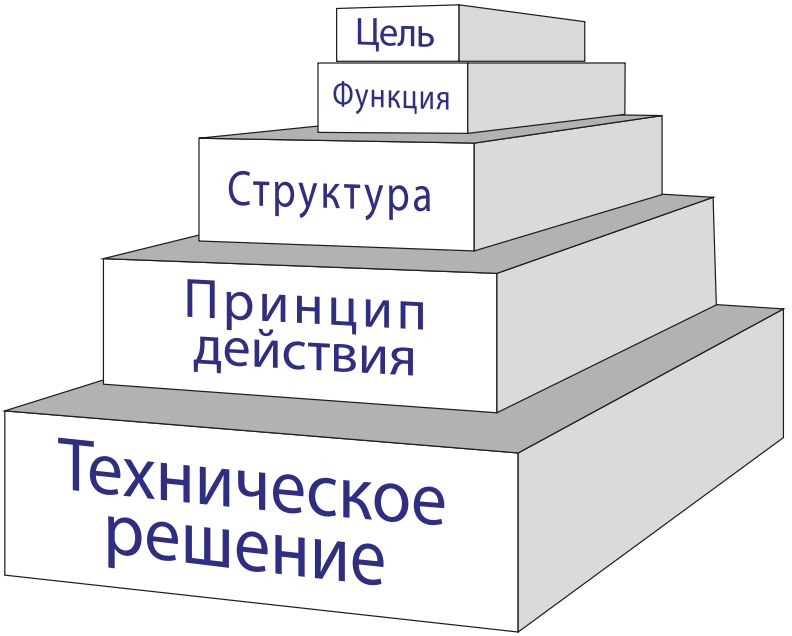
СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ | SIBERIAN FEDERAL UNIVERSITY

Б. И. Борде

Методы автоматизации проектирования неоднородных вычислительных систем и информационных моделей объектов



Министерство науки и высшего образования Российской Федерации
Сибирский федеральный университет

Б. И. Борде

МЕТОДЫ АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И ИНФОРМАЦИОННЫХ МОДЕЛЕЙ ОБЪЕКТОВ

Монография

Красноярск
СФУ
2020

УДК 004.41'22
ББК 32.97-02
Б820

Р е ц е н з е н т ы:

В. В. Шайдуров, доктор физико-математических наук, профессор, член-корреспондент РАН, руководитель научного направления ФИЦ «Красноярский научный центр СО РАН»;

А. П. Корпенко, доктор физико-математических наук, профессор, заведующий кафедрой САПР МГТУ им. Н. Э. Баумана;

Г. А. Доррер, доктор технических наук, профессор, профессор кафедры информационно-управляющих систем СибГУ им. М. Ф. Решетнёва

Борде, Б. И.

Б820 Методы автоматизации проектирования неоднородных вычислительных систем и информационных моделей объектов : монография / Б. И. Борде. – Красноярск : Сиб. федер. ун-т, 2020. – 212 с.

ISBN 978-5-7638-4097-1

Представлена открытая САПР, позволяющая из концептуального описания вариантов проекта в виде формализованного задания (ФЗ-FZ) получить проект или командный файл для автоматического выполнения. Время выполнения командного файла уменьшается на порядок по сравнению с ручным набором команд. Изложена технология, обобщающая проектирование на разных уровнях абстракции и развивающая комплексный подход STEM.

Предназначена для специалистов по комплексному проектированию объектов, управляемых вычислительными системами.

Электронный вариант издания см.:
http://catalog.sfu-kras.ru

УДК 004.41'22
ББК 32.97-02

ISBN 978-5-7638-4097-1

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 5

1. РАЗВИТИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СРЕДСТВ ИХ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ 10
   1.1. Развитие неоднородных вычислительных систем 10
   1.2. Уровни абстракции вычислительных систем 12
   1.3. Развитие структур неоднородных вычислительных систем 16
   1.4. Комплексы моделирования и проектирования неоднородных вычислительных систем 19

2. ПРИНЦИПЫ ПОСТРОЕНИЯ МНОГОУРОВНЕВЫХ СИСТЕМ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ 22
   2.1. Описание проектных решений вычислительных систем 22
   2.2. Структурный синтез вычислительных устройств и систем 29
   2.3. Аналого-цифровые устройства вычислительных систем 32
   2.4. Неоднородные вычислительные системы на базе микроЭВМ 36
   2.5. Вычислительные системы из сетевых устройств и вычислительных машин 45
   2.6. Процессы синтеза и анализа неоднородных вычислительных систем 47
   2.7. Принципы построения многоуровневых систем моделирования и проектирования вычислительных систем 49
   2.8. Информационные основы преобразования описаний в многоуровневых САПР 52
   2.9. Формализованное задание и алгоритмы для автоматизированного анализа 58

3. МНОГОФУНКЦИОНАЛЬНЫЕ МОДЕЛИ КОМПОНЕНТОВ 64
   3.1. Уровни абстракции компонентов 64
   3.2. Модели компонентов 65
   3.3. Представление моделей в САПР COD 71
   3.4. Модели компонентов обеспечения интерфейсов САПР COD с системой PCAD 72
   3.5. Автоматизация формирования моделей компонентов САПР PCAD 83
   3.6. Автоматизация формирования моделей текстовых описаний информационной поддержки жизненного цикла вычислительных систем 91

3.7. Модели компонентов обеспечения интерфейса САПР COD с САПР CATIA 96
3.8. Модели компонентов обеспечения интерфейса САПР COD со средой виртуальной реальности..... 100
3.9. Синтез модели и алгоритмов работы аналого-цифровой подсистемы с USB интерфейсом 101

4. РЕАЛИЗАЦИЯ МНОГОУРОВНЕВОЙ СИСТЕМЫ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ..... 109
4.1. Структура и функции программного комплекса..... 109
4.2. Информационное обеспечение моделирования и проектирования вычислительных устройств и систем..... 120
4.3. Формализованное задание для автоматизированного анализа 121
4.4. Параметры САПР и структуры данных..... 126
4.5. Интерфейсы исследовательской САПР COD с системой автоматизации проектирования PCAD 134
4.6. Маршруты проектирования модулей ЭВМ в системе PCAD200x 141
4.7. Результаты выполнения формализованных заданий 144

5. АВТОМАТИЗАЦИЯ ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ ОБЪЕКТОВ И ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ..... 155
5.1. Информационные модели объектов в САПР..... 155
5.2. Проектирование информационных моделей объектов на различных уровнях иерархии 161
5.3. Создание объекта в САПР REVIT 162
5.4. Проектирование кампуса в INFRAWORKS..... 166

6. ЛОКАЛЬНОЕ ВЫПОЛНЕНИЕ И СЕТЕВЫЕ СЕРВИСЫ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ..... 169
6.1. Программно-аппаратные комплексы САПР 169
6.2. Локальное выполнение формализованного задания на персональных ЭВМ в средах Windows, Linux и OS/2 171
6.3. Сетевые сервисы и выполнение формализованных заданий в среде Internet 182

ЗАКЛЮЧЕНИЕ..... 188

СПИСОК ЛИТЕРАТУРЫ 190

ПРИЛОЖЕНИЯ 203

ВВЕДЕНИЕ

Развитие вычислительной техники сопровождается увеличением сложности и разнообразия машин [1, 4, 6, 91] при снижении энергии переключения и стоимости выполнения вычислительных операций. Осуществление этих требований возможно только в условиях комплексной автоматизации производства средств вычислительной техники и компонент. Сложность компонент непрерывно возрастает, поэтому автоматизация проектирования средств вычислительной техники и компонент является актуальной задачей. Известны различные системы автоматизированного проектирования (САПР): промышленные, учебно-исследовательские и экспериментальные перспективные системы для проверки принципов действия.

За короткий срок в 60 лет (1956–2016) пройден путь от первых транзисторов НПО «Светлана» для макета ЭВМ и настольных аналоговых машин МН-7 на лампах до мобильных устройств и крупных дата-центров Сбербанка в Сколково [166, 168, 184], спроектированных в САПР REVIT [161]. Во всех машинах с индексом МН (модель нелинейная) использовались диоды на наборном поле для операций непрерывной логики и в блоках нелинейности [140, 141].

Промышленные САПР относятся к хорошо формализованным техническим этапам проектирования и подразделяются на специализированные и комплексные. Комплексные САПР отличаются расширенным множеством компонент и множеством соединителей и портов с различными принципами действия. Первой специализированной САПР вычислительных машин была САПР КАСПИ [9], созданная в ИТМиВТ РАН под руководством акад. С. А. Лебедева. В качестве специализированных промышленных САПР вычислительных систем используются отечественная система Прам5.3, ГРИФ4 [78] и зарубежные САПР для персональных ЭВМ: PCAD, ALTIUM, ORCAD, CADDY, PADS, EAGLE, CADSTAR. В качестве комплексных промышленных САПР стационарных объектов рассмотрим САПР фирм Autodesk Revit и Infra Works, АСКОН Renga, работающие с информационной моделью объекта BIM, в соответствии с ГОСТ и международными стандартами ISO [188–190]. В качестве комплексных промышленных САПР мобильных объектов рассмотрим САПР CATIA. Появление комплексных САПР для проектирования неподвижных и мобильных объектов, и стандартов их представления привело к созданию новых инст-

рументальных средств и возможности реализации STEM образования: учебный план, основанный на идее обучения четырём конкретным дисциплинам одновременно – науке (Science), технологии (Technology), инженерному искусству (Engineering) и математике (Math). STEM образование предполагает многовариантное проектирование и командную работу.

Важным является утверждение профессиональных стандартов для подготовки специалистов по созданию и модификации интеграционных решений, код 06.041, Минтруда РФ в 2017 году [191]. Целью подготовки является интеграция информационных систем и облачных сервисов. Работы должны проводиться с помощью автоматизированных систем. Профессиональные стандарты должны соответствовать потребностям промышленности в специалистах.

Учебно-исследовательские САПР (УИ САПР) соответствуют ранним слабоформализованным этапам проектирования. На ранних этапах проектирования желателен синтез и анализ множества вариантов вычислительных устройств и систем. Целью создания программно-методического комплекса COD (Conceptual Object Design) является повышение уровня сложности задач проектирования неоднородных вычислительных устройств и систем [12–15] и улучшение производительности труда инженера.

Для описаний проектируемых объектов на языке низкого уровня, используемых в промышленных САПР, объем описания пропорционален количеству вариантов. Описание объектов на языке высокого уровня позволяет приблизить объем описания множества объектов одного класса к объему описания одного объекта. С целью снижения объемов описаний используются многоуровневые модели вычислительных систем.

Уровням моделей вычислительных систем соответствуют различные инструментальные средства, комплекс которых реализован в системах автоматизированного проектирования. Различают САПР высокого и низкого уровней. САПР низкого уровня соответствуют нижним уровням абстракции моделей и служат для функционально-логического и конструкторского проектирования. САПР высокого уровня обеспечивают описание множества вариантов технических решений при системном и структурно-алгоритмическом проектировании.

Процесс проектирования сложных систем и ЭВМ в частности представляет итерационный процесс нисходящего и частично восходящего проектирования на нижних уровнях. САПР высокого уровня эффективны для автоматизации преобразования проектных решений при переходе к различным САПР низкого уровня. Комплекс инструментальных средств автоматизации преобразования проектных решений между САПР различных уровней назван в работе интерфейсом.

Проектное решение, воспринимаемое формальной системой, называется формализованным заданием, которое состоит из различных разделов описания проектных решений. Желателен однократный ввод формализованных заданий независимо от числа приложений. Преобразование единого формализованного задания в формат различных САПР обеспечивается интерфейсом между САПР высокого уровня и конкретной САПР низкого уровня. Стандарты на описание электронной аппаратуры EDIF (Electronic Data Interchange Format) и STEP (Standard for Exchange of Product data) ISO 10303 лишь частично обеспечивают преобразование, так как используются только для внешнего представления формализованных заданий.

Монография дополняет учебники [101–103, 110], и в ней рассматриваются многоуровневая модель вычислительной системы, особенности синтеза и анализа. Основное внимание уделено автоматизированному анализу неоднородных вычислительных систем с цифровыми и аналоговыми сигналами на различных уровнях абстракции с процедурно-ориентированным покомпонентным описанием множества технических решений на базе стандартных универсальных языков программирования высокого уровня и непроцедурными описаниями вариантов проектных решений на нижних уровнях.

Для автоматизированного проектирования, начиная с ранних этапов, под руководством автора разработана учебно-исследовательская САПР высокого уровня для автоматизации анализа работы вычислительных устройств и систем и автоматического преобразования формализованных заданий в формат САПР низкого уровня. Отличительной особенностью УИ САПР являются многократные использования единого формализованного задания для различных приложений и многофункциональные модели компонент с общим интерфейсом. Основным приложением является автоматизированный анализ работы устройства или системы, описанных в формализованном задании. В формализованном задании описываются структура устройства или системы и внешние воздействия. Основными результатами анализа являются временные диаграммы и оценка ресурсов, а дополнительными приложениями – преобразование формализованного задания в формат одной из промышленных САПР или язык виртуальной реальности. Автоматизация преобразования формализованного задания в формат конкретной САПР названа интерфейсом с этой САПР. Использование единого формализованного задания для различных приложений обеспечивается многофункциональными моделями компонент с общим интерфейсом. Интерфейс определяется таблицами синтаксиса и семантики для каждого типа компонент. Модели всех типов компонент для каждого приложения объединяются в статические или динамические библиотеки. Диалоговая оболочка обеспечивает соответствие приложения и библиотек моделей

компонент. Автоматизация анализа и оценка ресурсов позволяет инженеру или студенту сосредоточиться на творческих проектных процедурах синтеза описаний, а автоматическая оценка ресурсов облегчает выбор оптимального решения. Учебно-исследовательская САПР имеет интерфейс с промышленными САПР для импорта и экспорта описаний.

Программное обеспечение реализовано для персональных ЭВМ и серверов, что позволяет использовать программно-методический комплекс для локального и дистанционного обучения. На персональных ЭВМ комплекс реализован в средах Windows, LINUX и OS/2. Серверная часть комплекса выполнена в средах Windows, LINUX, OS/2, в подсистеме диалоговой обработки (ПДО или PTS) системы виртуальных машин S390 (VM/ESA, zVM). Формализованные задания могут быть выполнены на персональной ЭВМ или переданы для выполнения на серверы приложений по сети Интернет. В настоящее время все серверы размещаются на виртуальных машинах дата-центра СФУ и составляют основу системы дистанционных образовательных технологий (ДОТ) СФУ. Сетевые сервисы для обучения размещаются на e.sfu-kras.ru, а выполнение формализованных заданий проектов осуществляется на УИ САПР на серверах приложений. Множество примеров проектов размещены на серверах приложений. Серверы приложений могут сопрягаться с реальной аппаратурой. Серверы приложений и методический сервер размещены на виртуальных машинах дата-центра СФУ и доступны по сети Интернет. Таким образом обеспечивается среда для интерактивного онлайн-обучения. Результаты отсылаются пользователям по электронной почте или принимаются пользователем. В облачных структурах САПР размещается на дата-центре, и результаты хранятся там же. Пользователю разрешается скачать результаты или сделать их общедоступными. Облачные структуры более эффективны благодаря отсутствию необходимости приобретения и установки САПР на рабочих местах, доступу из браузера или сетевого устройства, но у пользователей возникают вопросы безопасности.

Программно-методический комплекс позволяет описывать множество технических решений в виде формализованных заданий на различных уровнях абстракции. Формализованные задания на универсальных языках высокого уровня унифицированы в различных синтаксических средах C++, ADA, PLI, JAVA. Обеспечивается диалоговая отладка формализованных заданий, автоматизированный анализ со сравнением предполагаемых и фактических результатов, а также автоматическая оценка ресурсов и документирование результатов. При удовлетворительных результатах анализа множество технических решений или оптимальные структуры могут быть переданы в промышленные САПР PCAD, ALTIUM, (PADS Mentor Graphics), ORCAD, CADDY, EAGLE, REVIT Autodesk, CADSTAR Zuken,

CATIA и PRAM 5.3. Примерами специализированных облачных САПР являются EasyEDA; комплексных Autodesk InfraWorks, Autocad 360, Fusion 360, Onshape [115].

Отраслевые нормы и правила используются в специализированных САПР по отраслям несмотря на их изменения во времени из-за развития материалов и технологий. В комплексных САПР все гораздо сложнее, и требуется большее время для реального использования систем правил (свода правил – СП). Развитие интернет-устройств (IOT) на базе систем на кристалле должно облегчить оценку состояний всех компонент и использование систем правил в САПР. Примерами энергоэффективной системы передачи данных для IOT являются LPWAN и NB (teleofis.ru).

Программно-методический комплекс УИ САПР <COD> представлен на оптическом диске (DVD-ROM), на серверах университета с сетевым доступом (e.sfu-kras.ru), содержит клиентские и серверные версии для операционных систем Windows, Linux, OS/2 и размещен в свободном доступе в библиотеке СФУ (bik.sfu-kras.ru).

Работы проведены в лаборатории неоднородных вычислительных систем кафедры вычислительной техники ИКИТ СФУ, используются в классах и в вычислительном центре Института космических и информационных технологий (ИКИТ) СФУ, а также в вычислительном дата-центре Сибирского Федерального университета. На виртуальных машинах дата-центра размещены методические материалы (e.sfu-kras.ru) и личные кабинеты сотрудников и студентов, а также серверы приложений для выполнения формализованных заданий проектов в различных средах с большим количеством примеров: CODWIN для Windows, CODLIN для Linux, CO-DOS для OS/2-EcomStation [50–57].

Работа выполнена при поддержке Министерства образования России по приказу № 195 от 16.03.1987 «Создание и развитие учебно-исследовательских программно-методических комплексов САПР (УИ ПМК САПР) в вузах». Пункт 3.2.17 приказа 195 утвердил работу автора по ПМК УИ САПР неоднородных вычислительных систем (ПМК УИ САПР НВС). Комплекс непрерывно развивается автором и дополняется лучшими проектами студентов ИКИТ СФУ, выполненными под руководством автора.

Пожелания по содержанию книги и комплекса можно направить профессору Б. И. Борде по электронной почте: bborde@sfu-kras.ru.

1. РАЗВИТИЕ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И СРЕДСТВ ИХ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ

1.1. Развитие неоднородных вычислительных систем

Рассмотрим основные виды вычислительных машин и этапы развития информационных технологий. По способам представления информации вычислительные машины подразделяются [1, 2, 6, 140] на аналоговые (АВМ) [140], цифровые (ЦВМ) [91] и аналого-цифровые (АЦВМ) [53, 111, 129, 141]. При аналоговом представлении результату соответствует значение сигнала, погрешность определяется компонентами и уровнем помех и возрастает в процессе вычислений, а также при хранении и передаче информации в аналоговой форме по линиям связи. Поэтому при аналоговом представлении все операции выполняются одновременно и ограничены в пространстве, время решения задачи не зависит от ее сложности. Однако количество аппаратуры и стоимость пропорциональны сложности задачи.

При цифровом представлении результату соответствует состояние множества элементов; каждый элемент может находиться в различных, устойчивых к помехам, состояниях. Элементы с двумя устойчивыми состояниями обеспечивают наибольшую помехоустойчивость и малую энергию переключения [35, 36, 125, 142, 144, 148]. Величина порогового сигнала и напряжение источника энергии определяются уровнем помех. В цифровых вычислительных машинах реализуются элементы выполнения логических операций и хранение информации. Однако погрешность выполнения вычислений и хранения данных снижается с ростом количества разрядных элементов. Передача информации в цифровой форме не приводит к увеличению погрешности. Поэтому в ЦВМ производится последовательное выполнение операций над хранимыми данными. Однако одновременное выполнение операций над потоком данных производится только в конвейерных ЦВМ. Цифровая форма представления информации позволила создать развивающуюся глобальную сеть вычислительных машин и центров сетевых сервисов (ЦСС) или центров обработки данных (ЦОД) [4, 5, 157–159].

На первом этапе из-за отсутствия общих коммуникационных подсистем группы пользователей работали на одной машине. Система виртуаль-

ных машин [5, 28–33, 86, 92] позволила обеспечить индивидуальную среду на рабочем месте за счет специализированной коммуникационной подсистемы с возможностью обмена сообщениями. Специализированные сетевые услуги стали доступны с появлением комплексов обработки и передачи данных на базе системы виртуальных машин. Комплексы, связанные сетью передачи данных, позволяли сотрудникам получать специализированные вычислительные и образовательные услуги [86].

Персональные вычислительные машины с локальными коммуникационными подсистемами позволили на рабочем месте устанавливать программное обеспечение и решать требуемые задачи. Эффективность персональных вычислительных систем ограничивалась квалификацией пользователей, большим расходом энергии и высокой ценой владения.

Сочетание персональных вычислительных машин с глобальными коммуникационными системами [91, 100, 107, 157] и сервисными центрами позволило поднять качество информационных систем. На рабочем месте устанавливается программное обеспечение для решения основных задач, а дополнительные задачи и поиск информации выполняются в виде сетевых услуг. Цена владения сократилась при повышении качества, но для управления необходим квалифицированный специалист. Рост количества и сложности вычислительных и коммуникационных систем во всех сферах деятельности стал ограничиваться количеством и квалификацией персонала.

Персональные вычислительные средства подразделяются на стационарные и мобильные [4]. Стационарные находятся на рабочих местах и имеют проводное соединение с сетью, их быстродействие и потребляемая мощность постоянно возрастают, они необходимы разработчикам и профессиональным пользователям. Несмотря на эффективность стационарных терминальных машин (тонких клиентов), они не получили распространения.

Мобильные персональные средства представлены компьютерами переносными – МПК (англ. MPC), планшетными – ППК (англ. TPC), телефонами и коммуникаторами с беспроводным соединением с сетью передачи данных. Повышение степени интеграции основных компонент обработки и передачи данных позволило создать мобильные интернет-устройства (MID) и ультрамобильные персональные компьютеры – УМПК (англ. UMPC), например Computer Card фирмы Intel. Мобильные сетевые устройства и компьютеры программно совместимы с персональными компьютерами, а размеры монитора – среднее между мониторами планшетных компьютеров и коммуникаторов. Мобильные средства оптимизированы не только по стоимости единицы производительности, но и по расходу энергии. Одним из основных параметров является время работы от автономного источника

питания. Для мобильных устройств разработаны специализированные системы на кристалле (SOC). На кристалле размещены аналого-цифровые и цифровые компоненты и датчики, соединения которых программируются и могут изменяться динамически (PSOC). Примерами PSOC являются кристаллы D1000, D2000 фирмы INTEL. Мобильные пользователи и доступные серверы образуют динамическую вычислительно-коммуникационную систему. Ограниченные мобильные вычислительные и энергетические ресурсы стимулируют использование сетевых сервисов, которые соответствуют стандартам и не зависят от аппаратной или программной платформы [4]. Сетевые сервисы выполняются на дата-центрах, объединяют вычислительные ресурсы, средства хранения, бесперебойное электроснабжение и охлаждение. Виртуализация позволяет повысить эффективность дата-центров [4, 166, 168, 184]. Основными являются качественные параметры и надежность оказываемых услуг.

Персональные и мобильные средства обработки и передачи данных позволяют сегодня получать сетевые образовательные и вычислительные услуги в любое время на рабочем месте и дома [4].

1.2. Уровни абстракции вычислительных систем

Проектные решения являются моделями объектов на различных уровнях абстракции. Начальные этапы проектирования [100–105] отличаются высоким уровнем абстракции, который снижается с конкретизацией структуры и параметров объектов.

