива применения среды Multisim для построения цифровых портретов / Р. Н. Бордиян, С. В. Кучевский, В. М. Дмитриев // Межвузовский сборник научных трудов : Сборник статей. Том Выпуск 26. – Краснодар: Федеральное государственное казенное военное образовательное учреждение высшего образования «Краснодарское высшее военное авиационное училище летчиков имени Героя Советского Союза А. К. Серова» Министерства обороны Российской Федерации, 2022. – С. 209-213.

91. Масляев, С. И. Основы работы в среде Circuit Maker : учебное пособие / С. И. Масляев ; С. И. Масляев. – Саранск : Изд-во Мордовского ун-та, 2005. — 106 с.
92. CircuitMaker Documentation [Электронный ресурс] – Режим доступа: https://www.altium.com/ru/documentation/altium-circuitmaker (дата обращения: 18.09.2020).
93. PSpice — моделирование схем [Электронный ресурс] – Режим доступа: https://www.pcbsoft.ru/pspice-tutorials (дата обращения: 17.08.2020).
94. Харлап, С. Н. Автоматизированный анализ результатов моделирования в PSpice электронных схем / С. Н. Харлап, Д. В. Сущинский // Проблемы и перспективы развития транспортных систем и строительного комплекса, Гомель, 27 октября 2013 года. – Гомель: Учреждение образования "Белорусский государственный университет транспорта", 2013. – С. 141-142.
95. Глухов, А. В. Проектирование электронных устройств в схемотехническом редакторе PSpice Schematics : Учебное пособие / А. В. Глухов, В. В. Шубин, Л. Г. Рогулина. – Новосибирск : Сибирский государственный университет телекоммуникаций и информатики, 2016. – 77 с.
96. Опадчий, Ю. Ф. Общая технология проектирования в среде Quartus II / Ю. Ф. Опадчий. — Москва: МАТИ, 2005. — 123 с.
97. Корнилков, А. Н. Диагностические возможности САПР Quartus II фирмы Altera / А. Н. Корнилков // Вестник Пермского университета. Математика. Механика. Информатика. – 2016. – № 1(32). – С. 22-28.
98. Антонов, А. Средства системной отладки САПР Quartus II / А. Антонов, А. Филиппов, Р. Золотухо // Компоненты и технологии. – 2008. – № 12(89). – С. 53-60.
99. Тюрин, С. Ф. Анализ настроек логических элементов при проектировании конечного автомата в системе QUARTUS II / С. Ф. Тюрин // Наука и технологические разработки. – 2015. – Т. 94, № 2. – С. 17-27.

100. Андреев, А. Е. Архитектуры вычислительных машин и систем / А. Е. Андреев, В. А. Егунов, П. Д. Кравченя [и др.] ; ВолгГТУ. – Волгоград : Волгоградский государственный технический университет, 2021. – 96 с.
101. Mourad, S. Principles of testing electronic systems / S. Mourad, Y. Zorian. — John Wiley and Sons, 2000. — 440 pp.
102. Gajsky, D. Principles of Digital Design / D. Gajsky, Y. Zorian. — Prentice Hall, New Jersey, 1997. — 447 pp
103. Kang, S. CMOS Digital Integrated Circuits. Analysis and Design / S. Kang, Y. Lebelevici. — Boston, McGrow-Hill, 2005. — 672 pp.
104. Воропаев, В. К. VHDL и Verilog-HDL-языки описания цифровой аппаратуры / В. К. Воропаев, А. Ю. Медведков, Ж. Б. Садыков // Динамика систем, механизмов и машин. – 2014. – № 4. – С. 15-18.
105. Курганский, С. И. Моделирование программируемых логических интегральных схем с использованием Verilog HDL / С. И. Курганский, М. А. Цырлов, Д. В. Матюшин // Информационные технологии в проектировании и производстве. – 2010. – № 4. – С. 48-51.
106. Компанейц, А. Н. Сравнение цифровых устройств, разработанных на логическом и поведенческом уровнях с использованием языка Verilog HDL / А. Н. Компанейц, Д. О. Кузина // Автоматизация, мехатроника, информационные технологии : Материалы V Международной научно-технической интернет-конференции молодых ученых, Омск, 19 мая 2015 года. – Омск: федеральное государственное бюджетное образовательное учреждение высшего профессионального образования "Омский государственный технический университет", 2015. – С. 164-168.
107. Компанейц, А. Н. Исследование параметров схемы на базе ПЛИС при её реализации в Block Diagram/Schematic Quartus II и на языке Verilog HDL / А. Н. Компанейц, С. А. Перепечко // Автоматизация, мехатроника, информационные технологии Automation, Mechatronics, Information Technologies : материалы IV Международной научно-технической интернет-