Известны следующие этапы проектирования и сопровождения жизненного цикла объектов: цель создания и оценки эффективности, эскизного, функционального, конструкторского, технологического, испытаний, сопровождения и утилизации объектов [28–33, 103, 104].

Разным этапам проектирования соответствуют различные описания объектов и критерии их эффективности. Отличают содержание и синтаксис описаний. Содержание описаний определяется предметной областью и этапом проектирования. Разнообразие описаний объясняется их синтаксисом, а не содержанием.

На верхнем уровне абстракции (рис. 1.1) рассматривают функциональные модели систем в целом, которые называются макромоделями и описываются функциями выходов и переходов. В различных САПР функциональные модели систем представляются в различных формах и на различных языках.

На следующем уровне абстракции находятся структурные модели, в которых отражается внутренняя структура компонентов. Такие модели будем называть микромоделями. На этом уровне можно грубо оценить ресурсы: массу и габариты, статическую мощность и энергию переключения, тепловые характеристики, внешнюю среду и стоимость [1, 2, 6, 9, 11, 27–33, 87, 101].

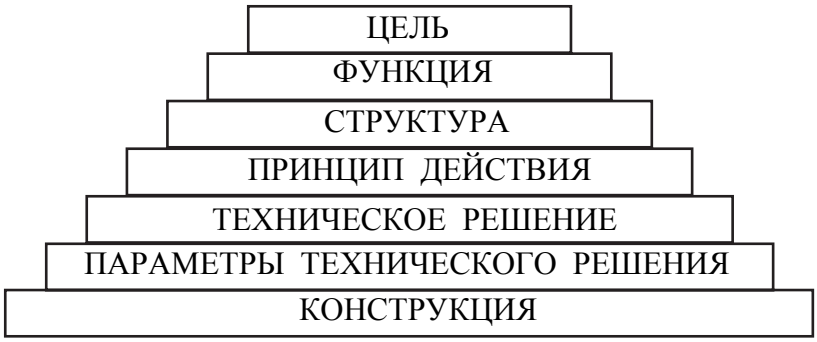


Рис. 1.1. Иерархия технических решений
на различных уровнях абстракции

Одна и та же структура может быть реализована с помощью компонентов с различными принципами действия. Возможно сравнение вариантов из компонентов с различными принципами действия. Обрабатывающая подсистема может быть реализована на одних физических принципах, а коммуникационная – на других. Физические принципы действия элементов ЭВМ подробно описаны в [1, 6, 8, 9, 16, 19–21] и будут рассмотрены только при оценке ресурсов или оптимизации [95, 96, 124, 125].

Зная принцип действия, с учетом уровня техники и технологии, можно перейти к уровню технического решения. Техническое решение должно быть выполнимым. После получения технического решения возможно изменение параметров решения с целью повышения качества изделия. Такие примеры рассмотрены в работах автора по параметрической оптимизации [12–15, 34–38].

Определив параметры технического решения, переходят к конструкции системы. Например, в зависимости от рассеиваемой кристаллом мощности ему будет назначен соответствующий корпус. На конструкцию существенное влияние оказывает внешняя среда (условия эксплуатации). Таким образом, на верхнем уровне абстракции находятся функциональные модели компонентов, а на нижнем уровне – конструкции компонентов и их геометрия. Геометрические модели компонент используются при конструировании [101].

Проектные решения вычислительных устройств и систем на разных этапах проектирования представляют собой множество компонентов и отношений. Отношения могут быть типа принадлежности (соответствие пе-

речню элементов, входящих в устройство) или передачи информации по каналам. Отношения различаются содержанием и формой представления. Основным является содержание отношений, а форма представления допускает множество реализаций, ограниченных отраслевыми стандартами и конкретными САПР. Международные стандарты ISO 10303 и EDIF, PDIF определяют протокол и структуры обмена данными между различными САПР [128, 130, 133, 134].

Различают задачи параметрической [95, 96, 112, 124, 149] и структурной оптимизации [3, 17, 27, 28, 60–63, 71, 83, 88, 102–105, 110]. При параметрической оптимизации оцениваются управляемые параметры, соответствующие экстремальной области значений выходного параметра [15–19, 34–38]. При структурной оптимизации синтезируется множество структур, из которых выбираются лучшие по критерию качества [103]. Структуры отличаются компонентами или отношениями компонентов. Для вычислительных систем отношение может представляться общим каналом передачи данных в виде соединения для стационарных компонентов и возможностью обмена сообщениями для мобильных компонентов.

В табл. 1.1 приведены основные уровни абстракции неоднородных вычислительных систем (НВС) и критерии их эффективности. Модели для каждого уровня независимы, однако при переходе к нанотехнологиям [59] уровни логический, физический и пространственный вычислительных систем зависимы и влияют на результат, поэтому необходим ряд итераций с учетом зависимых уровней.

На системном комплексном уровне (7) оценивается эффективность преобразования энергии или вещества при наличии структуры, обрабатывающей информацию, или без нее. На уровне конечного результата (6) оценивается совершенство структуры обработки информации. На системном информационном уровне (5) оценивается совершенство системы преобразования обработки информации при абстракции ее внутренней структуры. Структурный уровень (4) оценивается по числу тактов на обработку заявки на элемент.

Для логического уровня (3) характерны абстракции пространственного размещения компонентов и их физической реализации. Однако при согласовании вычислительной системы (ВС) с внешними каналами необходима конкретизация физической реализации компонентов. На физическом уровне (2) оценивается энергия изменения состояния для двоичного компонента – энергия переключения, которая определяет рассеиваемую объектом мощность для конкретной частоты и ограничивает степень интеграции компонентов [69, 84, 99, 125, 142]. На пространственном уровне (1) размещения элементов стремятся к максимальной объемной или поверхностной плотности компонентов, ограниченной технологическими нормами

и допустимой рассеиваемой мощностью. Пространственное размещение учитывается только при конструкторском проектировании объектов со стационарными компонентами [6, 120–123]. Уровень неоднородной физической среды (0) обеспечивает устойчивую обработку информации при возможных внешних воздействиях.

Таблица 1.1

Уровни абстракции НВС

Номер уровня	Абстракции	Уровень	Наименование критериев эффективности
7	Способа обработки информации	Системный комплексный	Совершенство системы преобразования энергии или вещества в нестационарной внешней среде
6	Структуры системы обработки информации	Конечного результата	Эффективность преобразования энергии или вещества при обработке информации
5	Внутренней структуры	Системный информационный	Совершенство системы преобразования информации
4	Логической реализации преобразователей информации	Структурный	Совершенство структуры – число тактов на обработку заявки на элемент
3	Физической реализуемости	Логический	Совершенство логических преобразователей информации
2	Пространственного размещения	Физический	Работа переключения полная
1	Процессов в неоднородной среде	Пространственный	Объемная (поверхностная) плотность
0	Нет	Неоднородной физической среды	Допустимые условия работы

Появление компонентов со средней и большой степенью интеграции привело к необходимости создания моделей ВС, неоднородных по уровням абстракции. Например, в модели ВС на базе микроЭВМ для обработки сигналов внешние цифровые или аналоговые сигналы должны представляться в многозначном алфавите. Аналогичен и уровень представления внутренних обрабатывающих компонентов, вплоть до интерфейсных узлов, выполняющих роль преобразователей уровней абстракции представления сигналов [33, 118, 138]. Если протоколы преобразований сигналов внутри микроЭВМ стандартны, то нет смысла в их детальном анализе и можно перейти к более высокому уровню абстракции. Поэтому необходимы модели ВС, неоднородные по уровням абстракции.

Процесс проектирования и испытаний нового объекта является итерационным (рис. 1.1). В каждом итерационном цикле выполняются про-

ектные процедуры синтеза, анализа и принятия решения. Процедура анализа решения для известных компонентов осуществляется автоматически, а синтез и принятие решения реализуются инженером или студентом.

Результатом синтеза является описание объекта, результатом анализа – оценка характеристик и диаграмма поведения объекта при определенных внешних воздействиях.

Для простых объектов анализ поведения проще его описания, однако с увеличением сложности объектов анализ поведения становится очень трудной задачей по сравнению с его описанием. Поэтому автоматизация анализа должна опережать автоматизацию синтеза вычислительных систем.

Управление итерационным процессом осуществляется с целью получения описания объекта, характеристики и поведение которого удовлетворяют заданию. Управлять можно описанием объекта, получаемым в результате синтеза, и внешними воздействиями на объект.

Целью создания системы или устройства являются конечные результаты, получаемые после реализации объектов, которые должны выполнять функции в заданных интервалах изменения дестабилизирующих факторов. Формированием технического задания завершается этап внешнего проектирования [48, 60–63].

1.3. Развитие структур неоднородных вычислительных систем

При выполнении работ по автоматизации испытаний и технологических процессов нужно было найти оптимальное распределение в пространстве программируемых устройств сопряжения с объектами [42–44]. Аналитическая стоимостная модель позволила найти решение.

МикроЭВМ позволяют реализовать многофункциональные активные устройства сопряжения с объектами (УСО). Накопление знаний и моделей процессов и сигналов предполагает ввод отклонений сигналов от предполагаемых и задание границ интервалов работы по определенному алгоритму [17, 29, 72, 84, 85, 155, 156].

Приближение активных УСО к объектам позволяет снизить уровень помех на входе устройств и увеличить информационную производительность всей системы. Однако распределение УСО в пространстве приводит к увеличению аппаратных затрат. Оптимальное распределение УСО в пространстве определяется уровнем развития элементной базы, от микроЭВМ с аналого-цифровыми устройствами до систем на кристалле (SOC) [42–44, 58].

Задача оптимального распределения в пространстве УСО на базе микроЭВМ решена [42–44] с использованием относительной стоимостной модели, обеспечивающей приемлемую достоверность результатов. Для времени физического и морального старения аппаратуры TAM и времени существования структуры подсистемы TAS (оценка среднего интервала времени между перестройками) получена оценка оптимального числа активных УСО на базе микроЭВМ NPUCO и оптимальное число сигналов для УСО – NCO. Информативным параметром является полная относительная стоимость УСО за все время существования системы:

$$KCUS = 2((TAS/TAM)\cdot CKUCO + TAS\cdot CTUCO)/(CPLA\cdot(LA + LB)), \quad (1.1)$$

где CKUCO, CTUCO – капитальные и эксплуатационные затраты на УСО на базе микроЭВМ; CPLA – погонная стоимость аналоговых магистралей; LA, LB – геометрические размеры объекта.

Зависимости NPUCO и NCO от числа сигналов NCS, параметра KCUS и относительной стоимости цифровых магистралей с учетом характера связей УСО со следующим уровнем иерархии (коэффициент КС) определяется выражениями

$$NPUCO = \sqrt{\frac{NCS}{\frac{CPLD}{CPLA\cdot KC} + KCUS}} \quad (1.2)$$

и

$$NCO = \sqrt{NCS(\frac{CPLD}{CPLA\cdot KC} + KCUS)}\ , \quad (1.3)$$

где CPLD – погонная стоимость цифровых магистралей.

Результаты расчетов по формулам (1.2) и (1.3) приведены в интерактивном графике в [28, 29]. Таким образом, со снижением стоимости УСО на базе микроЭВМ следует увеличивать степень их распределения в пространстве. Ограничивающим фактором является относительная стоимость цифровых магистралей. Уменьшение среднего интервала между перестройками системы увеличивает степень распределения УСО в пространстве. Распределение подсистемы преобразования и обработки сигналов на базе микроЭВМ экономически эффективно несмотря на увеличение стоимости аппаратуры, что подтверждается результатами разработок и внедрения таких подсистем. В ближайшее время они будут заменены проводными или беспроводными системами на кристалле (SOC) [58, 163, 168].

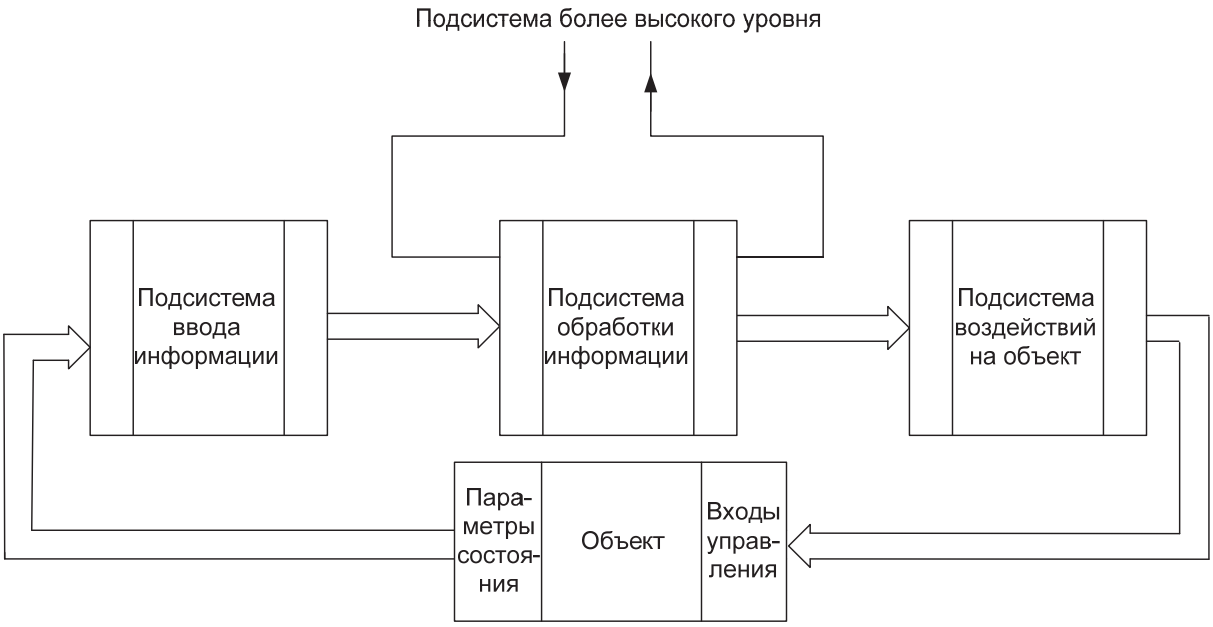


Рис. 1.2. Структура активного устройства сопряжения с объектом

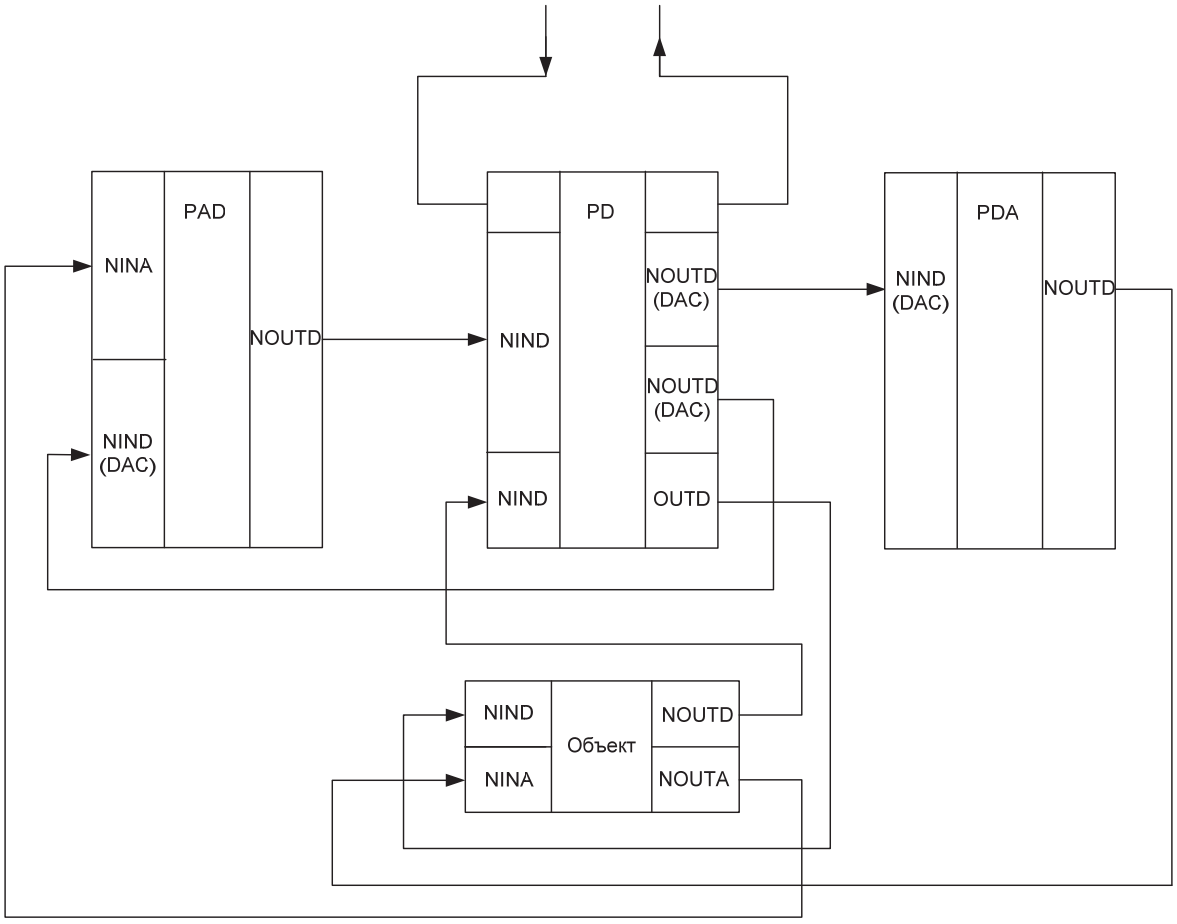


Рис. 1.3. Структура активного устройства сопряжения с объектом с цифроаналоговой и аналого-цифровой подсистемами: PD – процессор цифровой, PAD – процессор аналого-цифровой, PDA процессор цифроаналоговый

На рис. 1.2 приведена структура активного устройства сопряжения с объектом [2, 3, 28, 29], множество которых распределено в пространстве и работает одновременно. На рис. 1.3 показана обобщенная структура активного устройства сопряжения с объектом с подсистемами цифровых и аналоговых сигналов [2, 3, 28, 29]. Выходная подсистема (PDA) преобразует группы цифровых сигналов в аналоговые и состоит из двух частей. Первая подается на объект и мало зависит от алгоритма работы цифровой подсистемы. Вторая часть подается на входную подсистему преобразования аналоговых сигналов в цифровые (PAD) и по мере накопления знаний об объекте соответствует предполагаемым аналоговым сигналам. Цифровой процессор (PD) может содержать необходимое количество ядер в зависимости от сложности модели объекта. Во второй и последующих главах рассматриваются конкретные реализации структур на рис. 1.2 и 1.3.

1.4. Комплексы моделирования и проектирования неоднородных вычислительных систем

Одним из первых комплексов проектирования вычислительных машин была КАСПИ [9], созданная под руководством С. А. Лебедева, Б. А. Бабаян и Г. Г. Рябова. В КАСПИ была реализована подсистема моделирования логических схем, уровни физического и пространственного размещения компонент, трассировки и подготовки данных для технологического оборудования.

В 1970–1990 годах Министерство образования координировало и поддерживало разработку исследовательских программно-методических комплексов моделирования и проектирования вычислительных и испытательных систем [29, 101, 102, 135, 137–139]. Основная часть работ относилась к логическому и схемотехническому уровням со специализированными языками описания проекта подобно известной зарубежной системе моделирования SPICE [103]. Основные программно-методические комплексы созданы под руководством И. П. Норенкова [102, 135], Е. И. Артамонова [2, 3], В. И. Анисимова [139], В. М. Дмитриева в Томске [7], В. В. Топоркова [137, 138] и А. К. Полякова [114, 115]. Министерство радиопромышленности разрабатывало комплексы «ПРАМ» («РАПИРА») под руководством Ю. Х. Вермишева [86, 128], из которых нами использовались ПРАМ53 совместно с ПРАМ2, ПРАМ53М/РС97, ГРИФ-4 [77–79] (развитие PCAD-200X в НПО АЛМАЗ Ю. М. Елшиным).

В процессе развития вычислительных систем и повышения степени интеграции микросхем комплексы моделирования и проектирования стали специализированными для разработчиков микросхем [1, 132] и системных

интеграторов [8, 103, 114, 115]. Комплексы для системных интеграторов, которым уделим основное внимание, наряду с логическим уровнем с течением времени должны были решать задачи структурного и системного уровня. Если в существующих комплексах основной ввод информации является графическим и не требует знания языков программирования, то в перспективе основной ввод информации будет текстовым или интерактивным, на универсальных языках описания систем с дополнительными библиотеками.

Современные комплексы моделирования и проектирования представлены коммерческими и свободными продуктами [58, 116]. Для систем на кристалле основной задачей является создание «исполняемой системной модели», соответствующей формализованному заданию, введенному в практику В. М. Глушковым. Рассмотрим комплекс «Visual Elite» фирмы Mentor Graphics и «Riviera Pro» фирмы Aldec. Основным языком описания проекта является принятый в 2005 году стандарт IEEE std. 1666-2005 SystemC [183]. Предлагается использование стандартного языка C (C++) с ограничениями и дополнительными библиотеками. Наряду со стандартом разрешается использование фрагментов VHDL (подмножество языка ADA для описания цифровых систем) и ввод графических фрагментов. Для макропрограммирования в САПР принят язык NETScriptCAD.

Использование графических фрагментов не позволяет описать в одном формализованном задании множество вариантов проекта и перейти на второй уровень сложности задач проектирования [103]. Сложность комплексов с описанием на одном основном языке на одной платформе не исключает ошибки системы при высокой стоимости продукта.

Свободные САПР можно отнести к двум группам. Первая группа позволяет отказаться от коммерческих продуктов схемотехнического и конструкторского проектирования (Kicad, gEDA, FreePCB, Electric) и подходит для технического проектирования одного или двух вариантов объекта. Вторая группа свободных продуктов разрабатывается международным консорциумом на базе стандартов, благодаря интернет-технологии, как расширение международного инструментального проекта Eclipse [164].

Таким образом, существующие комплексы моделирования и проектирования не позволяют перейти к решению задач проектирования второго уровня сложности при повышении или сохранении производительности труда. Комплексы на одной платформе с описанием на одном основном языке не исключают ошибок системы. Отсутствует автоматическое преобразование вариантов формализованного задания в принципиальную схему и трехмерную модель с отображением сигналов на ранних этапах проектирования. Отсутствует абстрактный тип компонентов.

Показана необходимость решения актуальной проблемы создания открытого модульного многоуровневого программно-методического комплекса многовариантного моделирования и проектирования неоднородных вычислительных систем с вводом формализованного задания множества вариантов и автоматическим формированием диаграмм, схем, трехмерного облика объекта и оценкой его основных параметров в различных синтаксических средах и на различных платформах.

Нужно повысить производительность труда при многовариантном проектировании и в процессе обучения за счет автоматизации формирования моделей и документов [27–33].

Автором доказана тенденция оптимального распределения аналоговой и цифровой обработки данных в пространстве путем приближения средств аналого-цифровой обработки к источникам и потребителям сигналов в процессе развития вычислительных систем. В пределе при переходе к системам на кристалле (SOC) аналоговые сигналы обрабатываются непосредственно у датчика, и передача данных производится по последовательной цифровой сети. При производстве больших партий используются специализированные SOC, содержащие даже такие датчики, как DS18B20 фирмы Dallas, а позже фирмы Maxim. Для малых партий эффективны PSOC, содержащие цифровую и аналоговую подсистемы, входные и выходные коммуникационные подсистемы, выпускаемые фирмой Cypress Semiconductor совместно со средой программирования PSoC Development Tools.

С 2018 года мы используем систему на кристалле Intel D2000 в виде отладочного модуля совместно с Intel System studio.

2. ПРИНЦИПЫ ПОСТРОЕНИЯ МНОГОУРОВНЕВЫХ СИСТЕМ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

2.1. Описание проектных решений вычислительных систем

Проектные решения являются моделями объектов на различных уровнях абстракции. Начальные этапы проектирования [1, 9–12, 89, 93, 95, 103] отличаются высоким уровнем абстракции, который снижается с конкретизацией структуры и параметров объектов.

Сначала объект представляется функциональной макромоделью, затем микромоделью из абстрактных функциональных компонент. Они представляются микромоделями из абстрактных или конкретных функциональных компонент. Процесс продолжается до уровня конкретных функциональных компонент, для которых возможна оценка ресурсов создания объектов.

Задачи проектирования И. П. Норенков различает по уровням сложности [102, 103]. К первому уровню сложности относятся задачи, в которых технические решения заданы и синтез отсутствует. Второй уровень сложности отличается множеством возможных технических решений при наличии ресурсов для анализа, оценки и выбора лучшего варианта по критерию эффективности. К третьему уровню сложности [103] относят задачи с множеством возможных технических решений при дефиците ресурсов для анализа и оценки всех возможных вариантов, так что часть вариантов приходится исключать без детального анализа. Таким образом, задачи третьего уровня сложности преобразуются ко второму.

Автоматизированное проектирование занимает промежуточное положение относительно традиционного (ручного) и автоматического проектирования.

Государственным стандартом Р ИСО 10303, введенным 01.07.2000, при проектировании ВС предусмотрено формирование информации об изделии для ее дополнения в процессе производства, эксплуатации, технического обслуживания и утилизации [29, 33, 79, 104, 133]. Информация может быть использована в вычислительных системах в различных организациях. Описания изделий должны быть полными и совместимыми.

Описание вычислительных систем состоит из моделей компонент и соединений между ними. Оно может быть представлено различным образом. При описании системы в виде формализованного задания достаточно разделов «Компоненты» и «Соединения».

Вычислительная система как объект проектирования представляется множеством взаимодействующих компонентов, каждый из которых можно рассмотреть как составной объект. Модель абстрактного объекта представляется структурой данных, значение которой отражает его состояние, методами или операциями, преобразующими состояния объектов в соответствии с воспринимаемыми входными сообщениями. Модель объекта формирует выходные сообщения или значения параметров процедур для других объектов. Важным для восприятия человеком является форма представления объектов и учет исключительных ситуаций. Известны символьная, табличная и графическая формы представления. Поведение и взаимодействие объектов воспринимается лучше в графической форме (временные диаграммы и принципиальные схемы), а оценка ресурсов и степень соответствия предполагаемых и фактических результатов – в табличной форме.

Таковы концепции описания и представления предметной области неоднородных вычислительных систем. Формальное представление предметной области (ПРО) возможно с помощью онтологий [10, 70]. Модель онтологий представляется как

$$O = (T, TREL, FVIEW),$$

где T – конечное множество понятий (концептов, терминов); TREL – конечное множество отношений между концептами; FVIEW – конечное множество функций интерпретации, заданных на понятиях или отношениях.

Основные понятия в области информационных технологий и автоматизированных систем приведены в ГОСТ 34.003–90. Понятие функциональных компонентов и соответствующие им символы на принципиальных схемах определены в стандарте ЕСКД ГОСТ 2.743–91, а понятие «модель изделия» – в ГОСТ 2.052–2006. Отношения между функциональными компонентами (TREL) составляют вычислительный модуль, который может рассматриваться как часть формализованного задания (FT, FZ), раздел описания модуля (UNIT) и представляться (FVIEW) принципиальной схемой или объемной моделью [29, 32, 33].

Использование онтологий при проектировании и сопровождении объектов рекомендуется российским стандартом Р ИСО 15926, принятым в 2008 году. В связи с большим числом конкретных компонент и терминов используется многоуровневая модель с небольшим числом базовых понятий на верхнем уровне. По мере уточнения условий работы объекта конкретизируются компоненты и увеличивается их разнообразие. Многоуровневое

представление позволяет выровнять сложность принятия решений на различных уровнях.

Международные стандарты ISO 10303 определяют протокол и структуры обмена данными между различными САПР [103–105, 115, 133] и системами сопровождения объектов. Примером стандарта языка для описания цифровых систем (VHDL) [8] является документ Института инженеров по электротехнике и радиоэлектронике (IEEE) № 1076. Язык VHDL создан на базе универсального алгоритмического языка ADA. В новой редакции VHDL STD 1076.1–1999 (VHDL–AMS) введена возможность описания аналоговых сигналов [132]. В 2004–2008 гг. приняты стандарты на язык проектирования системного уровня IEEE STD 1666–2005 и его развитие. Отсутствие стандарта приводит к несовместимости представлений проектных решений в различных САПР при идентичном содержании [1, 31, 84, 104, 128].

Переход на более низкий уровень абстракции должен сопровождаться конкретизацией имеющихся описаний объектов и введением новой информации в форме отношений. Желателен однократный ввод информации тех разделов описаний, которые не изменяются на последующих этапах. Раздел описания, как правило, состоит из нескольких отношений.

Конкретизации подлежит неопределенность структуры или параметров системы [55–58, 65, 67, 90, 97, 155], которая велика на начальных этапах проектирования. Предпроектные исследования проводятся, когда известна только цель создания системы, а их результатом является техническое задание (ТЗ). Основную часть ТЗ составляет множество отношений в форме ограничений, например: производительность системы на комплексе задач не менее заданной, стоимость системы не более установленной или предельное значение комплексного критерия – отношение стоимости к производительности.

Наличие ТЗ позволяет составлять и анализировать технические решения, множество которых можно представить в форме дерева [87, 104, 112] из вершин типа И и ИЛИ. Вершины типа И определяют необходимый состав объекта, а вершины ИЛИ – неопределенные компоненты, подлежащие конкретизации путем анализа согласованности свойств компонента и объекта или анализа работы устройства.

Результатом структурного проектирования являются состав компонентов и отношения между ними. На этапе технического проектирования состав компонентов и соединения должны быть полностью конкретизированы. На рис. 2.1 приведена схема алгоритма проектирования структуры объектов.

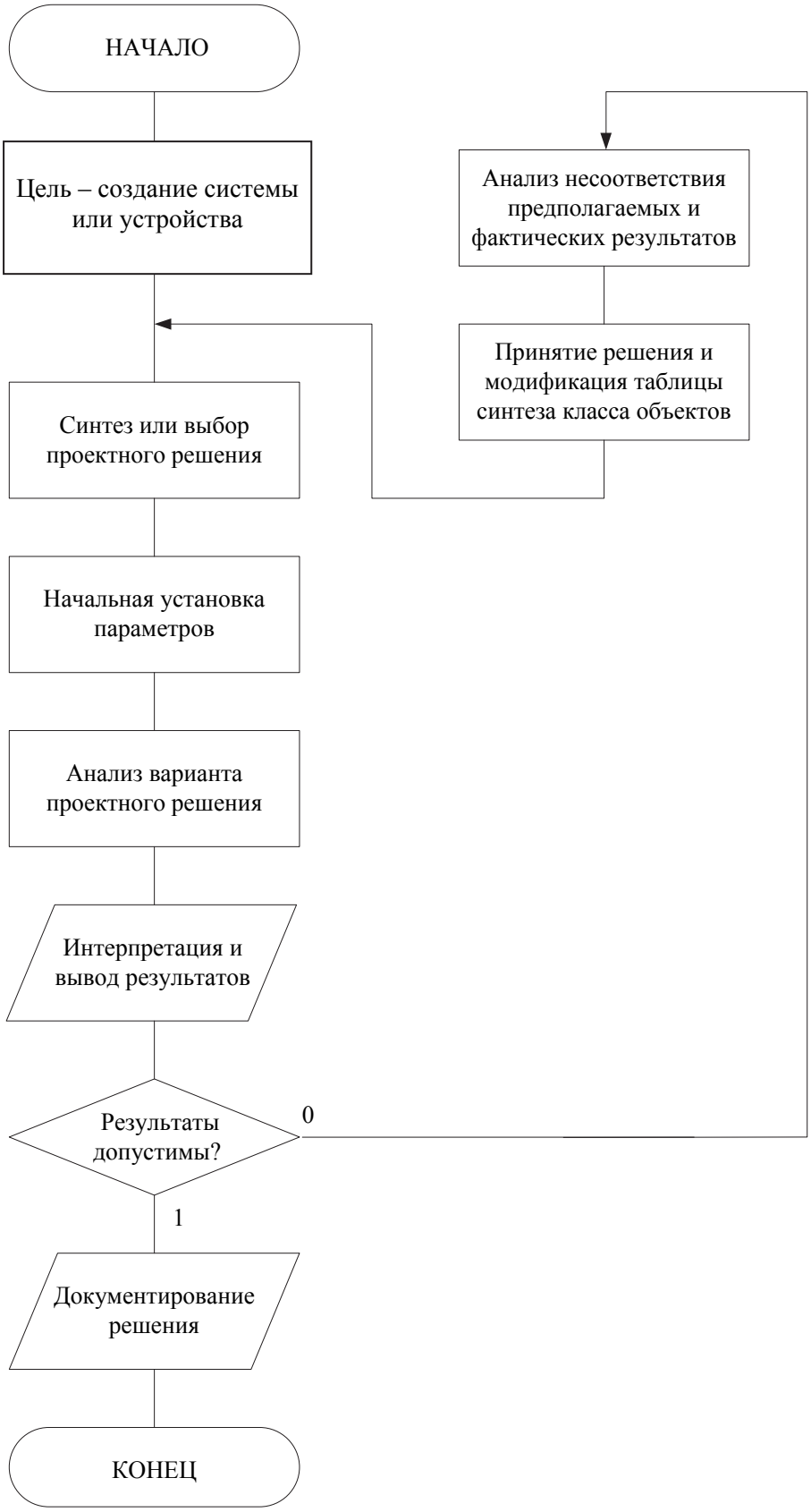


Рис. 2.1. Блок-схема алгоритма автоматизированного проектирования класса объектов

Разнообразие правил описания объектов в различных САПР порождает множество языков проектирования. Различны и способы ввода описаний: от набора текстов до интерактивных систем с графической интерпретацией описаний и результатов анализа.

Внешнее проектирование объекта включает построение макромодели и ограничение ресурсов. Внутреннее проектирование включает создание вариантов микромодели с различной степенью конкретизации компонентов. На нижнем уровне для конкретных компонент возможна оценка ресурсов и сравнение различных вариантов объекта. Оценку эффективности производят только для моделей объектов, у которых выходные сигналы соответствуют предполагаемым.

Особенностью вычислительных систем является выбор принципа действия для детальных функциональных схем. Принцип действия выбирается в зависимости от условий работы и внешней среды.

Процесс получения детальной микромодели объекта получил название «маршрут проектирования». В отличие от алгоритма маршрут проектирования может не выполниться за конечное число шагов. Проектирование объекта реализуют итерационным методом с обучением на каждой итерации. Творческие операции синтеза и интерпретации результатов анализа производит инженер, а отдельные простые операции и трудоемкие процедуры анализа осуществляются автоматически при наличии формализованного задания (ФЗ). Для автоматизации последующих этапов проектирования вычислительных устройств и систем результаты структурного синтеза должны быть представлены в виде формализованного задания.

Пространственное размещение структурных компонентов и соединений производится на конструкторском этапе проектирования. Модели геометрии компонентов хранятся в общей или локальной базе данных.

Многоуровневая САПР COD (Conceptual Object Design) служит для синтеза и анализа множества вариантов структур и их автоматического преобразования в множество проектных решений для промышленных САПР [12, 27, 32, 33].

Для принятия решения и синтеза объектов в программно-методическом комплексе COD из многовариантного объектно-компонентного формализованного описания объектов (ФЗ) формируются временные диаграммы с автоматическим сравнением предполагаемых и фактических сигналов, таблицы параметров и критериев оптимальности, принципиальные схемы и образы объектов для всех вариантов. На конструктивах компонентов могут отображаться цифровые сигналы и температура. Для перехода к техническому проектированию ФЗ преобразуется в формат конкретной системы проектирования.

Анализ несоответствия предполагаемых и фактических результатов позволяет принять решение о модификации правил синтеза класса объектов. Правила синтеза представляются таблицей решений и могут описываться в формализованном задании с помощью условных операторов или условий выбора.

Производительность определяется с помощью тестов: по выполненной полезной работе в единицу времени или по изменению цифрового или аналогового сигнала за заданный интервал времени [32, 33, 132, 148]. При оценке производительности задается номер изменяющегося цифрового (nppd) или аналогового (nppa) сигнала.

Минимальный объем описания объекта представляется в символьной форме, а максимальный – в графической. Поэтому минимальной трудоемкостью отличаются ввод символьных описаний и автоматическое преобразование описаний в формы, удобные для восприятия человеком.

Описание структуры объекта относят к результатам внутреннего проектирования, которое может выполняться различными методами. Если проектное решение, соответствующее требованиям задания, неизвестно, то выполняется основная процедура синтеза описания нового объекта. При наличии требуемого или близкого к нему проектного решения либо множества решений производят его выбор и путем внесения изменений в структуру объекта добиваются удовлетворения требований технического задания. Способы синтеза решений приведены на рис. 2.2. Процедуры экспертизы технических решений на новизну требуют выявления аналогичных решений и доказательства существенности отличий, что подтверждает необходимость выбранного направления дифференциального синтеза и анализа.

При описании множества решений на языке низкого уровня количество описаний равно мощности множества решений, а при описании на языках высокого уровня в одном описании может быть множество вариантов решений. Рациональным является описание на языке высокого уровня множества решений для одного класса вычислительных систем или устройств.

Для структурной оптимизации необходимы критерии эффективности [33, 90, 93, 95, 96, 124, 125]. Для стационарных вычислительных систем критерием эффективности является стоимость единицы производительности. Для мобильных объектов таким критерием может быть масса единицы производительности или масса вычислительной системы и источника энергии на единицу производительности [29–33]. Подобные оценки для множества решений представляют сложную и трудоемкую задачу, для реального решения которой требуются инструментальные средства и информационное обеспечение. Часто используется ограниченное количество критериев, выражения для которых сведены в табл. 2.1.

Неоднородность вычислительных систем связана с различными носителями и формами представления информации множества источников, приёмников и разной структурной организацией обрабатывающих подсистем с большим диапазоном производительности [2].

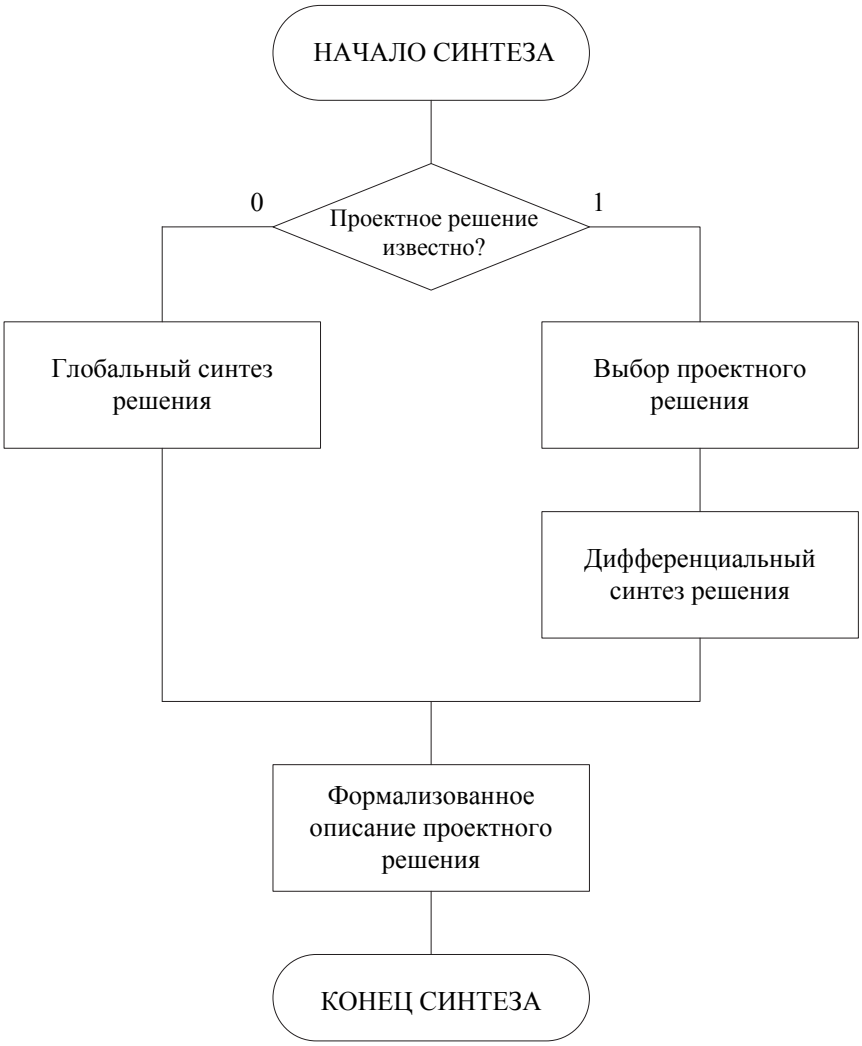


Рис. 2.2. Блок-схема синтеза проектного решения

Таблица 2.1

Основные критерии оптимальности вычислительных систем

Критерий	Размерность	Выражение	Область применения
Стоимость единицы производительности	Относительная стоимость / MIPS	$K = C / P$	Вычислительные системы общего назначения
Масса единицы производительности	г / MIPS	$K = M / P$	Бортовые вычислительные системы
Мощность и масса единицы производительности	Вт / MIPS · г / MIPS	$K = M/P \cdot PF/P$	Переносные вычислительные системы

Для гибких автоматизированных производств и рабочих мест функции вычислительных систем могут управляться событиями, что требует динамической перестройки структуры. Совместное использование средств обработки и передачи данных позволяет создавать динамически управляемые структуры информационных систем.

Условия работы вычислительной техники значительно отличаются на различных уровнях иерархии. На верхнем уровне иерархии находятся дата-центры [166, 168, 174, 184]. Дата-центры представляют собой вычислительные и коммуникационные ресурсы с подсистемой хранения информации с автономным электроснабжением, климатической системой и системой безопасности. Модульные дата-центры могут работать в различных условиях [166]. Рабочие станции и персональные серверы, как правило, используют в лабораторных условиях. Специальные условия эксплуатации на верхних уровнях иерархии, необходимые для вычислительных систем уровня предприятия, сменяются лабораторными для персональных серверов и жесткими условиями для специализированных систем непосредственно на рабочих местах и в составе технологического оборудования. Эксплуатация вычислительных систем нижнего уровня должна обеспечиваться технологическим персоналом, а не специалистами вычислительных центров на средних и верхних уровнях.

2.2. Структурный синтез вычислительных устройств и систем

Более детально методы структурного синтеза [1, 29–33, 102, 103, 132] представлены на рис. 2.3. Если технические решения (ТР) известны и достаточно ресурсов, то производится полный перебор вариантов, иначе выполняется уменьшение мощности множества решений. Для класса объектов алгоритм синтеза известен и возможен синтез в режиме диалога или описание на одном из языков высокого уровня (ЯВУ). В качестве примера можно привести синтез класса груботочных аналого-цифровых преобразователей последовательного счета [29–33]. Формализованное задание позволяет получить временные диаграммы для четырех вариантов структур и оценки эффективности для пятнадцати вариантов. Оптимальным оказался вариант с разделением счетчика на две части – грубую и точную.

Если ТР аналога известно, то выполняют дифференциальный синтез путем добавления-удаления компонента или связей с оценкой эффективности. Если же ТР неизвестно, то производят глобальный синтез.

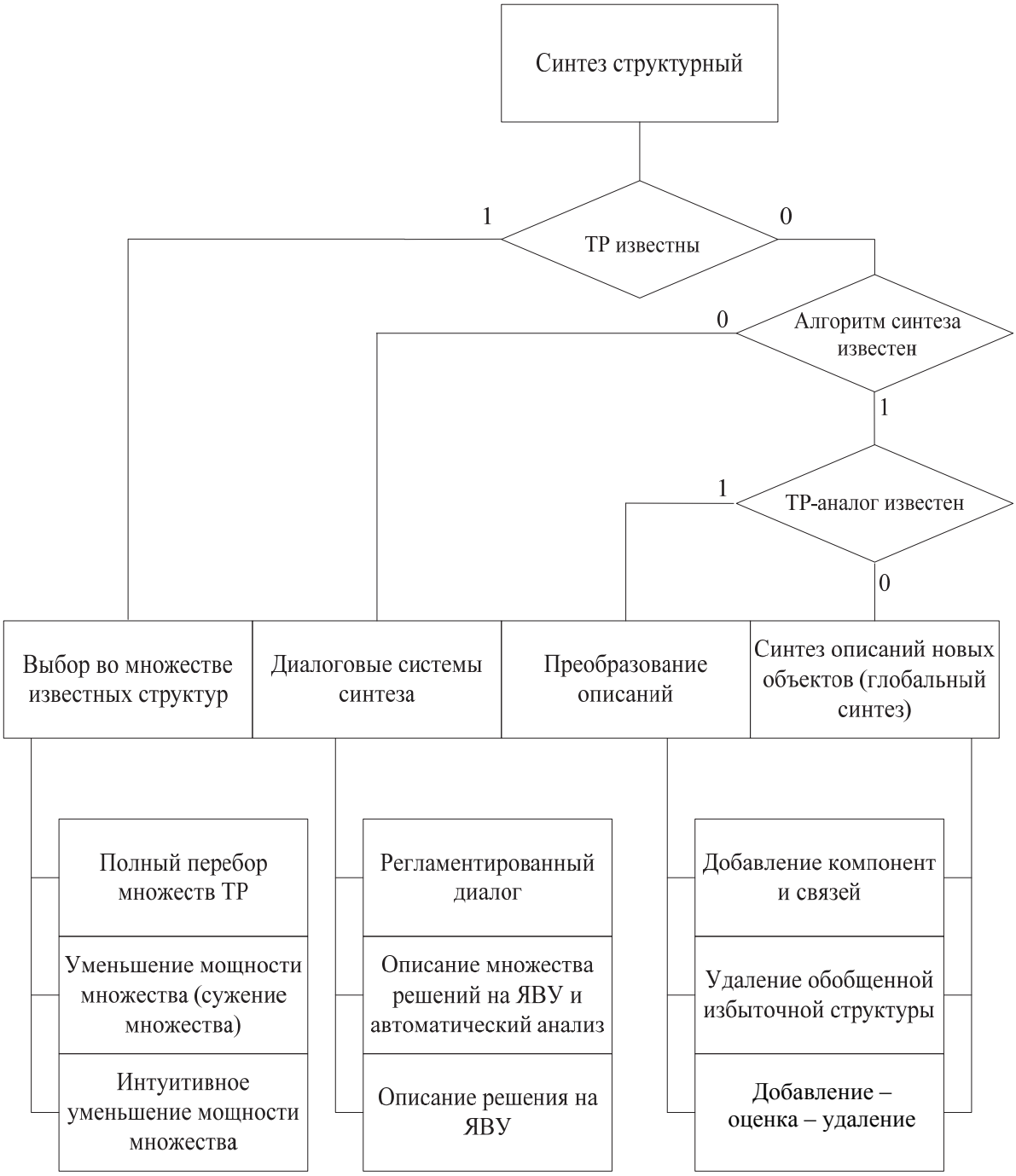


Рис. 2.3. Структурный синтез вычислительных систем

Структура системы определяется множеством компонентов и отношениями между ними. Структуры систем могут быть статическими для всех вариантов системы, статическими для одного варианта системы и динамически изменяться в пределах любого из вариантов. Описание перечисленных структур возможно на языках высокого уровня. На языках низкого уровня возможны описания только статических структур. При изменении структуры вычислительной системы в соответствии с вариантом

$$S(N + 1) = FS(S(N), RS), \qquad (2.1)$$

где $S(N + 1)$, $S(N)$ – структуры для $N + 1$ и N-го варианта системы; RS – правила изменения структур в зависимости от значений предполагаемых и фактических сигналов, результатов обработки, критериев эффективности и правил для класса структур, а также в зависимости от внешних воздействий; FS – функция формирования структуры варианта $N + 1$.