конференции молодых ученых, Омск, 14–15 мая 2014 года. – Омск: федеральное государственное бюджетное образовательное учреждение высшего профессионального образования "Омский государственный технический университет", 2014. – С. 150-153.

108. Yamin, L. Computer Principles and Design in Verilog HDL / L. Yamin. — John Wiley & Sons Limited, 2018. — 575 pp.
109. Поляков, А. К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры на ПЛИС / А. К. Поляков. – Москва : Издательский дом «МЭИ», 2012. – 221 с.
110. Угрюмов, Е. П. Цифровая схемотехника (3-е изд.) / Е. П. Угрюмов. — СПб: БХВ-Петербург, 2010. — 816 с.
111. Шило, В. Л. Популярные цифровые микросхемы: Справочник (2-е изд.) / В. Л. Шило. — М: Радио и связь, 1989. — 352 с.
112. Тули, М. Справочное пособие по цифровой электронике / М. Тули. — М: Энергоатомиздат, 1991. — 176 с
113. Власов, А. Б. Электроника. Цифровые элементы и узлы электронной аппаратуры / А. Б. Власов. — М: Инфра-Инженерия, 2022. — 216 с.
114. Алексеенко, А. Г. Микросхемотехника / А. Г. Алексеенко, И. И. Шагурин. — Москва: Радио и связь, 1990. — 496 с
115. Наундорф, У. Аналоговая электроника. Основы, расчет, моделирование / У. Наундорф. — Москва: Техносфера, 2008. — 472 с.
116. Красько, А. С. Схемотехника аналоговых электронных устройств: Учебное пособие / А. С. Красько. — Томск: Томский государственный университет систем управления и радио-электроники, 2005. — 178 с.
117. Крекрафт, Д. Аналоговая электроника. Схемы, системы, обработка сигнала / Д. Крекрафт, С. Джерджли. — Москва: Техносфера, 2005. — 360 с.
118. Волович, Г. И. Схемотехника аналоговых и аналого-цифровых электронных устройств / Г. И. Волович. — Москва: Издательский дом «ДодэкаXXI», 2005. — 528 с.

119. Лаврентьев, Б. Ф. Схемотехника электронных средств / Б. Ф. Лаврентьев. — Москва: Издательский центр «Академия», 2010. — 336 с.
120. Кучумов, А. И. Электроника и схемотехника / А. И. Кучумов. — Москва: Гелиос АРВ, 2004. — 336 с.
121. Петин, Г. П. Аналоговая схемотехника / Г. П. Петин. — Ростов-на-Дону: Южный федеральный университет, 2010. — 314 с.
122. Максфилд, К. Проектирование на ПЛИС. Архитектура, средства и методы. Курс молодого бойца / К. Максфилд. — Додэка XXI, ДМК Пресс, 2015. — 408 с.
123. Михайлов, А. Н. Модернизированная ЯСТЕК-среда для управления контрольно-диагностическими комплексами РЭА нового поколения / А. Н. Михайлов, О. И. Иванов, В. И. Пустоветов // Вопросы радиоэлектроники. – 2013. – Т. 1, № 1. – С. 115-119.
124. Петровский, И. И. Логические ИС КР1533, КР1554. Справочник. / И. И. Петровский, А. В. Прибыльский, А. А. Троян, В. С. Чувелев. — М: Бином, 1998. — 508 с.

Список иллюстративного материала

Приложение А

Свидетельство о государственной регистрации программы для ЭВМ.