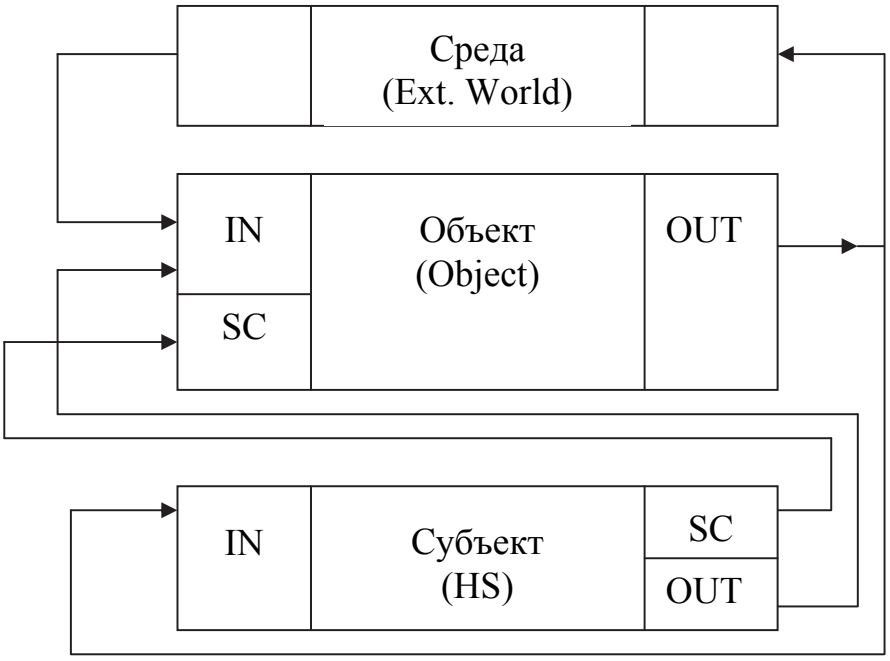


Рис. 2.4. Процедура синтеза систем с учетом внешней среды

Создаваемая система (объект) представляется конечным множеством отношений между концептами (TREL) [10], и для принятия решения об изменении варианта субъекту требуется представить в наглядной форме различные свойства объекта, для чего служат функции интерпретации (FVIEW). В результате выполнения проектной процедуры анализа получают поведение объекта и оценку ресурсов. Из формализованного задания можно создать схему в стандартной программе просмотра с отображением активности или объемный вид модуля или системы на ранней стадии про-

ектирования. Наглядность интерпретации результатов сокращает время принятия решения и формирования правил синтеза [2, 9–11, 55–63].

Внешняя среда является основным фактором при выборе принципа действия. Электронные компоненты работают в ограниченном температурном интервале и неустойчивы ко многим воздействиям. Поэтому необходимо рассмотреть принципы действия аналого-цифровых узлов, систем на базе микроЭВМ с использованием оптимальной структуры узлов в зависимости от приложений, хотя в дальнейшем можно использовать систему на кристалле или сигнальный процессор.

2.3. Аналого-цифровые устройства вычислительных систем

Основными узлами подсистем сопряжения с объектами являются аналого-цифровые преобразователи (АЦП – ADC), аналоговые мультиплексоры, узлы обработки аналоговых сигналов – аналоговые процессоры (РА) и цифроаналоговые преобразователи (ЦАП – DAC). Аналоговые и цифровые мультиплексоры – это типовые компоненты вычислительных систем. Их модели включены в библиотеку компонентов, а таблицы синтаксиса и семантики приведены в работах [6, 33].

В качестве примера аналогового процессора в библиотеку включены функциональные модели интегратора и масштабного преобразователя. Наряду с аналоговым входом и выходом, модели содержат входные цифровые управляющие сигналы и управляемый параметр. Для интегратора управляемым параметром является постоянная времени, а для масштабного преобразователя – коэффициент передачи.

Цифроаналоговый преобразователь относится к типовым компонентам [6, 31]. Функциональные модели ЦАП имеют произвольную разрядность и диапазон выходных сигналов. В табл. 3.15 приведены основные параметры преобразователей, выпускаемых различными предприятиями.

Аналого-цифровые преобразователи относятся к наиболее сложным узлам, включающим все перечисленные компоненты, и отличаются структурой, быстродействием, погрешностью работы, устойчивостью к помехам. Различают [2, 6, 33] АЦП последовательного счета, поразрядные и одного отсчета. Первые отличаются минимальным быстродействием и минимумом аппаратных затрат. Вторые занимают промежуточное положение, а преобразователи одного отсчета характеризуются наибольшими быстродействием и аппаратными затратами. Сравнительная характеристика трех типов АЦП приведена в табл. 2.2.

Таблица 2.2

Сравнительная характеристика АЦП

Параметр	АЦП последовательного счета	АЦП поразрядного уравновешивания	АЦП одного отсчета
Время выполнения преобразования (в тактах)	2^n	n	1
Относительное количество аппаратуры	$3 \cdot n$	$4 \cdot n$	$2 \cdot 2^n$
Критерий эффективности C/P	$3 \cdot 2^n \cdot DELT$	$4(n \cdot DELT)$	$2 \cdot 2^n \cdot DELT$

В табл. 2.2 n – количество разрядов, $DELT$ – длительность одного такта, C – относительное количество аппаратуры (оценка стоимости), P – производительность.

Использование языка высокого уровня для описания аналого-цифровых устройств позволяет моделировать аналого-цифровые устройства с различными типами компонентов. Например, в АЦП на рис. 2.5 в разных вариантах можно использовать отличные типы компараторов.

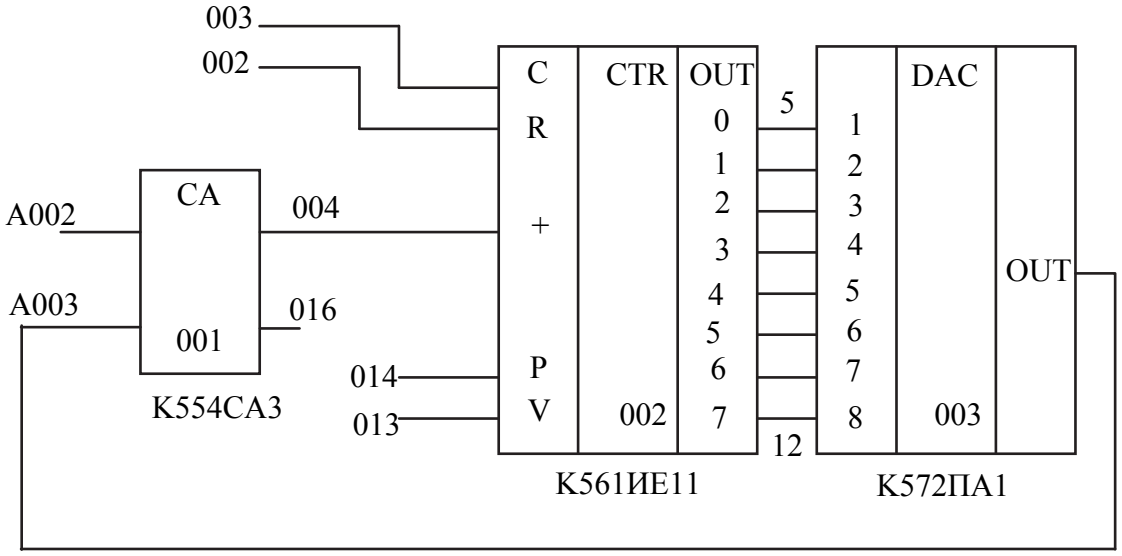


Рис. 2.5. Следящий АЦП с реверсивным счетчиком

Схема следящего АЦП с реверсивным счетчиком FP06R [31] приведена на рис. 2.6.

Рассмотрим пример оптимизации аналого-цифрового преобразователя последовательного счета. Достоинством следящего АЦП последовательного счета с реверсивным счетчиком является устойчивость к помехам. Наличие даже большой помехи на аналоговом входе может привести к более длительному переходу в равновесное состояние, но не приведет к грубой ошибке. Недостатком следящего АЦП является большое время изменения

сигнала обратной связи во всем диапазоне. Сократить это время можно путем разделения последовательного счетчика на секции с разрешением работы секции в зависимости от разности входного сигнала (цепь A002 на рис. 2.5) и сигнала обратной связи (цепь A003 на рис. 2.5). Выходы всех счетчиков, как и в основной схеме на рис. 2.5, подключены ко входу ЦАП. Вместо одного компаратора в основной схеме подключается группа компараторов, количество которых соответствует числу секций счетчика CTR. На выходах компаратора, с целью синхронизации изменений сигнала и исключения промежуточных состояний, устанавливают синхронные триггеры в тех случаях, когда они включены в состав компаратора. Компараторы должны отличаться порогами срабатывания, выходы компараторов подключены к логическим схемам, а выходы последних разрешают работу младших секций счетчика. Нужно оценить оптимальное число секций CTR по выбранному критерию. Для критерия «стоимость – производительность» оптимальная структура АЦП последовательного счета достигается разделением счетчика на две части для старших и младших разрядов, введением дополнительного компаратора и логической схемы, управлением работы счетчиков.

Незначительные аппаратные затраты позволяют на порядок повысить производительность АЦП [31]. Результаты подтверждаются анализом значений критерия эффективности. Основными преимуществами груботочного АЦП являются возможность задания прогнозируемого интервала значений с помощью установочных входов счетчика и снижение входного потока информации [12]. Выходные результаты соответствуют отклонению от их прогнозируемых значений.

Оптимизация структур эффективна для аналого-цифровых устройств, построенных на различных принципах. Например, АЦП одного отсчета может быть построен из двух АЦП одного отсчета меньшей разрядности, ЦАП старших разрядов, аналогового процессора, выполняющего функции вычитания выходного напряжения грубого ЦАП из входного сигнала, умножения разности на масштабный коэффициент, и цифрового сумматора, вычисляющего сумму кодов АЦП младших и старших разрядов. Входной сигнал преобразуется АЦП старших разрядов. Выходной код АЦП старших разрядов преобразуется ЦАП в аналоговый сигнал и вычитается аналоговым процессором (АП) из входного сигнала. С выхода АП разность сигналов, умноженная на масштабный коэффициент (КМ), преобразуется АЦП младших разрядов. Выходной код АЦП старших и младших разрядов подается на цифровой сумматор, с выхода которого считывается результат.

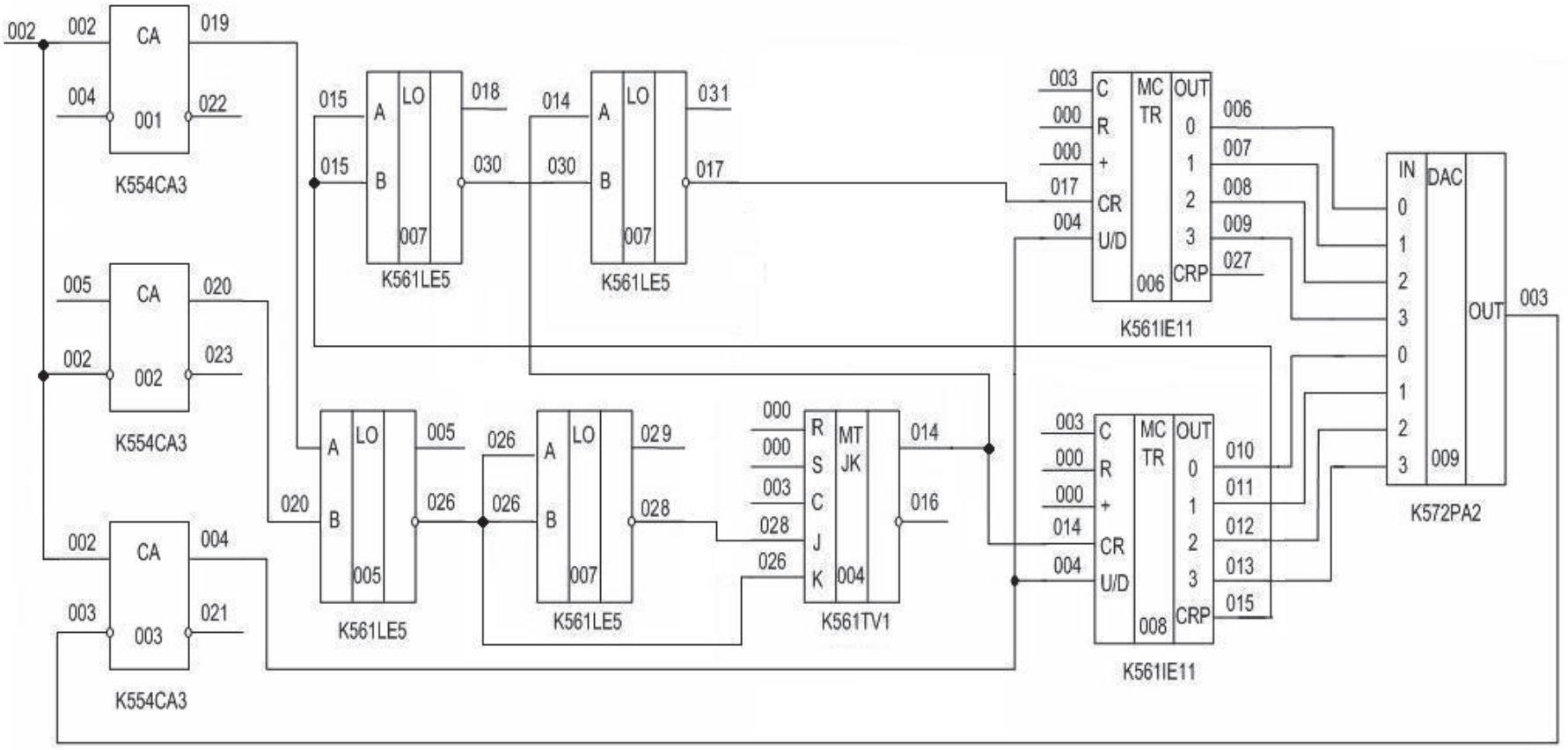


Рис. 2.6. Принципиальная схема груботочного АЦП fp06rgt

Для проверки эффективности АЦП из схемы была удалена вся рабочая часть – грубый счетчик, точный счетчик и логическая структура, обеспечивающая их работу. Вместо них были введены микромашина и два параллельных интерфейсных регистра типа MIR: один для приема информации от компараторов и передачи её микромашине, второй – для получения информации от микромашины и передачи её на ЦАП.

Введение микромашины позволяет производить как оперативную обработку, так и представление результатов оператору. Кроме того, появилась возможность гибко менять алгоритм оперативной обработки. Формализованное задание представлено в файлах fp08padc [27–29].

В микромашине реализованы три режима прогнозирования сигнала: нулевой режим – эмуляция работы следящего счетчика; первый режим – заранее известна скорость изменения сигнала; режим с полной информацией – известен входной сигнал или модель объекта.

2.4. Неоднородные вычислительные системы на базе микроЭВМ

Сложность задач проектирования зависит от неопределенности структуры системы [29–33, 101–103], компонентов и внешних воздействий. В качестве примера рассмотрим выбор варианта вычислительной системы на базе микроЭВМ, структуры которых приведены на рис. 2.7–2.14. Системы предназначены для оперативной оценки параметров входных сигналов по нескольким каналам и отличаются производительностью.

Структура системы, приведенной на рис. 2.7, состоит из микроЭВМ и интерфейсных узлов для ввода и вывода цифровых сигналов. Аналоговые сигналы должны предварительно преобразовываться в код. Обработка информации производится только программно, поэтому производительность системы будет минимальной.

Подсистема на базе микроЭВМ с интервальным прогнозированием и оценкой отклонений сигнала от предполагаемого приведена на рис. 2.8. Зная количество команд на обработку одного входного слова NK и среднюю длительность выполнения команды TK, можно оценить производительность P(1) обработки входных слов [31].

Во вторую структуру включен общий для всех каналов управляемый операционный автомат предварительной обработки входных сигналов. Подсистема на базе микроЭВМ с интервальным прогнозированием и оценкой отклонений сигнала по коммутируемым каналам приведена на рис. 2.10. С целью снижения количества аппаратуры и стоимости операци-

онный автомат выбран общим для всех каналов. МикроЭВМ используется оперативно для управления автоматом, для окончательной обработки результатов и представления их оператору. Управление автоматом сводится к передаче в соответствующий интерфейсный регистр управляющих слов начальной установки для разрешения работы и хранения результатов до их приема в микроЭВМ для каждого из каналов.

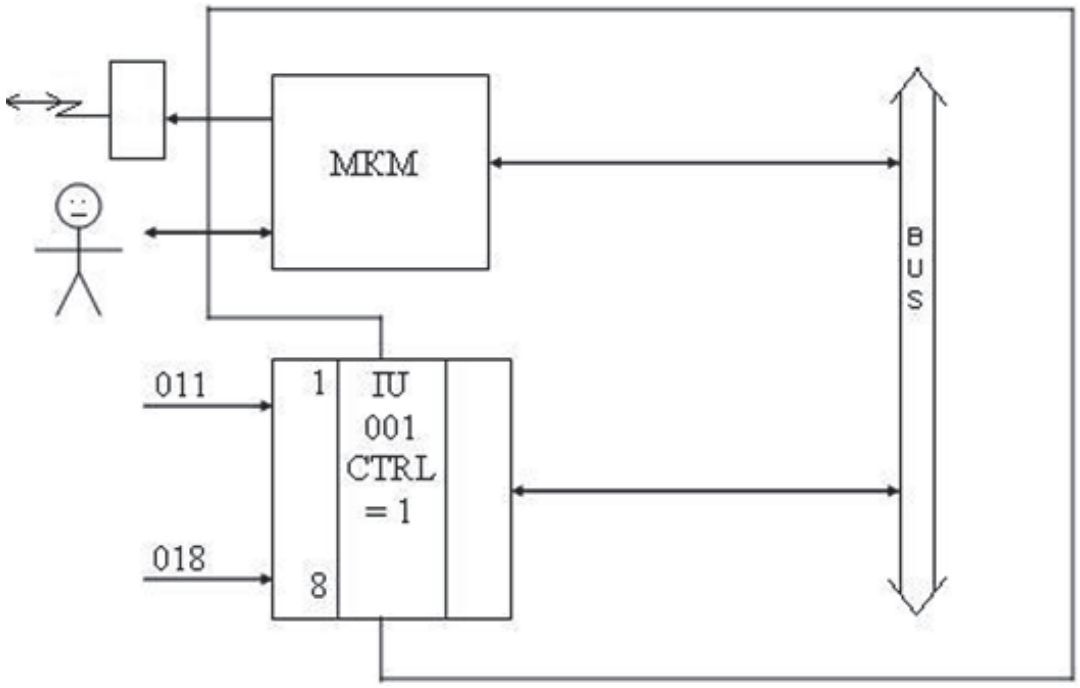


Рис. 2.7. Подсистема на базе микроЭВМ
без дополнительных операционных узлов

Третья структура (рис. 2.11) отличается повышенной производительностью за счет параллельной работы автоматов в каждом канале. МикроЭВМ переводит автоматы в начальное состояние и одновременно разрешает работу. По истечении заданного интервала времени разряды разрешения работы устанавливаются в нуль, и автоматы хранят результат до приема в микроЭВМ. Многоразрядный мультиплексор ММХ экономичнее нескольких интерфейсных узлов на элементах средней степени интеграции. После приема данных и обработки в микроЭВМ результаты представляют оператору.

Четвертая структура (рис. 2.12) аналогична третьей, но более однородна за счет идентичных каналов предварительной обработки. Структура эффективна при наличии большой интегральной схемы (БИС) интерфейсных узлов. Регулярную часть вычислительной системы удобно описывать на языках высокого уровня.

Структура системы на рис. 2.13 представляет собой одноканальный конвейер операционных узлов, управляемых микроЭВМ. Система отличается не только высшей производительностью для отдельного канала, но и большими затратами аппаратуры. Возможен многоканальный вариант системы. Подсистема на базе микроЭВМ со многими процессорами и интерфейсными регистрами с интервальным прогнозированием и оценкой отклонений сигнала по каналам приведена на рис. 2.14.

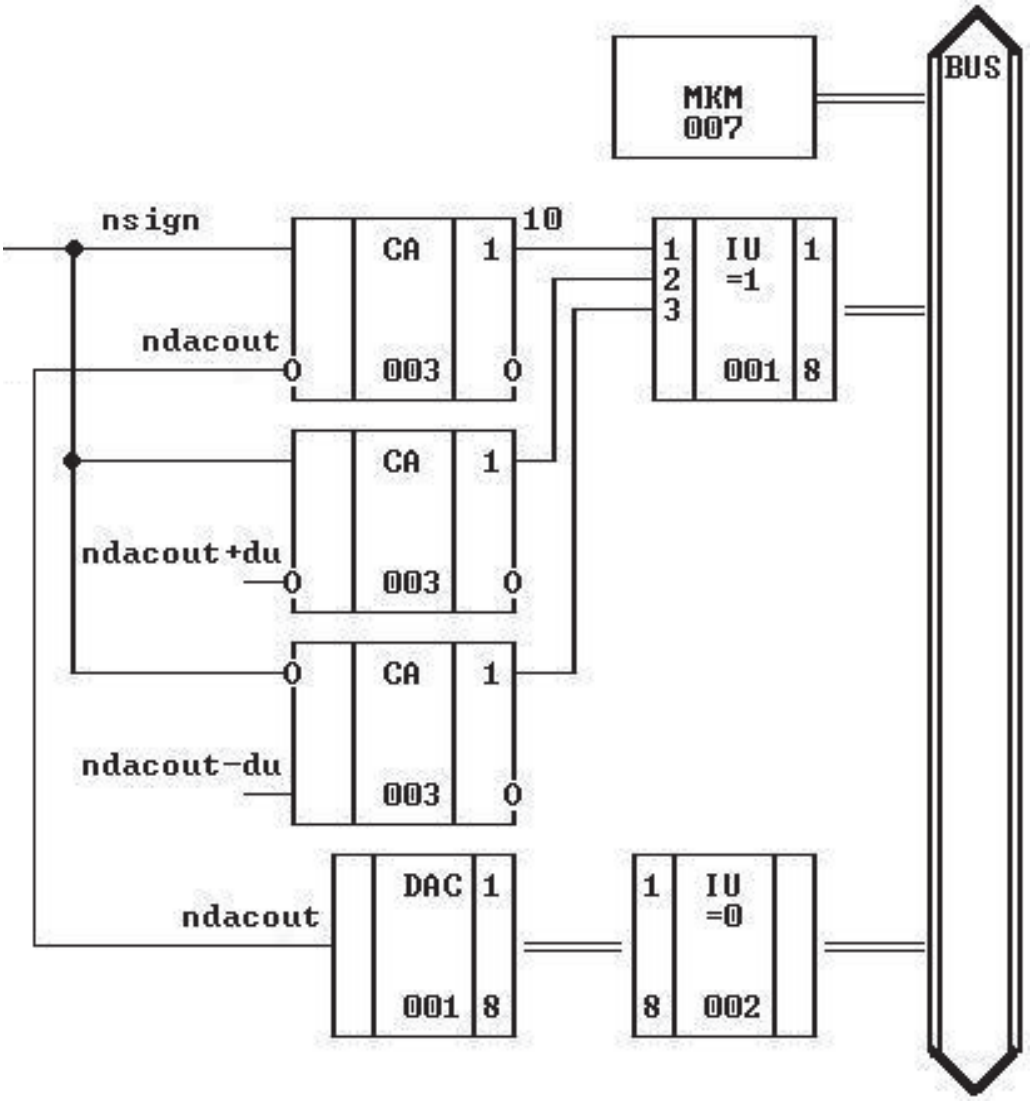


Рис. 2.8. Подсистема на базе микроЭВМ с интервальным прогнозированием и оценкой отклонений сигнала

Для выбора конкретного варианта системы необходимо сравнение вариантов по какому-либо критерию эффективности. В качестве критерия [33] принимают отношение производительности P к стоимости C или отношение стоимости к производительности. В ряде случаев вместо стоимости изделия используют стоимость компонентов, которую проще оценить, или даже

относительное взвешенное количество компонент. В работе используется относительная стоимость компонентов, отношение стоимости компонентов к базовому элементу, в качестве которого принят двухвходовой вентиль. Значение относительной стоимости используется в таблицах параметров компонентов UIPCAD.

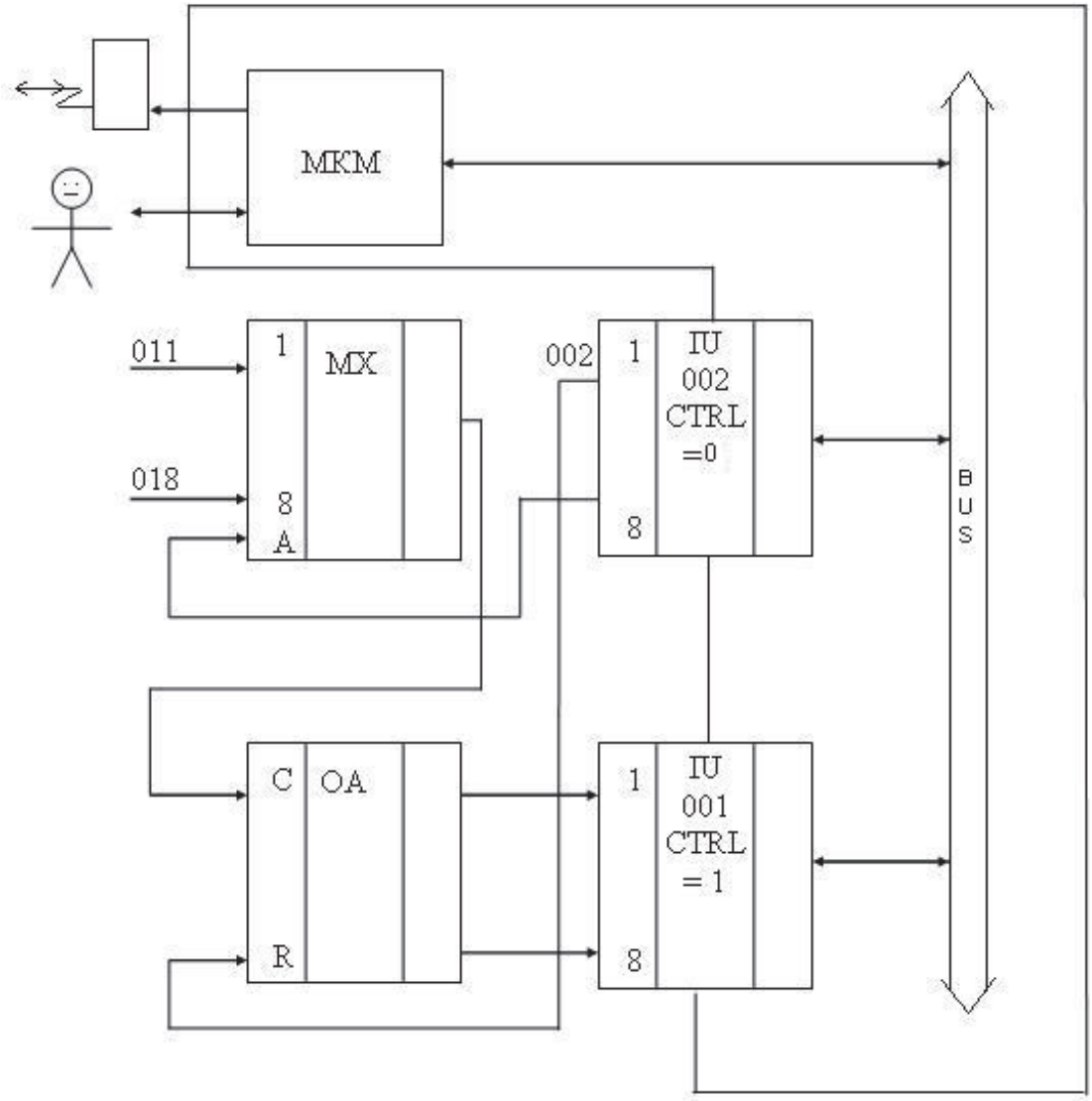


Рис. 2.9. Подсистема на базе микроЭВМ
с общим дополнительным операционным узлом

При ограничении рассеиваемой мощности результат относят к мощности, а при ограничении массы – к массе изделия. Стоимость компонентов изделия оценивают по его составу и реализуют программно в виде суммы. Сложнее оценить производительность. В простых случаях ее выражают аналитически [29, 33].

Для пяти структур вычислительных систем на рис. 2.7–2.14 составлена программа оценки стоимости компонентов *C* и производительности *P*, а также вычисления критерия *K* – отношения «производительность – стоимость». Программа доступна из основного меню и реализована в графическом и табличном вариантах. Синтаксис и семантика параметров программы приведены в табл. 2.3, перечень структур – в табл. 2.4.

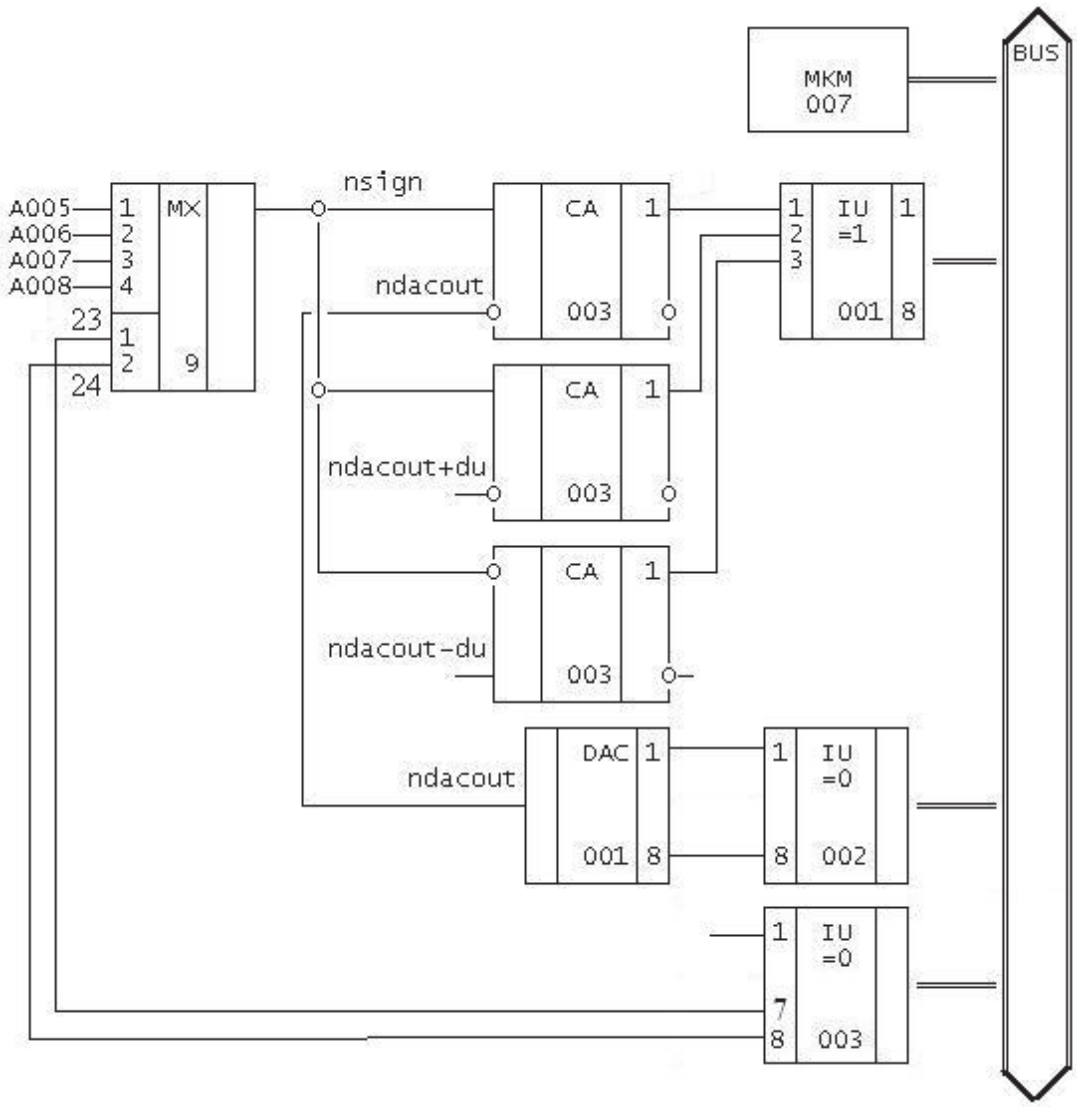


Рис. 2.10. Подсистема на базе микроЭВМ с интервальным прогнозированием и оценкой отклонений сигнала по коммутируемым каналам

Программа в диалоговом режиме позволяет корректировать параметры вычислительной системы по табл. 2.4. Стоимость и эффективность рассмотренных пяти вариантов систем выводится в виде графика и таблицы. Программу EFFECTG можно использовать для исследования зависимости оптимального варианта системы (рис. 2.7–2.14) от изменения стоимости компонентов или технических характеристик. Стоимость компонентов

и технические характеристики определяются технологией и уровнем развития электронной техники. Поэтому структура ВС, далекая от оптимальной сейчас, может дать лучший результат в будущем. Для исследования выбираем изменяемый параметр, в соответствии с интервалом изменения которого в программе организуется цикл. Результаты лучше оформить в виде графика зависимости оптимального варианта от изменяемого параметра.

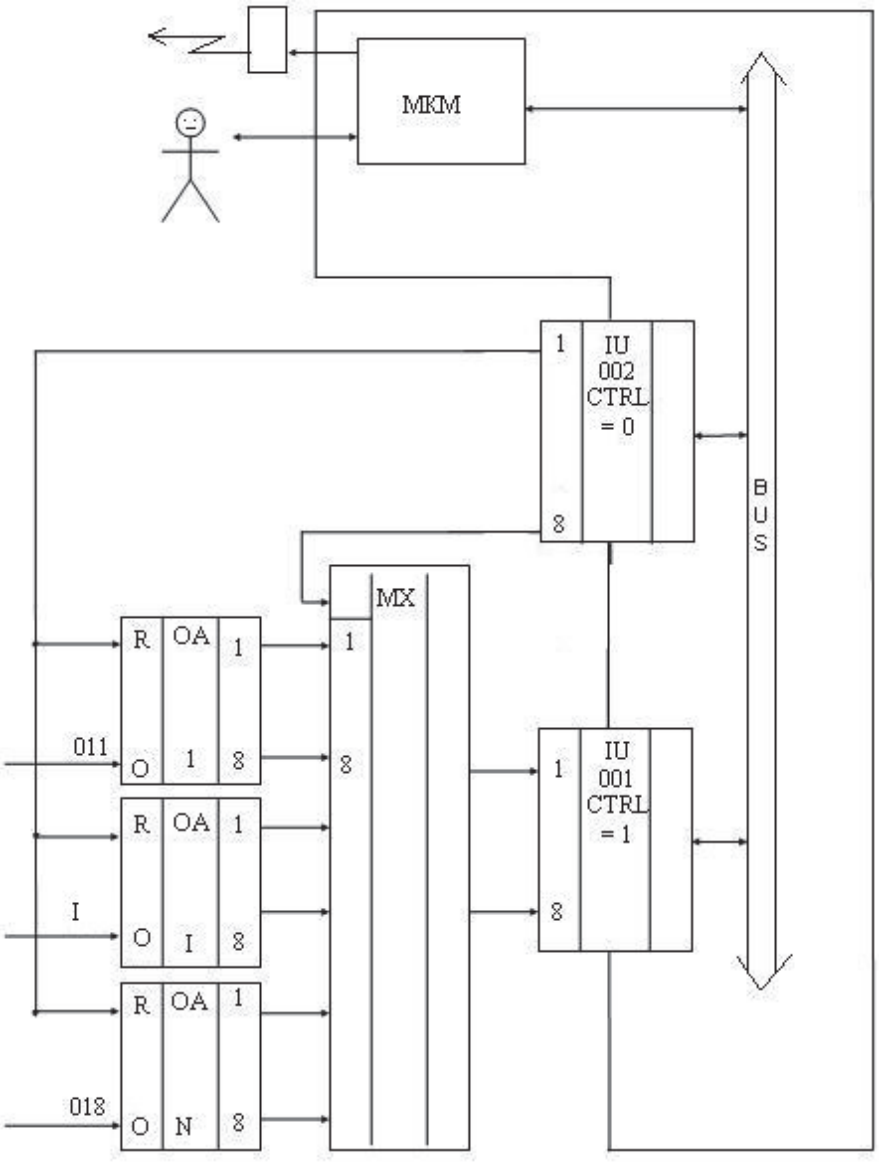


Рис. 2.11. Подсистема на базе микроЭВМ с индивидуальными операционными элементами для первичной обработки сигналов и общими интерфейсными узлами

В результате анализа выбирают из приведенных на рисунках структур вычислительных систем или дополнительно предложенную, оптимальную

по критерию эффективности. Выбранную систему оптимальной структуры описывают и анализируют как поведение, так и оценки ресурсов. После доработки описание системы может быть передано на конструкторский уровень проектирования.

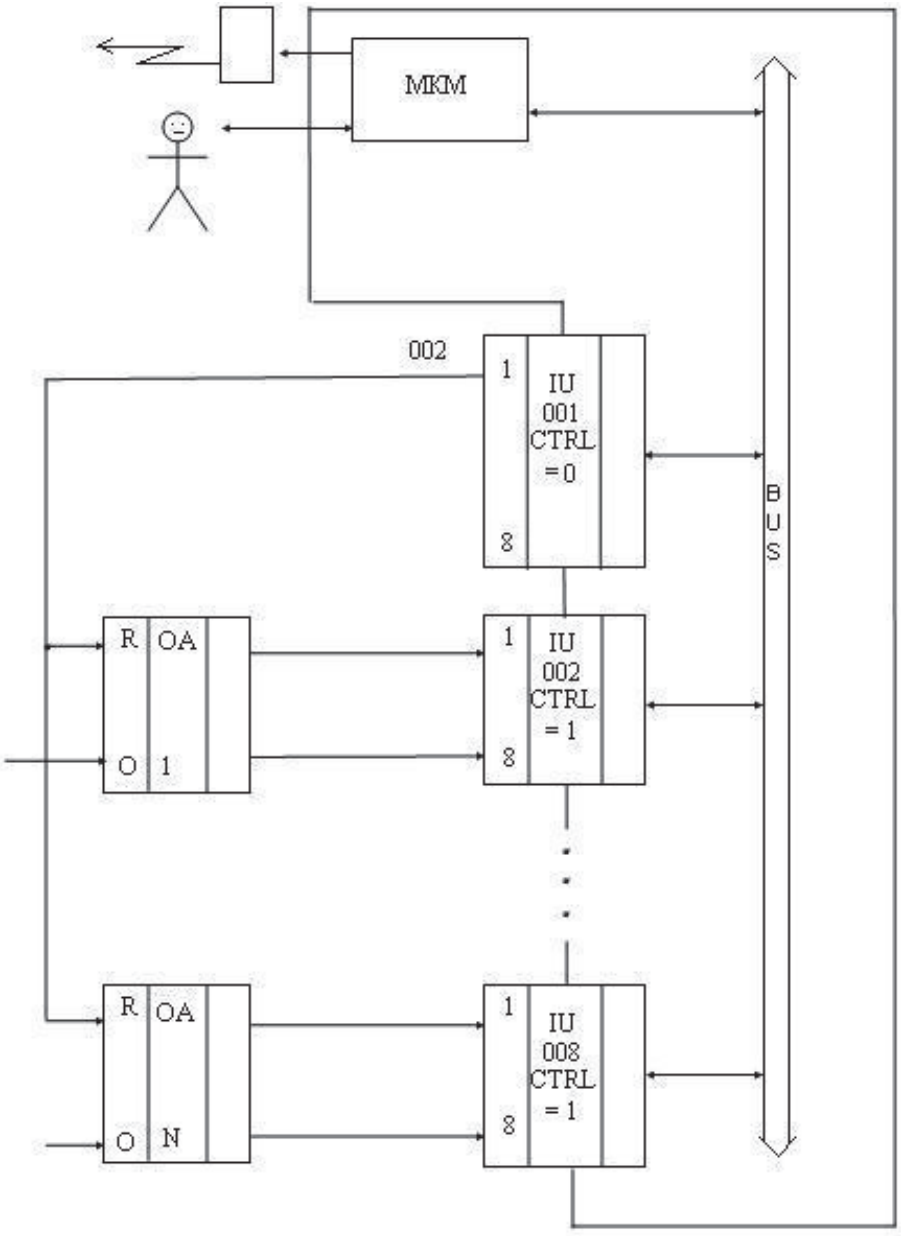


Рис. 2.12. Подсистема на базе микроЭВМ с индивидуальными операционными элементами и интерфейсными узлами для первичной обработки сигналов

На рис. 2.7 представлена подсистема на базе микроЭВМ без дополнительных операционных узлов. При использовании микроЭВМ в груботочном интервальном АЦП на рис. 2.6 можно исключить цифровые компоненты и получить схему на рис. 2.8. Состояния компараторов вводятся

с помощью первого интерфейсного регистра, а через второй предполагаемые значения выводятся на ЦАП. На рис. 2.9 представлена схема вычислительной системы с общим операционным автоматом ОА и коммутацией входных сигналов. Многоканальный вариант с груботочным интервальным АЦП представлен на рис. 2.10. Дополнительный третий интерфейсный регистр служит для управления мультиплексором.

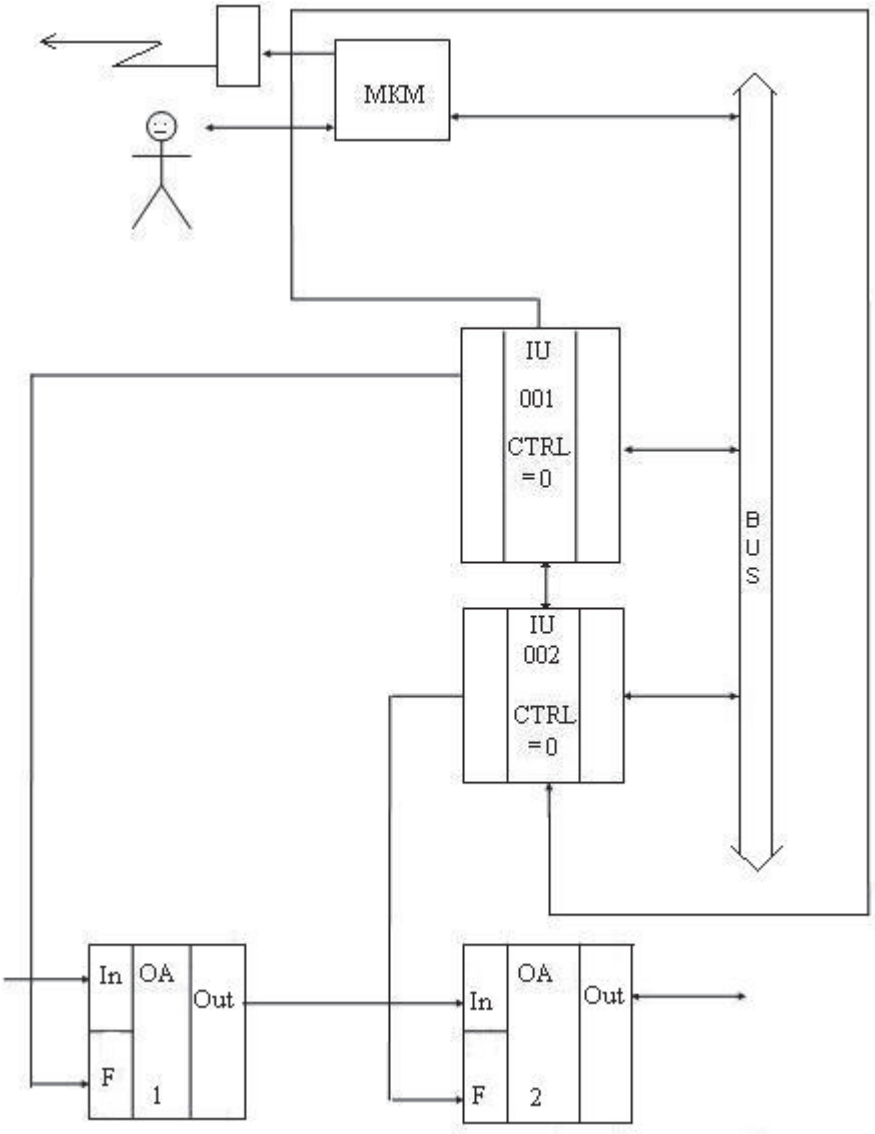


Рис. 2.13. Система на базе микроЭВМ с конвейером управляемых операционных узлов

Применение многопроцессорной микроЭВМ с индивидуальными регистрами для интервальной оценки сигналов показано на рис. 2.14. Блок компараторов можно заменить на АЦП одного отсчета с небольшим числом (4–6) разрядов. Подобные схемы приведены на рис. 4.19 и 4.20.