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2017612352

«Программное обеспечение комплексной разработки инструментальных тестов цифровых устройств» (CRIT)

Правообладатель: ***Общество с ограниченной ответственностью «Центр информационно-диагностических систем СПбГУ» (ООО «Центр ИНДИС СПбГУ») (RU)***

Авторы: ***Гришкин Валерий Михайлович (RU), Овсянников Дмитрий Александрович (RU), Лопаткин Григорий Сергеевич (RU), Елаев Евгений Валерьевич (RU)***

Заявка № **2016664716**

Дата поступления **28 декабря 2016 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***20 февраля 2017 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

Приложение Б

Акт об использовании результатов диссертационной работы в производственной деятельности в АО «Производственная Компания «Специальные Инновационные Технологии».

Акционерное общество
«Производственная Компания
«Специальные Инновационные Технологии»
198095, Россия, г. Санкт-Петербург, Химический пер., д. 1ав, оф. 515
Телефон: (812) 339-96-99 Факс: (812) 339-96-46 e-mail: office@specinntech.ru
ОГРН: 1197847235244 ИНН/КПП: 7805759690/780501001

АКТ
об использовании результатов диссертационной работы
Елаева Евгения Валерьевича
«Автоматизация тестового контроля цифровых радиоэлектронных устройств»

Комиссия в составе:

председателя комиссии:

Зубенко Ю.Н. – генерального директора

членов комиссии:

Дорофеева А.А. – заместителя генерального директора;

Михайлова А.Н. – начальника отдела, профессора, д.т.н.

составила настоящий акт о том, что результаты диссертационной работы Елаева Евгения Валерьевича «Автоматизация тестового контроля цифровых радиоэлектронных устройств» могут быть использованы в производственной деятельности в Акционерном обществе «Производственная Компания «Специальные Инновационные Технологии» при тестировании цифровых радиоэлектронных устройств, а именно:

1) метод формирования программных моделей цифровых микросхем в контексте задач тестирования;
2) алгоритм интерфейсного метода автоматизированного построения тест-программ;
3) программный комплекс CRIT, автоматизирующий процесс построения тестовых программ.

Применение разработанных методов, алгоритмов и реализующих их программного обеспечения позволит повысить эффективность качество тестового контроля систем цифровых радиоэлектронных устройств.

Председатель комиссии Зубенко Ю.Н.

Члены комиссии Дорофеев А.А.

Михайлов А.Н.

25.06.2025

СПЕЦИНТЕХ

Система менеджмента качества
соответствует требованиям
ГОСТ Р ИСО 9001
ГОСТ РВ 0015-002
Сертификат выдан 20.07.2020 г.

Приложение В

Акт об использовании результатов диссертационной работы в учебном процессе.

АКТ
об использовании результатов диссертационной работы
Елаева Евгения Валерьевича
«Автоматизация тестового контроля цифровых радиоэлектронных устройств»
от 26.06.2025 года

Комиссия в составе:
председателя комиссии:

- декана факультета Прикладной математики — процессов управления Санкт-Петербургского государственного университета, профессора, д.ф.-м.н. Петросяна Леона Аганесовича;

членов комиссии:

- заведующего кафедрой Теории систем управления электрофизической аппаратурой», профессора, д.ф.-м.н., Овсянникова Дмитрия Александровича;
- доцента кафедры Теории систем управления электрофизической аппаратурой, к.ф.-м.н., Едаменко Николая Семеновича

удостоверяет, что результаты диссертационной работы Елаева Евгения Валерьевича «Автоматизация тестового контроля цифровых радиоэлектронных устройств» внедрены в учебный процесс кафедры «Теория систем управления электрофизической аппаратурой» при подготовке выпускных бакалаврских работ и магистерских диссертаций студентов, при чтении курса по дисциплине «Современные проблемы естествознания» для обучающихся по основным образовательным программам высшего образования «Прикладные математика, физика и процессы управления» уровня бакалавриат направления «03.03.01. Прикладные математика и физика» и «Математические и информационные технологии» уровня магистратура направления «03.04.01. Прикладные математика и физика», а также при проведении дополнительной профессиональной образовательной программы «Основы проектирования тестов цифровых устройств».

В указанных курсах и при подготовке выпускных работ студентов широко использовались разработанные диссертантом подходы и методы к моделированию цифровых устройств и автоматизации процессов их тестирования; методы интегрирования нецифровых аналоговых компонентов в программную модель объекта контроля; алгоритм интерфейсного метода автоматизированного построения тест-программ, а также разработанный программный комплекс CRIT, автоматизирующий процесс построения тестовых программ.