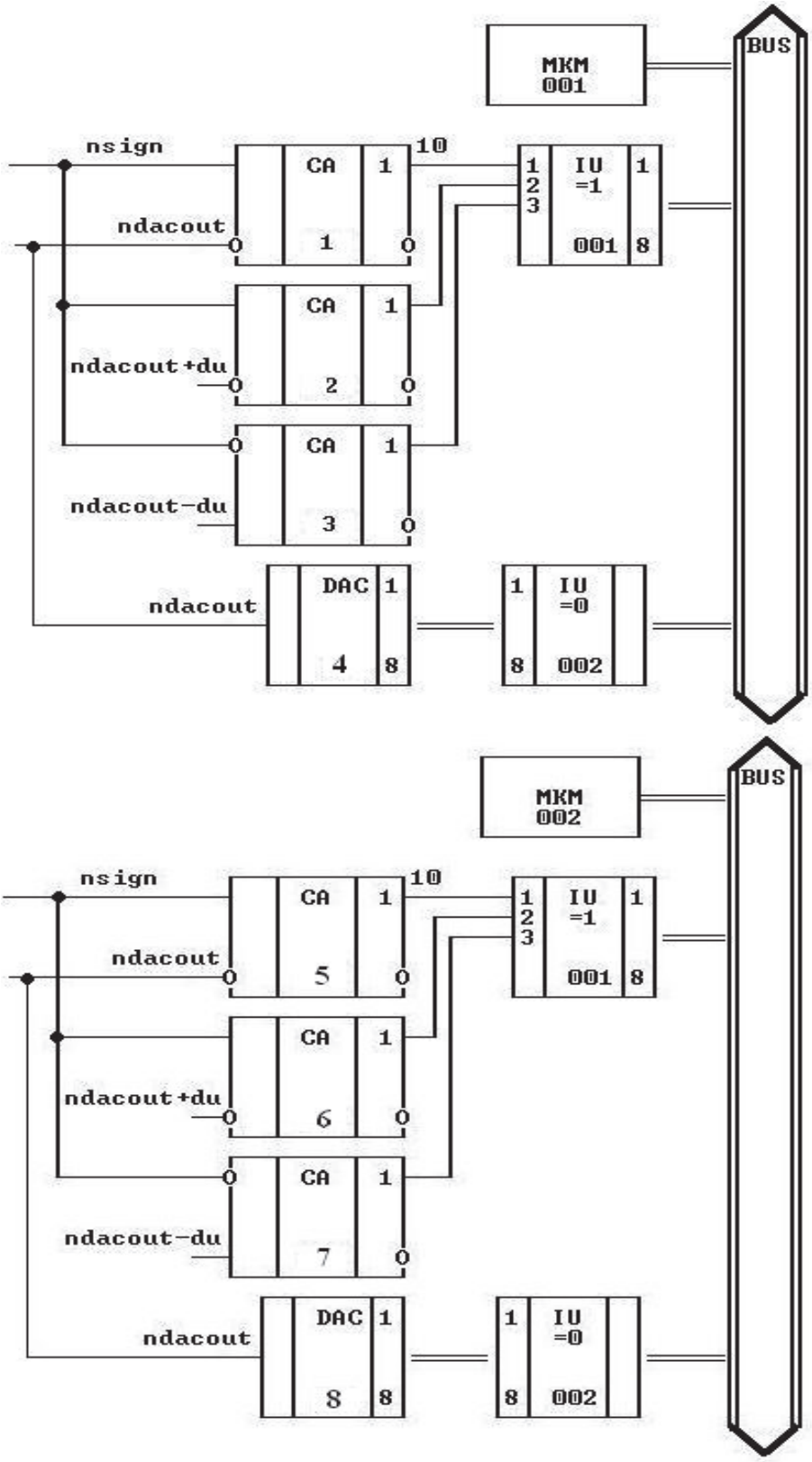


Рис. 2.14. Система на базе многопроцессорной микроЭВМ

Таблица 2.3

Синтаксис и семантика параметров процедуры оценки эффективности структуры системы

№ п/п	Идентификатор	Формат	Семантика	Примечания
1	CMKM	int	Стоимость микроЭВМ	Относительные единицы
2	CIU		Стоимость интерфейсного узла	Относительные единицы
3	COE		Стоимость операционного элемента	Относительные единицы
4	CMX		Стоимость мультиплексора	Относительные единицы
5	NCAN		Количество каналов	Число
6	TCAN		Время на канал	Такты
7	NK		Количество команд на обработку	Число
8	DELT	float	Длительность выполнения команд	с
9	FC		Частота синхросигнала	Гц
10	LRG	int	Разрядность регистров	Число
11	FAM	char	Идентификатор пользователя	FA22218

Таблица 2.4

Структуры вычислительных систем

Номер структуры	Тип структуры подсистемы на базе микроЭВМ
1	Без операционных узлов
2	Общий операционный узел
3	Каналы операционных узлов и общий процессор
4	Каналы операционных и интерфейсных узлов
5	Конвейер управляемых операционных узлов
6	Каналы операционных узлов на процессорное ядро

Основное внимание нужно уделить системам с прогнозом интервала сигналов или с моделью объекта: с накоплением знаний о сигналах и объектах в процессе жизненного цикла систем уменьшается поток входной информации и увеличивается поток прогнозных оценок сигналов.

2.5. Вычислительные системы из сетевых устройств и вычислительных машин

Сетевые программируемые устройства с датчиками составляют основу решений для автоматизации стационарных и мобильных объектов. Обработка аналоговых сигналов локализована, и передача данных производится в цифровой форме. Каждое устройство имеет сетевой адрес и доступ

из сетевой программы просмотра для управления и считывания результатов. Основные варианты программируемых устройств на базе микроЭВМ с различным уровнем аппаратной поддержки рассмотрены в п. 2.4. Подсистемы хранения и обработки данных являются стационарными и размещаются в дата-центрах (DC), а подсистемы сбора и первичной обработки могут содержать стационарные и мобильные устройства и стали называться интернет-устройствами (IOT). Концентрация обработки и хранения информации продолжается и будет потреблять значительную долю вырабатываемой электроэнергии [64, 107, 110, 143, 191]. В отличие от интернет-устройств дата-центры нужно рассматривать как комплексные объекты, и к ним должны применяться стандарты информационного моделирования [188, 191].

Синтез локальной сети, в виде формализованного задания, для решения задач САПР производится в среде САПР COD [27, 33]. Компонентами сети являются машины пользователей, серверы и коммуникационное оборудование. Машины пользователей отличаются типом, характеристиками рабочего места и мобильностью. Мобильность оценивается скоростью перемещения компонента (м/с) и для стационарных машин равна нулю. Реально перемещается не компонент, а его носитель. Например, мобильный персональный компьютер (MPC) может размещаться на рабочем месте – столе, при этом мобильность равна нулю, или в автомобиле, тогда мобильность равна скорости движения транспортного средства. В зависимости от положения машины, ее скорости перемещения MPC может быть компонентом сети или нет. Мобильные интернет-устройства (Mobile Internet Devices – MID) могут использоваться на рабочем месте, при перемещении человека или в транспортном средстве. Коммуникатор (PPH) предназначен для передачи данных и голосовой связи и перемещается вместе с человеком. Серверы (SERV), коммутаторы (SWITCH) и точки беспроводного доступа (AP) [4, 7, 10–12, 107, 110], как правило, размещаются стационарно. Модели компонентов характеризуются таблицами синтаксиса и семантики. Каналы передачи данных отличаются принципами действия и соответствуют международным стандартам [4]. Принцип действия канала передачи данных должен быть согласован с портом (выводом) компонента.

С технологическим и испытательным оборудованием связаны серверы приложений, на них же выполняются формализованные задания [4, 7, 10–12, 107, 110]. Контент для чтения может размещаться на сетевом сервере, а при выполнении задания может переключаться на выбранный сервер приложений. Стационарные объекты размещаются в соответствии с правилами и нормами. Мобильные объекты образуют динамически изменяющуюся среду, и для них необходима специальная структура данных о состоянии и скорости перемещения.

2.6. Процессы синтеза и анализа неоднородных вычислительных систем

Процедуры синтеза позволяют получить описание объекта, а процедуры анализа – оценить свойства объекта и его поведение. Для простых объектов трудоемкость анализа меньше трудоемкости описания, но с увеличением сложности объектов она значительно возрастает. Поэтому автоматизация анализа сложных объектов повышает эффективность проектирования. Процедуры структурного анализа позволяют оценить функции и поведение объекта, состав, соответствие компонентов и связей, параметры связей. Процедуры параметрического анализа обеспечивают получение информации о статике и динамике объекта во временной или частотной области, анализ чувствительности.

Наряду с процедурами основного анализа необходимы новые процедуры дифференциального анализа для оценки отличий структуры или параметров объектов. Примером является процедура сравнения группы сигналов. Вычислительная система как объект проектирования представляется множеством взаимодействующих компонентов, каждый из которых можно рассмотреть как составной объект. Модель абстрактного объекта представляется структурой данных, значение которой отражает его состояние, методами или операциями, преобразующими состояния объектов в соответствии с воспринимаемыми входными сообщениями. Модель объекта формирует выходные сообщения или значения параметров процедур для других объектов.

Описание проектного решения, достаточное для автоматизированного ввода и интерпретации формальной системой, называется формализованным заданием, широко используемым в САПР. Формализованные задания (*FZ*) состоят из множества разделов, а разделы (*Pi*) – из предложений соответствующего языка:

$$FZ = \{P1, P2, ..., PN\}. \qquad (2.2)$$

Каждый раздел синтаксически и семантически однороден и соответствует определенному отношению. В качестве примера рассмотрим исследовательскую САПР COD, в которой формализованное задание состоит из разделов, представленных в табл. 2.5.

В разделе INPUT описываются отношения между номерами цепей, типом сигнала, тактами начала и конца изменения сигнала. Информацию раздела можно представить в текстовой форме.

В разделе UNIT описываются отношения между типами и номерами компонентов, номерами цепей, подключенных к выходам и входам, при-

чем описание производится покомпонентно. Информацию раздела также можно представить в текстовой форме.

Таблица 2.5

Разделы формализованного задания

Имя раздела	Описание раздела
INIT	Начальная установка
INPUT	Описание внешних воздействий
UNIT	Описание устройства
CTRL	Управление
MOD	Модели новых компонентов

Функциональные модели компонентов MFComp состоят из разделов PMF:

MFComp=<PMF1, PMF2, …, PMF5>.

Разделы описания функциональных моделей приведены в табл. 2.6.

Таблица 2.6

Разделы функциональной модели компонента

Имя раздела	Имя раздела (метка)	Описание раздела
1	FINIT	Проверка допустимости параметров, размещение структур данных и их заполнение
2	FTYP	Контроль типов компонентов и выбор возможных реализаций
3	FS	Восстановление состояний компонентов
4	FOUT	Формирование и контроль выходных сигналов
5	FEND	Освобождение ресурсов

Конкретные функциональные модели компонентов приведены во втором, третьем разделе.

Рассмотрим особенности текстовой и табличной форм описания. Табличная форма позволяет описать конкретное отношение и используется в САПР низкого уровня, например в САПР PCAD [22], и в качестве промежуточного уровня одного варианта технического решения в САПР COD. Текстовая форма используется в исследовательской системе COD, комплексной САПР ПРАМ5.3, пакетах моделирования [1, 7, 13, 120–122] и описания вычислительных устройств [13]. Текстовая форма на языках низкого уровня позволяет описать один вариант конкретного технического решения и эквивалентна табличной форме. Текстовая форма с использова-

нием языка высокого уровня позволяет описывать как конкретные технические решения, так и множество технических решений. Примеры описаний на языках высокого уровня PL/1, C++, JAVA и ADA приведены в [27, 31, 113, 114], а на языке VHDL – в [8, 115].

Однако полная конкретизация компонентов и соединений не позволяет представить множество проектных решений одним описанием. Введение переменных видов компонентов и соединений позволяет объединить множество вариантов проектных решений в одном формализованном задании.

Отношение включения и основные свойства элементов с указанием нормативного документа обычно объединяются в одноименный раздел формализованного задания, называемый *перечнем элементов*. Отношения между компонентами сводятся к наличию или отсутствию гальванической связи между контактами или оптической связи между портами. Эти отношения можно записать по компонентам или по цепям. При описании по компонентам каждому типу компонента соответствует определенное множество цепей, с которыми гальванически связаны контакт и компонент. Это отношение может быть описано с помощью правил какого-либо языка. Хорошо документированы стандартные языки программирования [1, 9, 10, 29]. Например, можно использовать непроцедурную или табличную формы. Непроцедурная форма обычно используется на технических этапах проектирования, а процедурная форма часто применяется для функционального или логического анализа работы устройств или систем. Универсальные языки программирования знакомы студентам с первого курса, что облегчает освоение синтаксиса описаний. Для трансляции и интерпретации описаний используется только базовое программное обеспечение ЭВМ, дополненное правилами построения и размещения разделов формализованного задания, правилами покомпонентного описания структуры устройства и программными моделями компонентов. Внешние воздействия описываются по входным цепям и интерпретируются специальными модулями.

2.7. Принципы построения многоуровневых систем моделирования и проектирования вычислительных систем

В процессе проектирования исходное описание преобразуется в требуемое описание – проектное решение. Для конкретизации проектных решений требуется дополнительная информация. Исходным описанием в многоуровневой САПР является формализованное задание. Формализованное задание, использующее возможности языка высокого уровня, представляет описание множества технических решений. В промышленных САПР [8]

проектом является одно техническое решение. Поэтому в многоуровневой САПР между верхним уровнем, в котором можно представить множество технических решений, и нижним уровнем, представляющим одно техническое решение, должен быть средний уровень с полной информацией о варианте технического решения и параметрах компонентов.

Многоуровневая САПР COD служит для синтеза и анализа множества вариантов структур и их автоматического преобразования во множество проектных решений для промышленных САПР. И состоит из множества подсистем:

$$COD = < HSC, COMM, SAT, AAT>, \quad (2.3)$$

где HSC (Human Control) – подсистема управления проектированием, служит для облегчения работы пользователя при переходе на второй уровень сложности задач проектирования; COMM – коммуникационная подсистема проектирования. Обеспечивает возможность проектирования объектов в сети Интернет; SAT (Synthesis Automation Tools) – инструментальные средства автоматизированного синтеза объектов; AAT (Analysis Automation Tools) – инструментальные средства автоматического анализа поведения, оценки ресурсов и сравнения объектов.

Подсистема управления проектированием HSC состоит из множества подсистем:

$$HSC = < SETSEL, PRJSEL, RESSEL, RPRJ>, \quad (2.4)$$

где SETSEL – подсистема выбора формализованного задания, языков описания проекта, выбора САПР и типа описания для импорта, выбора САПР и типа интерфейса для экспорта, языка сообщений и сервера в сети; PRJSEL – подсистема выбора результатов проектирования; RESSEL – подсистема выбора представления результатов; RPRJ – правила выбора последовательности проектных процедур и операций, маршрутов проектирования для получения результатов проектирования и анализа. По мере развития комплекса увеличивается доля правил, реализуемая в подсистемах синтеза и анализа. Например, в подсистеме анализа автоматически сравниваются предполагаемые и фактические результаты, а также описания объектов и требуемые ресурсы.

Пользователь САПР COD выбирает требуемый результат проектирования или анализа, а не последовательность проектных процедур и операций для достижения цели. Таким образом снижается нагрузка на пользователя и повышается уровень интеллекта комплекса [31, 70, 87, 126, 127]. Формализуемая часть подсистемы управления проектированием представлена в форме оболочки САПР COD, которая может быть реализована различными средствами. В [11–13, 29, 33] представлены реализации оболочки для различных операционных систем на базе многофункционального ре-

дактора, входящего в инструментальные средства IBM Visual Age, сетевых программ просмотра (Mozilla Firefox, Google Chrome) и инструментальной среды ECLIPSE.

Многовариантный анализ, оценка и выбор требуемого решения возможны при управлении внешними воздействиями, видами компонентов и их работоспособностью, наличием и отсутствием соединений. Средства универсального языка высокого уровня позволяют решать подобные задачи на этапах функционального и структурного проектирования.

В процессе проектирования формализованное задание на языке высокого уровня, представляющее множество технических решений, преобразуется в текстовую или табличную форму уровня одного варианта решения, которая может быть представлена в текстовом, командном или табличном формате конкретной САПР или в виде формализованного задания (ФЗ) [29–33]. Количество вариантов определяется исходным ФЗ.

Построение системы из множества независимых модулей, базовой модели – словаря терминов и информационной модели, связывающей известные термины с правилами выполнения модулей, называется *функционально-ориентированным проектированием* [87].

Программно-методический комплекс COD в выражении (2.3) содержит полное множество компонентов. Возможны серверные и клиентские версии с частью компонентов или клиентские версии с добавлением компонент по мере увеличения сетевых сервисов САПР. Информационная часть комплекса может находиться на сетевом сервере (например: e.sfu-kras.ru), а выполнение заданий передается на один из серверов приложений. Серверы приложений могут отличаться операционными системами (Windows, OS2 или ECS, LINUX, VM) и аппаратурой. Стандарты сетевых сервисов позволяют клиентам выполнять задания на любом предлагаемом в меню сервере.

Клиент должен содержать программу просмотра, с помощью которой он выберет на сервере формализованное задание и выполнит его. Для просмотра результатов моделирования ему требуется программа просмотра временных диаграмм, для просмотра схем PCAD – бесплатная программа VIEWER версии PCAD, для выполнения командных файлов конкретной САПР – пакет САПР (PCAD, ORCAD, CATIA, AUTOCAD EAGLE, CADDy). Принципиальную схему можно увидеть в программе просмотра, если установлен пакет JAVA, а объемную модель с изменением сигналов или активности – при наличии плагинов виртуальной реальности или X3D для программ просмотра. Для снижения объема передаваемых данных результат передается и хранится в символьном виде и преобразуется в изображение по требованию клиента. На оптическом диске находятся программно-

методический комплекс COD, примеры для выполнения FA, демонстрационные версии САПР и программ просмотра [33].

Для выполнения заданий на серверах САПР требуется значительное число операций и энергии. Ограниченные операционные и энергетические ресурсы мобильных клиентов стимулируют развитие сервисов САПР.

Система автоматизированного проектирования конструкторского и технологического проектирования может использоваться в диалоговом или пакетном режимах. Примером диалоговой САПР на базе ПЭВМ может служить PCAD [120–122] и ее развитие ГРИФ-4. Несмотря на недостаточные средства диалога в комплексной промышленной САПР ПРАМ5.3 [30, 101] в ней предусмотрено согласование подсистем, реализующих различные этапы проектирования. Поэтому предпочтение следует отдавать комплексным САПР с согласованием результатов отдельных этапов, которые позволяют организовать сквозной цикл «проектирование – производство».

Внедрение АРМ первого поколения не изменило технологии проектирования, но переход к комплексным САПР позволяет внедрить новые технологии проектирования, производства и испытаний. Повышение эффективности использования создаваемых объектов достигается улучшением проектных решений путем синтеза и анализа большего числа вариантов. Автоматизация даже отдельных этапов проектирования в рамках комплексных САПР позволяет снизить время и затраты на получение одного решения, что обеспечит многовариантное проектирование без увеличения времени и трудозатрат инженеров.

2.8. Информационные основы преобразования описаний в многоуровневых САПР

САПР COD позволяет автоматически оценить ресурсы для конкретных типов компонентов. Возможна оценка ресурсов для множества конкретных типов компонентов при многовариантном анализе. Переменные типы компонент являются массивами конкретных значений, размерность которых определяется числом вариантов (NVARMAX). Оценки ресурсов выводятся по каждому варианту в отдельности и в целом по всем вариантам при каждом анализе формализованного задания. Для вычисления критериев эффективности необходима оценка производительности подсистемы (Мбит/с). Значение производительности присваивается переменной PP вещественного типа (float). Приведем пример оценки ресурсов первого варианта схемы переноса в формализованном задании fp02s. Производи-

тельность схемы оценена по изменению выходного сигнала, заданного номером цифрового сигнала nppd (vext.nppd для Java).

Сигнал цифровой 11. Количество изменений – 2.

Общая потребляемая мощность, МВт	GPF	20
Общая площадь микросхем, мм^2	GS	65
Общая масса микросхем, г	GM	3
Общая стоимость, отн. ед.	GC	3.2
Производительность, Мбит/с	PP	2.5
Стоимость/производительность, отн. ед /(Мбит/с)	KEFC	1.28
Масса/производительность, г/(Мбит/с)	KEFM	1.2
Мощность/производительность, мДж/Мбит	KEFP	8

Производительность определяется с помощью тестов [1, 13, 105, 107], по выполненной полезной работе в единицу времени или по изменению цифрового или аналогового сигнала за заданный интервал времени. При оценке производительности задается номер изменяющегося цифрового (nppd) или аналогового (nppa) сигнала.

Допускается одновременное задание номеров цифрового и аналогового сигналов для оценки производительности, которая вычисляется и выводится для обоих сигналов. Критерий эффективности вычисляют по производительности аналогового сигнала. Для аналогового сигнала сумма изменений должна умножаться на эффективную разрядность (lra), определяемую как двоичный логарифм полной относительной погрешности (era):

$$\mathrm{lra} = \log_2(\mathrm{era}).$$

Начальные значения эффективной разрядности и погрешности имеют начальное значение минус один (–1), которое не может быть реально присвоено и свидетельствует об ошибке, поскольку производительность принимает отрицательное значение. Эффективная разрядность оценивается в модели компонента при совпадении номера цепи выходного аналогового сигнала с номером сигнала оценки производительности (nppa).

Алгоритм для оценки эффективной разрядности и производительности по изменению аналогового сигнала приведен на рис. 2.15. Он позволяет оценить эффективную разрядность аналогового сигнала как в случае, когда длительность такта превышает время завершения переходных процессов и основной является статическая погрешность, так и в случаях, когда длительность такта меньше времени завершения переходных процессов.

Для цифрового сигнала оценивают количество его изменений (KDOUT), исключая выбросы и помехи, длительность интервала времени (равного разности тактов конца (NTEND) и начала (NTBEG) интервала) и такта

(DELT). Таким образом, информационная производительность сигнала в Мбит/с

$$PP = 10^3 \cdot KDOUT/(NTEND - NTBEG + 1)*DELT.$$

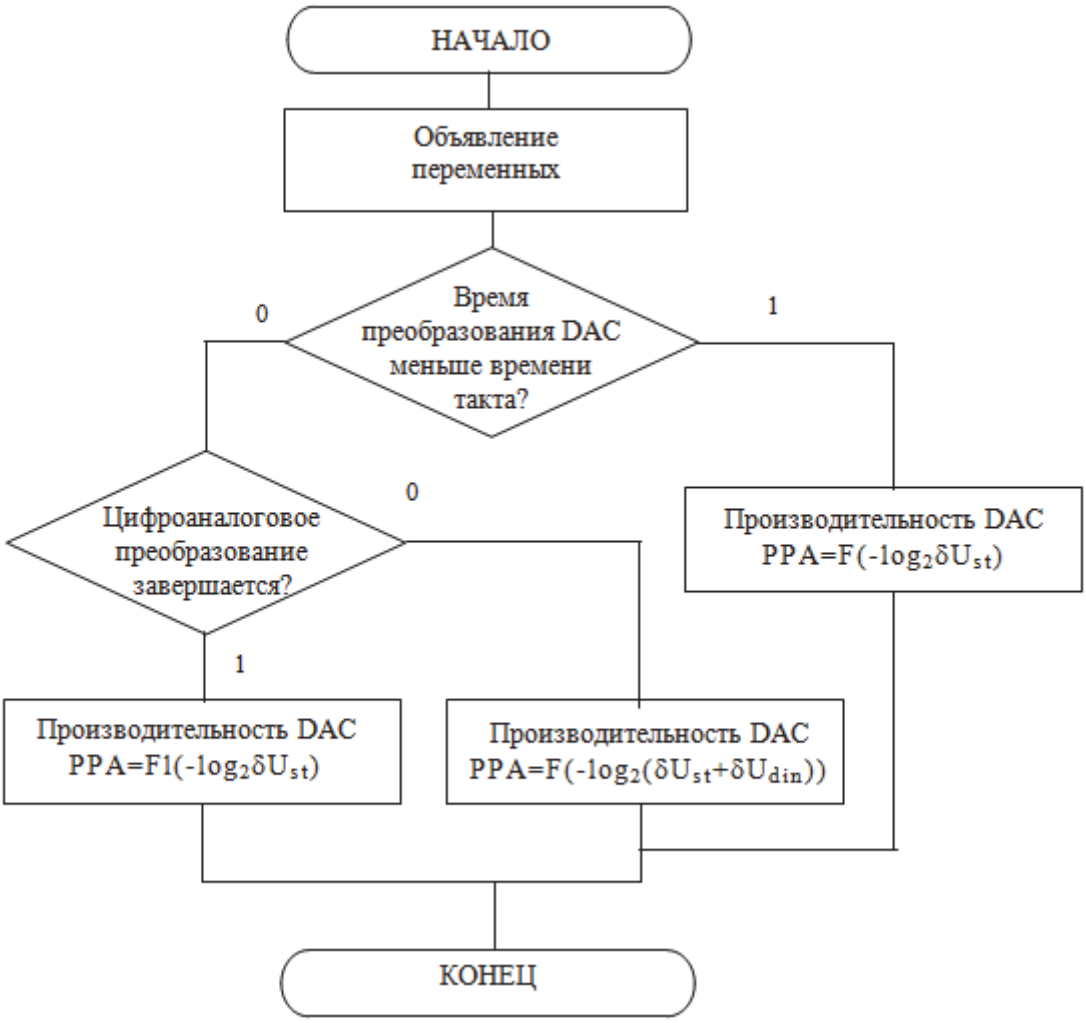


Рис. 2.15. Блок-схема алгоритма оценки производительности выходного сигнала цифроаналогового преобразователя (DAC)

Для оценки информационной производительности цифровых и аналоговых сигналов по их изменению за заданный интервал времени по алгоритмам, составленным пользователем, введены функции Sign PPD и Sign PPA на языках C++, PL/1, JAVA. Синтаксис и семантика параметров процедур SignPPD и SignPPA приведены в табл. 2.7.

Таблица 2.7

Синтаксис и семантика параметров функций SignPPD и SignPPA

Номер п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика	Примечание
1	NSIGNAL	DEC FIXED(3)	Int	Integer	Номер сигнала	–
2	TYP	CHAR(*)	char C++) String (Java)	Character	Тип компонента	Тип может быть абстрактным и конкретным
3	NTBEG	DEC FIXED(6)	Int	Integer	Номер такта начала оценки	–
4	NTEND	DEC FIXED(6)	Int	Integer	Номер такта конца оценки	–

Использование предложенных имен функций и параметров для оценки производительности обеспечивает выполнение интерфейсов для ФЗ с конкретной САПР. Примеры функции оценки производительности цифровых сигналов на универсальных языках программирования приведены ниже. Словарь процедур оценки производительности представлен в табл. 2.8.

На языке PL/1 процедура SignPPD находится в файле SignPPD.inc:

```
SIGNPPD: PROC(NSIGNAL,TYP,NTBEG,NTEND);
 DCL TYP CHAR(*); DCL (NSIGNAL,KDOUT INIT(0) STATIC)
 FIXEDEC(3);
 DCL (NTBEG,NTEND) FIXED DEC(6);
   IF (PPT&(NT>NTBEG)&(NT<=NTEND)) THEN
    IF (OUT.NEX(NSIGNAL)^=OUT.OLD(NSIGNAL)) THEN
      KDOUT=KDOUT+1; /* НАКОПЛЕНИЕ ИЗМЕНЕНИЙ */
       IF(NT=NTMAX)&PPT THEN DO;
 PP=KDOUT/((NTEND-NTBEG+1)*DELT*1E-3);
 PUT SKIP EDIT(' СИГНАЛ ',NSIGNAL)  (A,F(6));
 PUT SKIP EDIT(' КОЛИЧЕСТВО ИЗМЕНЕНИЙ ',KDOUT) (A,F(6));
 PUT SKIP EDIT(' PP-ПРОИЗВОДИТЕЛЬНОСТЬ MBIT/C ',PP)
 (A,E(14,3,6)); END;
END SIGNPPD;
```

Функции SignPPD на языках C++ и Java находятся в файлах control.cpp и control.jav.

Приведен пример функции SignPPD на языке C++:

```
void SignPPD(int nsignal,char *typ,int ntbeg,int ntend)
    { static int kdout;
    if(nt==ntmin) kdout=0;
     if((nt>=ntbeg)&&(nt<=ntend))
     if((ppt==1)&&(out[nsignal].Nex!=out[nsignal].Old))
 /* сравнение текущего и предыдущего  сигналов */
  kdout=kdout+1; /* накопление изменений   */
    if((nt==ntmax)&&(ppt==1))
{ pp=kdout/((ntend-ntbeg+1)*delt*1e-3);
 printf("\n сигнал %d  количество изменений %d ",nsignal,kdout);
 printf("\n pp-производительность Mbit/c %e ",pp); }
    } //End SignPPD
```

Приведен пример функции SignPPD на языке Java:

```
 public static void SignPPD(int nsignal,String typ,int NtBeg,int NtEnd)
    { int kdout,kppd=0;
  if(vext.nt==vext.ntmin) vext.pp=0;
     if((vext.nt>=NtBeg)&&(vext.nt<=NtEnd))
  if((vext.ppt==true)&&(vext.out.Nex[nsignal]!=vext.out.Old[nsignal]))
/* сравнение текущего и предыдущего  сигналов */
vext.pp=vext.pp+1; /* подсчет изменений */
  if ((vext.ppt==true)&&(vext.nt==vext.ntmax))
  { if (vext.nppd > 0) kppd=vext.tpc[vext.nppd][7];kdout=(int)vext.pp;
 vext.pp=(float)((kppd/((NtEnd-NtBeg+1)*vext.deltn*1e-9))*1e-6);
System.out.println(" pp-производительность Mbit/c "+vext.pp);
 System.out.println(" сигнал "+vext.nppd+" количество изменений
"+kppd);
 } } //End SignPPD
```

Основой предлагаемого подхода является однократный ввод описаний объектов в виде формализованных заданий для различных уровней анализа и конструкторско-технологического проектирования. Привычные инженеру графические документы в виде схем и сборочных чертежей должны получаться автоматически в результате интерпретации формализованных заданий и решений.

Однако вычислительная система, реализованная в соответствии с полученным описанием, под воздействием дестабилизирующих факторов

(механические воздействия, циклические изменения температур и др.) может отличаться от первоначальной. Отличия проявляются на внешних выходах системы и могут носить случайный или систематический характер.

Причиной отличий является изменение фактического состояния структуры объекта относительно синтезированного описания. В соответствии с принятым составом формализованного задания отличия могут вызываться изменением состава компонентов или выполняемыми ими функциями, а также изменением связей.

Таблица 2.8

Словарь процедур оценки производительности

Номер п/п	Имя переменной	Назначение	Тип данных	Размерность
1	PPA	Производительность аналогового сигнала	float	Мбит/с
2	PPDAC	Производительность цифроаналогового преобразователя	float	Мбит/с
3	PPD	Производительность цифрового сигнала	float	Мбит/с
4	KDOUT	Количество изменений цифрового сигнала за время анализа	int	–
5	SPPA	Сумма изменений аналогового сигнала за время анализа	float	–
6	TDAC	Время преобразования сигнала	float	Нс
7	DELT	Длительность такта	float	Нс
8	F	Тактовая частота основная	float	МГц
9	Fl	Тактовая частота работы DAC	float	МГц
10	δUst	Относительная статическая погрешность DAC	float	%
11	δUdin	Относительная динамическая погрешность DAC	float	%
12	δU	Относительная полная погрешность DAC	float	%

Процедуры анализа должны допускать реализацию изменений в описании объекта. Синтез варьируемых описаний объектов в отличие от основных можно назвать дифференциальным. Процедуры дифференциального синтеза позволяют получать описание объекта для многовариантного анализа, вносить изменения для получения нового описания из существующего. Описание отличий удобно использовать в многовариантном анализе вычислительных систем.

2.9. Формализованное задание и алгоритмы для автоматизированного анализа

Исследовательская САПР неоднородных вычислительных систем предназначена для автоматизации анализа функционирования вычислительных устройств и систем.

После освоения методов автоматизированного анализа исследуются основные узлы вычислительных устройств. Завершается цикл исследованием и оптимизацией вычислительного устройства – узла вычислительной системы более высокого уровня. Степень автоматизации анализа повышается постепенно по мере усложнения задач с обязательным усвоением принципов работы.

Схема алгоритма автоматизированного анализа приведена на рис. 2.16. Для лучшего усвоения алгоритма рекомендуется провести вручную решение системы логических уравнений в двоичном и троичном базисах до совпадения результатов.

Исследуется генерация часто используемых видов сигналов: с изменяемым временным сдвигом и количеством импульсов для устройств управления и синхронизации, с изменяемой частотой и скважностью для внешних устройств. Генерация непериодических и периодических сигналов выполняется специальной процедурой с помощью битовой строки модели сигнала. Различные варианты моделей сигналов формируются алгоритмически, получаются с помощью модели вычислительного устройства или считываются из набора данных.

Результаты генерации сигналов используются во внешних входных цепях исследуемой схемы. Рассматривается простая схема из логических элементов. В результате пользователь знакомится как с генерацией сигналов, так и с особенностями автоматизированного анализа.

В разделе представления системы или устройства производится покомпонентное описание всех вариантов и развивается объектно-компонентная методология реализации систем для всех приложений [30, 31, 33, 39, 40]. Раздел состоит из обобщенных компонентов, конкретизируемых абстрактным или конкретным типом компонента.

Многовариантный анализ вычислительного устройства с простейшим способом управления внешними воздействиями – считыванием битовых строк из подготовленного раздела библиотеки – используется для контроля узлов вычислительных систем. Схема алгоритма многовариантного анализа приведена на рис. 2.17 и реализуется монитором IPRINT, алгоритм многовариантного анализа с выводом аналоговых сигналов – монитором APRINT, с выводом аналоговых массивов – монитором APRINTM.

Многовариантный анализ можно выполнить и с помощью простого монитора со считыванием данных из одного или различных разделов библиотеки за счет усложнения только задания. Модель реализуется с помощью внутренней процедуры, в которой должно быть предусмотрено исключение запрещенных состояний сигналов.

Рис. 2.16. Блок-схема алгоритма одновариантного анализа

Далее исследуются макромодели и микромодели компонентов и узлов. Например, при анализе мультиплексора составляются макромодель и микромодель из логических элементов, модели исследуют в идентичных усло-

виях, результаты сравнивают и заносят в таблицу цепей, каналов и результатов. В качестве примеров используют одноразрядные и многоразрядные мультиплексоры, компараторы адреса, преобразователи кодов.

Рис. 2.17. Блок-схема алгоритма многовариантного анализа

Анализ устройств с изменяемой структурой выполнен в зависимости от неопределенности внешней среды [17, 27, 45, 67, 87, 89, 97, 155]. Структура устройства может изменяться статически (путем смены модулей, выполняющих различные функции) или динамически (в зависимости от условий и ситуаций). Статические изменения структуры устройства или системы можно внести в описание заранее и проанализировать все варианты. Динамическое изменение структуры реализуется аппаратно (с помощью управляемых коммутационных элементов) и программно (в разделе описания соединений). Например, можно описать условное соединение, соответствующее разрыву или замыканию цепей. Условия временные, сочетания значений сигналов или сравнения сигналов можно реализовать с помощью операторов условий и выбора универсальных языков программирования [7, 4–12, 119].

Можно использовать средства автоматической оценки состояний всех сигналов. Это позволяет оставить на диаграмме самые информативные сигналы.

Средства сравнения сигналов в заданных временных интервалах можно использовать для оценки идентичности макромоделей и микромоделей функционирования вычислительных устройств. Результаты оценки состояний и сравнения сигналов выводятся в табличной форме в соответствии с уровнем результатов, приведенным в табл. 2.9.

Таблица 2.9

Таблица уровня результатов

Значение уровня результата (LRES)	Наименование дополнительно представляемых результатов
–5–0	Символьный файл состояний для графического отображения
–1, 1	Временная диаграмма работы и оценка ресурсов
–2, 2	Таблица сравнения предполагаемых и фактических цифровых сигналов
–3, 3	Таблица сравнения цифровых и аналоговых сигналов. Вывод сообщений для отладки интерфейсов
–4, 4	Вывод таблицы состояния цифровых сигналов
–5, 5	Оптимизация

Анализируется структура на базе микроЭВМ без операционных узлов. Обработка информации производится программно, и производительность такой системы меньше, чем у систем с аппаратной поддержкой. Эффективность данной структуры будет расти с увеличением степени интеграции компонентов – однокристальных микроЭВМ.

Многоканальная цифровая обработка сигналов рассматривается на базе микроЭВМ с различной аппаратной поддержкой. Минимальная аппаратная поддержка используется в структуре с общим операционным узлом и входным многоканальным коммутатором. Оперативная обработка сигналов по различным каналам производится общим операционным узлом в режиме разделения времени. Результаты оперативной обработки записываются в памяти микроЭВМ. Окончательная обработка и формирование результатов в удобной для пользователя форме производятся с помощью микроЭВМ.

В последующих разделах используются структуры с постепенным увеличением аппаратной поддержки. Приведены примеры автоматизированного анализа структур с параллельной обработкой сигналов по всем каналам. Завершающей является конвейерная структура.

Для облегчения структурной оптимизации вычислительных устройств обеспечивается автоматическая оценка ресурсов. Оценку производительности выполняет пользователь, а значения основных критериев эффективности вычисляются автоматически.

В основу методического и программного обеспечения исследовательской САПР положен принцип однократного описания технического решения и минимум затрат труда инженера или студента для использования в различных приложениях. В качестве приложений мы рассматриваем поведение вычислительного устройства или системы, техническое проектирование модулей ЭВМ в различных САПР: ПРАМ5.3, PCAD, ALTIUM DESIGNER, EAGLE, ORCAD, CADDY и CATIA [11–110].

Интерфейс с системами автоматизации конструкторского проектирования в основном выполняется автоматически. В качестве САПР конструкторского уровня используются отечественная система ПРАМ5.3 и зарубежные системы PCAD, ALTIUM DESIGNER, CADDY и ORCAD. Интерфейс обеспечивает преобразование формализованных заданий с описанием одного и множества вариантов вычислительных устройств. Информационное обеспечение интерфейса для конкретизации формализованных заданий реализовано с помощью таблиц.

Созданы представления объектно-компонентных моделей множества технических решений вычислительных систем на различных уровнях абстракции на языках высокого уровня в различных синтаксических средах и алгоритмы их построения для различных приложений. В исследовании развивается объектно-компонентная методология описания систем для всех приложений. Новое совмещение в одном описании вычислительных систем на различных уровнях абстракции и алгоритмы их преобразования в различных синтаксических средах подтверждает преимущество семанти-

ки перед синтаксисом, уменьшает объем описаний, позволяет обеспечить концептуальное единство и снизить возможность ошибки среды.

Проектирование производится при неполной информации об объекте и внешней среде. Информация дополняется на каждой итерации проектирования объекта. Анализ различия предполагаемых и фактических результатов и критериев эффективности позволяет формировать правила синтеза объекта. Основное внимание следует уделить системам с прогнозом интервала сигналов или с моделью объекта. С накоплением знаний о сигналах и объектах уменьшается поток входной информации и увеличивается поток прогнозных оценок сигналов.

На схемотехническом уровне определяются энергия переключения и сквозные токи, температурный режим и анализ погрешности. На логическом уровне выполняется синтез и оптимизация вычислительных устройств. Проектирование неоднородной вычислительной системы производится с использованием двух уровней представления: логического и регистровых передач. Вычислительная система, включающая фрагмент сети, представляется на уровнях сервисов, сообщений или пакетов.

3. МНОГОФУНКЦИОНАЛЬНЫЕ МОДЕЛИ КОМПОНЕНТОВ

3.1. Уровни абстракции компонентов

На верхнем уровне абстракции (см. рис. 1.1) рассматривают функциональные модели компонентов в целом, которые называются макромоделями компонентов и описываются функциями выходов и переходов. В различных САПР функциональные модели компонентов представляются в различных формах и на различных языках [14, 87, 132]. Например, в п. 3.2 представлены модели компонентов в САПР COD, в [14, 132] описаны модели компонентов неоднородных систем.

На следующем уровне абстракции находятся структурные модели, в которых отражается внутренняя структура компонент. Такие модели будем называть микромоделями. На этом уровне можно оценить ресурсы.

Одна и та же структура может быть реализована с помощью различных принципов действия. Поэтому одна структура вычислительной системы может быть реализована на различных физических принципах (электроника, оптика, пневмоника), устойчивых к разным средам. Физические принципы действия элементов ЭВМ подробно описаны в [1, 6, 11, 16, 29, 59] и будут рассмотрены только при оценке ресурсов или оптимизации.

Зная принцип действия, с учетом уровня техники и технологии, можно перейти к уровню технического решения. Техническое решение должно быть выполнимым. После получения технического решения возможно изменение параметров решения с целью повышения качества изделия. Такие примеры рассмотрены в разделе параметрической оптимизации.

Определив параметры технического решения, можно перейти к конструкции компонента. Например, в зависимости от рассеиваемой кристаллом мощности ему будет соответствовать корпус. На конструкцию существенное влияние оказывает внешняя среда (условия эксплуатации). Таким образом, на верхнем уровне абстракции находятся функциональные модели компонентов, а на нижнем уровне – конструкции компонентов и их геометрия. Геометрические модели компонентов используются при конструировании [6, 134].

Модели компонентов могут использоваться на различных уровнях абстракции. Например, функциональные модели могут применяться для автоматического анализа поведения объекта; модели параметров компо-

нентов – для оценки вариантов реализации устройства. Ряд моделей компонентов используется для интерфейса с системами конструкторского проектирования, находящимися на другом уровне абстракции.

Рассмотрим функциональные модели компонентов.

3.2. Модели компонентов

Модели компонентов могут быть функциональными, геометрическими. Модель геометрии компонента может соответствовать символу на принципиальной схеме или геометрии конструктива.

Функциональные модели компонентов MFComp состоят из разделов PMF [39–41, 49]:

MFComp=<PMF1, PMF2, …, PMF5>.

Разделы описания функциональных моделей сведены в табл. 3.1.

Таблица 3.1

Разделы функциональной модели компонента

Имя раздела	Имя раздела (метка)	Описание раздела
1	FINIT	Проверка допустимости параметров, размещение структур данных и их заполнение
2	FTYP	Контроль типов компонентов и выбор возможных реализаций
3	FS	Восстановление состояний компонентов
4	FOUT	Формирование и контроль выходных сигналов
5	FEND	Освобождение ресурсов

Для каждого класса компонентов разрабатываются модели с общей структурой данных. В табл. 3.2 сведены основные классы компонентов, имена процедур и таблиц синтаксиса и семантики параметров. В табл. 3.3–3.8 приведены синтаксис и семантика основных классов компонентов.

Обязательными в общей структуре данных являются тип компонента, его номер и данные, соответствующие именам выходных и входных цепей. Сначала описываются номера выходных, а затем входных цепей. Состояние сигналов, номера которых соответствуют номерам цепей, хранятся в глобальной структуре данных. Для входных цепей процедура только считывает информацию о состоянии сигнала, а для выходных цепей процедура производит запись новых состояний.

С целью сокращения объема описания для каждого класса компонентов используется не одна, а две основные структуры данных. В первой структуре данных применяются массивы номеров входных или выходных цепей. В этом случае на порядок следования номеров цепей отсутствуют ограничения. Для второй структуры указывается только номер цепи старшего разряда (минимальное значение номера) и количество цепей. Но в этом случае номера цепей должны быть расположены в порядке возрастания, и процедура заполняет массив номеров цепей, освобождая от этого студента или инженера. Имена для первой точки входа в виде массива номеров цепей образуются из имени класса компонента и окончания NIN, а для второй точки входа – окончания MIN (минимальный).

Таблица 3.2

Модели компонентов

№ п/п	Наименование	Имя процедуры	Имя таблицы параметров
1	Генерация цифровых сигналов	SIGNAL	HSIGN
2	Управление многовариантным анализом	IPRINT	HIPRINT
3	Управление процессом анализа	APRINT	HAPRINT
4	Обобщенный логический элемент	LO	HLO
5	Триггер	MTJK	HMTJK
6	Мультиплексор	MX	HMX
7	Узел мультиплексора многоразрядного	MMX	HMMX
8	Последовательный сумматор-счетчик	MCT	HMCT
9	Последовательный сумматор реверсивный	MCTR	HMCTR
10	Параллельный интерфейсный узел	MIR	HMIR
11	Последовательный интерфейсный узел	MIS	HMIS
12	Программируемая логическая матрица	MLO	HMLO
13	Модель процессора	MPU	HMPU
14	Процессор аналоговый	MPA	HMPA
15	Пороговый элемент	PE	HPE
16	Компаратор аналоговый	CA	HCA
17	Цифроаналоговый преобразователь	DAC	HDAC
18	Аналого-цифровой преобразователь	ADC	HADC
19	Мультиплексор универсальный	MXU	HMXU
20	Мажоритарный элемент	MAJ	HMAJ
21	Регистр последовательного приближения	RPP	HRPP
22	Узел оперативной памяти	RAM	HRAM
23	Кодер и декодер	DC	HDC
24	Сумматор цифровой	SUMD	HSUMD
25	Таймер	TIM	HTIM

Таблица 3.3

Синтаксис и семантика модуля управления IPRINT

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика
1	NSMIN	DEC FIXED(3)	int	integer	Минимальный номер сигнала
2	NSMAX				Максимальный номер сигнала
3	NTMIN	DEC FIXED(6)			Номер такта начала вывода сигналов
4	NTMAX				Номер такта окончания выводов сигнала

Таблица 3.4

Синтаксис и семантика модуля управления APRINT

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика
1	NSMIN	DEC FIXED(3)	int	integer	Минимальный номер сигнала
2	NSMAX				Максимальный номер сигнала
3	NTMIN	DEC FIXED(6)			Номер такта начала вывода сигналов
4	NTMAX				Номер такта окончания вывода сигналов
5	NAMIN				Минимальный номер аналогового сигнала
6	NAMAX				Максимальный номер аналогового сигнала

Таблица 3.5

Синтаксис и семантика процедуры генерации входных сигналов (SIGNAL, SIGNALD)

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика
1	NSIGNAL	DEC FIXED(3)	int	integer	Номер сигнала
2	STRBIT	BIT (*)	BitString	Символьная строка	Битовая строка
3	NTMIN	DEC FIXED(6)	int	integer	Номер такта начала изменения сигнала
4	NTMAX				Номер такта окончания изменения сигнала

Таблица 3.6

Синтаксис и семантика параметров обобщенной модели логического элемента LO

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика	Примечания
1	TYP	CHAR (*)	char (C++) String (JAVA)	character	Тип элемента: min и max аналогового сигнала «MIMAA», для цифровых сигналов ключи приведены в примечании	Ключи: ‘&’,’|’, ‘ЛА’, ‘ЛЕ’, ‘ЛИ’, ‘ЛП’, ‘М2’
2	NEL	DEC FIXED(3)	int	integer	Номер элемента	–
3	NOUTP				Выход неинвертирующий	–
4	NOUTN				Выход инвертирующий	–
Отличие для точки входа LONIN						
5	NIN	(*) DEC FIXED(3)	IntVector (C++) Int[] (JAVA)	integer	Массив номеров входов	–
Отличие для точки входа LOMIN						
6	NMIN	DEC FIXED(3)	int	integer	Номер входа минимальный	–
7	NSIGN				Количество входов	–

Таблица 3.7

Синтаксис и семантика параметров триггера MTJK

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика	Примечания
1	TYP	CHAR (*)	Char (C++) String (JAVA)	character	Тип элемента	564ТВ1 – ключ ТВ
2	NTJK	DEC FIXED(3)	int	integer	Номер элемента	–
3	NOUTP				Выход триггера	–
4	NOUTN				Выход триггера инвертирующий	–
5	NINR				Номер входа нулевой установки	Установка по ‘11’B
6	NINS				Номер входа установки единицы	Установка по ‘11’B
7	NINC				Номер синхросигнала	Работа по ‘01’B
8	NINJ				Номер входа J	–
9	NINK				Номер входа K	–

Таблица 3.8

Синтаксис и семантика моделей мультиплексоров цифровых и аналоговых сигналов (MX и MXU)

№ п/п	Идентификатор	Тип данных PL/1	Тип данных C, C++, JAVA	Тип данных ADA	Семантика	Примечания
1	TYP	CHAR (*)	Char (C++) String (JAVA)	character	Тип мультиплексора КП предназначен для коммутации цифровых сигналов, КН – для аналоговых	564КП1 564КП2 К561КП1 К561КП2 К1561КП1 К1561КП2 К555КП: 2, 7, 12, 16 155КП: 1, 2, 5, 7 К590КН1 К590КН6 К591КН3
2	NEL	DEC FIXED(3)	int	integer	Номер элемента	–
3	NOUTP				Номер выхода	–
4	NOUTN				Номер выхода инвертирующий	–
5	NC				Номер входа разрешения	–
6	NA	(*) DEC FIXED(3)	int	integer	Массив номеров цепей передачи адреса	Размерность массива должна соответствовать типу мультиплексора
7	JIN		IntVector (C++) int[](Java)		Массив номеров входных цепей	–
Отличительные особенности параметров точки входа MXMIN						
8	NMIN	DEC FIXED(3)	int	integer	Номер входа минимальный	–
9	NSIGN				Число входных цепей	–

Структуры данных для этих точек входа должны синтаксически отличаться, а для их использования в формализованное задание должны включаться файлы описания точек входа. Например, для среды PL/1 [5, 14] файлы имеют имя, составленное из символа W и общего имени процедуры, а для C++ включается файл типа *.H. Для языка C имеется только одна точка входа.