Председатель комиссии Петросян Л.А.

Члены комиссии Овсянников Д.А.

Едаменко Н.С.

Личную подпись Л.А. Петросяна, Д.А. Овсянникова, Н.С. Едаменко заверяю
И.О. начальника отдела кадров
И.И. Константинова
26.06.2025

Приложение Г

Благодарственное письмо за участие в организации, подготовке и проведения курса «Основы проектирования тестов цифровых устройств» для сотрудников ОАО «Авангард».

ОТКРЫТОЕ АКЦИОНЕРНОЕ ОБЩЕСТВО
АВАНГАРД

Кондратьевский пр., д. 72
Санкт-Петербург, 195271
Тел.: (812) 540-15-50
Факс: (812) 545-37-85
e-mail: avangard@avangard.org
http://www.avangard.org

14.04.2015 № 41-ОК
На № _____ от _____

БЛАГОДАРСТВЕННОЕ ПИСЬМО

Отдел подготовки кадров ОАО «Авангард» выражает благодарность сотруднику Санкт-Петербургского Государственного университета ЕЛАЕВУ ЕВГЕНИЮ ВАЛЕРЬЕВИЧУ за его деятельное участие в организации, подготовке и практической реализации курса повышения квалификации для группы сотрудников предприятия по тематике «Основы проектирования тестов цифровых устройств».

По результатам обучения сотрудники отмечают ярко выраженную практическую направленность подготовки, высокий методический и организационный уровень учебных занятий, проведенных Евгением Валерьевичем, и, как следствие, высокую результативность курса в плане формирования практических навыков разработки тестов.

Благодарим за плодотворное сотрудничество.

Желаем дальнейших успехов в работе.

Начальник отдела подготовки кадров А.П. Меньков

ОТДЕЛ КАДРОВ

Приложение Д

Программная модель микросхемы 1533ТМ2, созданная на основе функционального метода:

```
module ic_1533tm2 (
input S, R, C, D,
output Q, Q_inv);
reg Q_temp;

always @(C)
begin
if (C == 1 && C == 1'b1 && R == 1'b1)
Q_temp <= D;
else if ( C == 1'b0 && S == 1'b1 && R == 1'b1)
Q_temp<= Q;
end

assign Q = (S != R) ? R :
(R == 1'b1 && R == 1'b1) ? Q_temp : 1'b1;
assign Q_inv = ~ Q;

endmodule
```

Программная модель микросхемы 1533ТМ2, созданная на основе интреграционного метода:

```
module 1533TM2 (
S,
R,
C,
D,
```

```
Q,
Q_inv
);


input S;
input R;
input C;
input D;
output Q;
output Q_inv;

wire connection_1;
wire connection_2;
wire connection_3;
wire connection_4;
wire out_Q;
wire out_Q_inv;

ic_1533la4 b2v_LOGIC_ELEM_1(
.i1(S),
.i2(connection_4),
.i3(connection_2),
.o(connection_1));

ic_1533la4 b2v_LOGIC_ELEM_2(
.i1(connection_1),
.i2(R),
```

```
.i3(C),
.o(connection_2));

ic_1533la4 b2v_LOGIC_ELEM_3(
.i1(connection_2),
.i2(C),
.i3(connection_4),
.o(connection_3));

ic_1533la4 b2v_LOGIC_ELEM_4(
.i1(connection_3),
.i2(R),
.i3(D),
.o(connection_4));

ic_1533la4 b2v_LOGIC_ELEM_5(
.i1(S),
.i2(connection_2),
.i3(out_Q_inv),
.o(out_Q));

ic_1533la4 b2v_LOGIC_ELEM_6(
.i1(out_Q),
.i2(R),
.i3(connection_3),
.o(out_Q_inv));
assign Q = out_Q;
assign Q_inv = out_Q_inv;
```

```
endmodule
```

Приложение Е

Программная модель микросхемы *1533АП5*:

```
module ic_1533ap5(
input D1,D2,D3,D4,E_inv,
output Q1,Q2,Q3,Q4);

assign  Q1 = (E_inv == 0) ? D1 : 1'bz;
assign  Q2 = (E_inv == 0) ? D2 : 1'bz;
assign  Q3 = (E_inv == 0) ? D3 : 1'bz;
assign  Q4 = (E_inv == 0) ? D4 : 1'bz;

endmodule
```

Программная модель микросхемы *1533ИД3*:

```
module ic_1533id3 (
input D1,D2,D3,D4,C1_inv,C2_inv,
output Y0_inv,Y1_inv,Y2_inv,Y3_inv,Y4_inv,Y5_inv,Y6_inv,
Y7_inv,Y8_inv,Y9_inv,Y10_inv,Y11_inv,Y12_inv,Y13_inv,
Y14_inv,Y15_inv);

wire[15:0] Y, Y_inv;
wire[3:0] D;

assign D = {D4,D3,D2,D1};

assign  Y = (D == 4'b0000) ? 16'b1111111111111110 :
(D == 4'b0001) ? 16'b1111111111111101 :
(D == 4'b0010) ? 16'b1111111111111011 :
(D == 4'b0011) ? 16'b1111111111110111 :
```

```
(D == 4'b0100) ? 16'b1111111111101111 :
(D == 4'b0101) ? 16'b1111111111011111 :
(D == 4'b0110) ? 16'b1111111110111111 :
(D == 4'b0111) ? 16'b1111111101111111 :
(D == 4'b1000) ? 16'b1111111011111111 :
(D == 4'b1001) ? 16'b1111110111111111 :
(D == 4'b1010) ? 16'b1111101111111111 :
(D == 4'b1011) ? 16'b1111011111111111 :
(D == 4'b1100) ? 16'b1110111111111111 :
(D == 4'b1101) ? 16'b1101111111111111 :
(D == 4'b1110) ? 16'b1011111111111111 :
16'b0111111111111111;

assign Y_inv = (C1_inv == 0 && C2_inv == 0) ? Y :
(C1_inv == 1  C2_inv == 1) ? 16'b1111111111111111 :
16'b1111111111111111;

assign {Y15_inv, Y14_inv, Y13_inv, Y12_inv, Y11_inv,
 Y10_inv, Y9_inv, Y8_inv, Y7_inv, Y6_inv, Y5_inv,
 Y4_inv, Y3_inv, Y2_inv, Y1_inv, Y0_inv} = Y_inv;

endmodule
```

Программная модель микросхемы *1533СП1*:

```
module ic_1533sp1 (A,B,AB,A0,A1,A2,A3,B0,B1,B2,B3,
Aout, Bout, ABout);

input A0,A1,A2,A3,B,A,AB,B0,B1,B2,B3;
output  Aout, Bout, ABout;
```

```
wire [3:0] a;
wire [3:0] b;

assign {a[3],a[2],a[1],a[0]}={A3,A2,A1,A0};
assign {b[3],b[2],b[1],b[0]}={B3,B2,B1,B0};

assign Aout=((a>b)((a==b)&&(A==1'b1)&&(B==1'b0)&&
(AB==1'b0))((a==b)&&(A==1'b0)&&(B==1'b0)&&
(AB==1'b0)))?(1'b1):(1'b0);

assign Bout =((a<b)((a==b)&&(A==1'b0)&&(B==1'b1)&&
(AB==1'b0))((a==b)&&(A==1'b0)&&(B==1'b0)&&
(AB==1'b0)))? (1'b1) : (1'b0);

assign ABout =((a==b)&&(AB==1'b1))?(1'b1):(1'b0);

endmodule
```

Программная модель микросхемы *1533ИР37*:

```
module ic_1533ir37 (
input D0,D1,D2,D3,D4,D5,D6,D7,EZ_inv,C,
output Q0,Q1,Q2,Q3,Q4,Q5,Q6,Q7);

wire[8:0] D;
reg[8:0] Q;

assign D = {D7,D6,D5,D4,D3,D2,D1,D0};
```

```
initial wait (EZ_inv == 1) begin
Q = 8'bz;
end

always@(EZ_inv) begin
if (EZ_inv == 1) Q = 8'bz; else Q = Q;
end

always@(posedge C) begin
if (EZ_inv == 1) Q = 8'bz; else Q = D;
end

assign {Q7,Q6,Q5,Q4,Q3,Q2,Q1,Q0} = Q;

endmodule
```