Приведем примеры описаний для выбора точки входа процедуры моделирования универсального логического элемента:

на языке PL/1:

```
DCL (LONIN ENTRY (CHAR (*), DEC FIXED (3), DEC FIXED (3),
DEC FIXED (3), (*)DEC FIXED (3)), LOMIN ENTRY (CHAR(*),
DEC FIXED (3), DEC FIXED (3), DEC FIXED (3), DEC FIXED (3),
DEC FIXED (3)));
DCL LO GENERIC (LONIN WHEN (....), LOMIN WHEN (....));
```

на C++:

```
void  LO (char *typ, int nel, int noutp, int noutn, IntVector& jin);
void  LO (char *typ, int nel, int noutp, int noutn, int ninmin, int nsign);
```

на JAVA:

```
public static void LO (String typ, int nel, int noutp, int noutn, int ninmin,
int nsign);
public static void LO (String typ, int nel, int noutp, int noutn,  int[] mnin);
```

Семантика параметров приведена в табл. 3.6.

Модель компонента состоит из ряда разделов и наряду с выполнением основных функций выдает диагностические сообщения.

В первом разделе в соответствии с классом компонентов объявляются структуры данных и, независимо от используемой точки входа, заполняются массивы номеров входных и выходных цепей. Желательно объявление массивов произвольной размерности с последующим определением размеров фактически переданных массивов по каждому измерению.

Во втором разделе процедуры (FTYP) проверяется допустимость типа компонента и определяются тип и параметр выполняемой функции. Для недопустимого типа компонента выдается сообщение об ошибке. После проверки типа компонента оператор с меткой FPAR запоминает тип и номер компонента путем вызова функции FPAR. Тип и номер компонента в дальнейшем используются для автоматической оценки параметров устройства с помощью процедуры FPARSUM. Информация о параметрах компонентов хранится в таблице UIPCAD.DBT. В этой таблице также имеются параметры конкретных типов компонентов, для которых известны потребляемая мощность, стоимость, площадь, масса. Абстрактные типы компонентов допускают произвольную разрядность и произвольное число входов-выходов, но не имеют конкретных параметров. Абстрактные типы компонентов заменяются конкретными при определении элементной базы.

Третий раздел (FS) служит для оценки состояния компонента, а четвертый раздел (FOUT) – для вычисления новых значений выходных сигналов.

В пятом разделе (FF) оценивается допустимость полученных выходных сигналов, и в случае необходимости запрещенные значения заменяются на

допустимые. Например, для троичной модели уровни сигналов обозначены: C0 – уровень нуля, C1 – уровень единичного состояния; CF – фронт; CZ – обозначение запрещенного состояния. Использование таких обозначений облегчает переход из одной языковой среды в другую. Имя завершающего раздела – FEND.

3.3. Представление моделей в САПР COD

Пример процедуры обобщенного логического элемента для цифровых и аналоговых сигналов на языке PL/1 с общим именем LO и именем точки входа LONIN для объединенных неупорядоченных цепей (массив номеров цепей) и именем точки входа LOMIN для объединенных упорядоченных в порядке возрастания номеров цепей с параметрами номера цепи старшего разряда и количества входных цепей приведен ниже:

```
LONIN:    PROC(TYP,NEL,NOUTP,NOUTN,JIN);
          DCL(NOUTP,NOUTN,NEL,I,J,II)DEC FIXED(3);
          DCL(JIN(*),NSIGN,MNIN,NMIN)DEC FIXED(3);
          DCL FPAR ENTRY;  DCL TYP CHAR(*);
          DCL NEWI BIT(2) INIT('11'B), FB BIT(4) INIT('1111'B);
          DCL NIN(MNIN)CTL DEC FIXED(3); MNIN=DIM(JIN,1);
          ALLOCATE NIN;NIN=JIN;GOTO FTYP;
LOMIN:    ENTRY(TYP,NEL,NOUTP,NOUTN,NMIN,NSIGN);
          MNIN=NSIGN;     ALLOCATE NIN;
          DO II=1 TO NSIGN;NIN(II)=II+NMIN-1;END; /*LOMIN*/
          DCL NTYP INIT(13) DEC FIXED(3), MTYP(NTYP) CHAR(10)
VAR INIT    ('&','AND','ЛА','ЛИ','|','OR','ЛЕ','ЛЛ','M2','=','ИП','#','MAJ');
FTYP: /* АНАЛОГОВАЯ ЧАСТЬ */ ;
  DCL ( AMAX, AMIN ) DEC FLOAT;DCL IA DEC FIXED(3);
  IF INDEX(TYP,'MIMAA')>0 THEN DO;
  AMIN=AS(NIN(1)).ANEX; AMAX=AS(NIN(1)).ANEX;
  DO IA=1 TO MNIN; AMAX=MAX(AMAX,AS(NIN(IA)).ANEX);
          AMIN=MIN(AMIN,AS(NIN(IA)).ANEX);END;
  /* ВЫВОД РЕЗУЛЬТАТА */
  AS(NOUTN).ANEW=AMIN;AS(NOUTP).ANEW=AMAX;
END;/* MIMAA  */
ELSE DO;/* LO  */
      DO J=1 TO NTYP ;   IF INDEX(TYP,MTYP(J))>0 THEN DO;
          SELECT(J);
```

```
        WHEN (1,2,3,4) DO NEWI=C1;FB='0001'B;END;
        WHEN (5,6,7,8) DO NEWI=C0;FB='0111'B;END;
        WHEN (9)       DO NEWI=C0;FB='0110'B;END;
        WHEN (10,11)  DO NEWI=C0;FB='1001'B;END;
        WHEN (12,13)  DO; CALL MAJ(TYP,NEL,NOUTP,NOUTN,NIN);
                GOTO FEND;END;
                OTHER  DO;
                PUT SKIP(1) EDIT('ТИП ',TYP,' ОТСУТСТВУЕТ ЦРОЦ LO
                NEL=',NEL)  (A,A,A,F(3)); ERC=MAX(ERC,12);
                GOTO FEND; END;    END;/*SELECT*/;END;END;
FOUT:   /* FPAR */  IF NT=0&PPT THEN CALL FPAR(TYP,NEL);
                DO I=1 TO MNIN;
                IF NEX(NIN(I))=CZ THEN NEX(NIN(I))=CF;
                NEWI=BOOL(NEWI,OUT(NIN(I)).NEX,FB);END;
                IF NEWI=CZ THEN NEWI=CF;NEW(NOUTP)=NEWI;
                IF NEWI=CF THEN NEW(NOUTN)=CF; ELSE
                NEW(NOUTN)=^NEWI;
                FEND:FREE NIN;       END LONIN;
```

Пример процедуры обобщенного логического элемента на языке C++:

```
#include <vextcpp.h>
void LO(char *typ,int nel,int noutp,int noutn,int nmin,int nsign)
{ IntVector jin (nsign);
  for (int i=0; i<nsign; i++)  jin[i] = i + nmin;
LO(typ,nel,noutp,noutn,jin); }
void LO(char *typ,int nel,int noutp,int noutn,IntVector& jin)
{ int nsign,nmin,i,j;
unsigned int newi;
int mnin=jin.Dim();
IntVector nin(mnin);
static char *typs[] = { "&","ЛА","ЛИ","1","ЛЕ","ЛЛ","М2","ИП"};
const int NTYP = sizeof(typs)/sizeof(*typs);
static struct logic
        { unsigned ex1 : 2;
        unsigned ex2 : 2;
        unsigned ex3 : 2;
        unsigned ex4 : 2;
        }   ft[NTYP]  =
{ {C0,C0,C0,C1}, {C0,C0,C0,C1}, {C0,C0,C0,C1}, {C0,C1,C1,C1},
  {C0,C1,C1,C1}, {C0,C1,C1,C1}, {C0,C1,C1,C0}, {C1,C0,C0,C1} };
```

```
      unsigned int newo[NTYP] =
            { C1,C1,C1,C0,C0,C0,C0,C0 };
      nin = jin;
FTYP:  // Begin MINMAX
        if ( strstr(typ,"MIMAA")!=NULL )
 {  int ia; float amin,amax;
  amin = as[nin[0]].ANex; amax = as[nin[0]].ANex;
   for (ia=0; ia<mnin; ia++)
    {  if (as[nin[ia]].ANex > amax) amax = as[nin[ia]].ANex;
      if (as[nin[ia]].ANex < amin) amin = as[nin[ia]].ANex;  }
  as[noutp].ANew = as[noutp].ANex = amax;
  as[noutn].ANew = as[noutn].ANex = amin;
 } // end MINMAX
 else {     for ( j=0; j<NTYP ; j++)
            if (strstr( typ,typs[j] ) != NULL )
                  newi = newo[j];
              break;   }
if ( j>>=NTYP )
      { fprintf(stderr," В процедуре LO отсутствует тип  %s,
        № элемента:        %d\n",      typ,nel);
        return;   }
FPAR:  if (nt == 0 && ppt) { Fpar (typ,nel); }
FOUT:  for ( i=0 ; i<mnin ; i++ )
     { if ( out[nin[i]].Nex == CZ )
       out[nin[i]].Nex = CF;
       newi = (
         (ft[j].ex1 & ~newi & ~out[nin[i]].Nex)  |
         (ft[j].ex2 & ~newi &  out[nin[i]].Nex)  |
         (ft[j].ex3 &  newi & ~out[nin[i]].Nex)  |
         (ft[j].ex4 &  newi &  out[nin[i]].Nex)   );  }
        newi = newi==CZ ? CF:newi;
      out[noutp].New = newi;
      out[noutn].New = (newi == CF) ? CF : ~newi;
       }  } // end of LO
```

Пример процедуры обобщенного логического элемента на языке Java:

```
public static void LO(String typ,int nel,int noutp,int noutn,int nmin,int nsign)
  {int jin[] = new int[nsign];
 for (int i = 0; i < nsign; i ++)   jin[i] = i + nmin;
```

```
   LO(typ, nel, noutp, noutn, jin);   }
public static void LO(String typ, int nel, int noutp, int noutn, int jin[])
   {int nsign, nmin, i, j;    byte newi = 0;
    int mnin = jin.length;   int nin[] = new int[mnin];
    String[] typs = {"&","ЛА","ЛИ","|","ЛЕ","ЛЛ","М2","ИП"};//char *typs[] =
{...};
    logic ft = new logic(8);
    ft.Set(0,vext.C0,vext.C0,vext.C0,vext.C1);
ft.Set(1,vext.C0,vext.C0,vext.C0,vext.C1);
    ft.Set(2,vext.C0,vext.C0,vext.C0,vext.C1);
ft.Set(3,vext.C0,vext.C1,vext.C1,vext.C1);
    ft.Set(4,vext.C0,vext.C1,vext.C1,vext.C1);
ft.Set(5,vext.C0,vext.C1,vext.C1,vext.C1);
    ft.Set(6,vext.C0,vext.C1,vext.C1,vext.C0);
ft.Set(7,vext.C1,vext.C0,vext.C0,vext.C1);
byte newo[] =
{vext.C1,vext.C1,vext.C1,vext.C0,vext.C0,vext.C0,vext.C0,vext.C0}; nin = jin;
     //FTYP:
if (typ.indexOf("MIMAA") == 0)
 {  int ia; float amin,amax;
     amin = vext.as.ANex[nin[0]]; amax = vext.as.ANex[nin[0]];
       for (ia=0; ia<mnin; ia++)
        { amax = Math.max(amax, vext.as.ANex[nin[ia]]);
          amin = Math.min(amin, vext.as.ANex[nin[ia]]);         }
    vext.as.ANew[noutn] = vext.as.ANex[noutn] = amin;
    vext.as.ANew[noutp] = vext.as.ANex[noutp] = amax;   }
else    {    for (j = 0; j < 8; j ++)
         if (typ.lastIndexOf(typs[j]) != -1 )
           { newi = newo[j];     break;    }
  if ( j >= 8 )
    { System.err.println("Ошибка!!Error!! В процедуре LO отсутствует тип
"+ typ+",N° элемента : "+nel);  return;    }
  //FPAR:
  if (vext.nt == 0 && vext.ppt)  control.Fpar(typ, nel);
  //FOUT:
  for (i = 0; i < mnin; i ++)
      {if (vext.out.Nex[nin[i]] == vext.CZ)
       vext.out.Nex[nin[i]] = vext.CF;
       newi = (byte)((ft.ex1[j] & (newi^3) & (vext.out.Nex[nin[i]]^3))|
  (ft.ex2[j] & (newi^3) &  vext.out.Nex[nin[i]])   |  // C0^3 == C1;   T1^3 == C0
```

```
                (ft.ex3[j] &  newi & (vext.out.Nex[nin[i]]^3))   |
                (ft.ex4[j] &  newi &  vext.out.Nex[nin[i]])   ); }
  newi = (newi == vext.CZ) ? vext.CF : newi;
  vext.out.New[noutp] = newi;
  if (newi == vext.CF) vext.out.New[noutn] = newi;
  else vext.out.New[noutn] = (byte)(newi^3);
 } } // end of LO
```

3.4. Модели компонентов обеспечения интерфейсов САПР COD с системой PCAD

Возможные структуры интерфейсов САПР COD верхнего уровня с промышленными САПР приведены на рис. 4.1. Использование приведенных на рис. 4.1 уровней позволяет автоматизировать преобразование описаний множества проектных решений на языках высокого уровня [1, 6, 13, 14] в множество схем для промышленных САПР [2, 8, 28, 29, 32, 33, 103, 120, 121, 160]. Результаты исследований многократных реализаций интерфейсов САПР COD с различными промышленными САПР, классификации функций и данных позволили обеспечить общие модели компонентов для различных приложений. В качестве приложений могут применяться как САПР PCAD, ALTIUM DESIGNER, ORCAD, CADDY, так и среда виртуальной реальности (VRML2) [34] или стандартное описание вычислительных систем в форматах ISO10303, EDIF и PDIF [10, 29, 33, 81, 75, 79, 116, 133].

Интерфейс обеспечен для базового множества моделей компонентов, включающего цифровые интегральные схемы, аналого-цифровые и цифроаналоговые компоненты. Необходимость синтеза моделей компонентов возникает для новых компонентов.

В качестве примера моделей компонентов для табличного интерфейса УИ САПР PCAD далее приведена модель ЦАП с входными цифровыми сигналами и выходным аналоговым сигналом. Модель ЦАП имеет две точки входа с массивом входных цепей и с минимальным номером цепи старшего разряда с указанием количества входов. Основная модель представлена с массивом входов, а вторая преобразуется к ее формату. Аналоговые и цифровые цепи могут иметь одинаковые номера, так как отличаются префиксом NA и ND.

Примером составного функционального компонента, которому соответствует множество реальных компонентов, является реверсивный счетчик

К561ИЕ11. При этом создаются номера цепей входов и выходов таким образом, чтобы номера цепей не совпадали. Например, для счетчика они увеличены на 100. Приведем текст интерфейсных моделей ЦАП с различными стилями описаний на языке PL/1:

```
DACMIN:PROC (TYP,NEL,NOUTA,NIN1,NINNEL);
      DCL TYP CHAR(*);
      DCL (NEL,NOUTA,NIN1,NINNEL) FIXED DEC(3);
      DCL I FIXED BIN;
      DCL JIN(NINNEL) FIXED DEC(3);
      DO I=0 TO NINNEL-1;
        JIN(I+1)=NIN1+I;
      END;
      CALL DACNIN(TYP,NEL,NOUTA,JIN);
  END DACMIN;
DACNIN:procedure(TYP,NEL,NOutA,Jin);
      declare TYP char(*);
      declare (NEL,NOutA) fixed dec(3);
      declare Jin(*) fixed dec(3);
      declare i fixed dec(3);
      declare CTYP varying char(20);
      declare NElm fixed dec;
      NElm=ElmCopy(TYP,'DAC');
      if NElm>0 then begin;
        call NetAdd(NElm,'NOUTA','NA',NOutA);
        do i=0 to Hbound(Jin)-1;
          CTYP='NIN'||trim(char(i),' ');
          call NetAdd(NElm,CTYP,'ND',Jin(i+1));
          end;
      end;
  END DACNIN;
```

При вызове модели компонента производится его поиск в таблице соответствия (uipcad.dbm, uipcad.dbn или uipcad.dbt). В случае успешного поиска компонента запись копируется в таблицу варианта схемы. После добавления записи компонента нужно к требуемым выводам подключить цепи. Это выполняется с помощью функции добавления цепи (NetAdd), которой передаются параметры: номер элемента, имя вывода в УИ САПР, тип и номер цепи. Аналоговой цепи соответствует тип NA, а цифровой – ND. Модели компонентов находятся в файле COMP и используются для командного, текстового и табличного интерфейсов. Автоматическое преоб-

разование описаний может производиться с использованием статических и динамических библиотек интерфейса. Для формирования динамической библиотеки объединенного интерфейса в среде Windows должен быть подготовлен файл описания типа def с помощью команды GENDEFW.BAT и командный файл ipwgi.bat:

```
REM Создание DLL-библиотеки с LIB файлом импорта.
if exist ipgi.obj del ipgi.obj
pli lpgi.pli (m s dllinit default(linkage(system)) langlvl(saa2))
if exist ipwgi.lib del ipwgi.lib
ilib /GI:ipwgi lpgi.obj ipwgi.def
if exist ipwgi.dll del ipwgi.dll
ilink /OUT:ipwgi.dll /DLL lpgi.obj ipwgi.exp > ipwgi.err
```

Ниже приведен текст моделей ЦАП и счетчика для интерфейса на языке C++:

```
void MCTR(char* _FAR typ,int nel,int Lct,IntVector& _FAR  JOut,int NOutP,IntVector& _FAR  JIn,int NinP,IntVector& _FAR  Kin)
{    int Count,Fin,FOut,i, inp, outp;
 TSch *tmp;
 char CName[20];
 if((Lct%4)==0)
 {     Count=Lct/4;
  //если длина счетчика больше 4, то номера корпусов +100
  if(Count>1)
  {   if(NCR==0) NCR=ns+1;  nel=nel+100;  inp=NinP; outp=NOutP; }
 }
 else  {
   if(strcmp(CodSrch("LANG"),"eng")==0)
      printf("The number of categories of the counter is incorrect!!!\n");
   else    printf("Число разрядов счетчика неверно!!!\n");   return;
      }
 for(i=0;i<Count;i++)
 {
  tmp=ElmCopy(typ,nel);
  if(tmp!=NULL)
  {   Fin=i*4;  FOut=i*4;
   NetAdd(tmp,"NOUT0","ND",JOut[FOut]);
   NetAdd(tmp,"NOUT1","ND",JOut[FOut+1]);
   NetAdd(tmp,"NOUT2","ND",JOut[FOut+2]);
```

```
NetAdd(tmp,"NOUT3","ND",JOut[FOut+3]);
NetAdd(tmp,"NIN0","ND",JIn[Fin]);
NetAdd(tmp,"NIN1","ND",JIn[Fin+1]);
NetAdd(tmp,"NIN2","ND",JIn[Fin+2]);
NetAdd(tmp,"NIN3","ND",JIn[Fin+3]);
NetAdd(tmp,"NOUTN0","ND",JOut[FOut]);
NetAdd(tmp,"NOUTN1","ND",JOut[FOut+1]);
NetAdd(tmp,"NOUTN2","ND",JOut[FOut+2]);
NetAdd(tmp,"NOUTN3","ND",JOut[FOut+3]);
NetAdd(tmp,"NINN0","ND",JIn[Fin]);
NetAdd(tmp,"NINN1","ND",JIn[Fin+1]);
NetAdd(tmp,"NINN2","ND",JIn[Fin+2]);
NetAdd(tmp,"NINN3","ND",JIn[Fin+3]);
if(Count>1)
{       if(i==Count-1)  outp=NOutP;
  else   { outp=NCR+i; NCR++; }
  NetAdd(tmp,"NOUTP","ND",outp);   //NDC
  NetAdd(tmp,"NOUTNP","ND",outp);  //NDC
}
else  {
  NetAdd(tmp,"NOUTP","ND",NOutP);  //NDC
  NetAdd(tmp,"NOUTNP","ND",NOutP); //NDC
       }
NetAdd(tmp,"NINP","ND",NinP);
NetAdd(tmp,"NINNP","ND",NinP);
if(strstr(typ,"IE11")!=NULL || strstr(typ,"ИЕ11")!=NULL)
{
 NetAdd(tmp,"NINR","ND",Kin[0]);
 NetAdd(tmp,"NINB","ND",1);
 NetAdd(tmp,"NINNR","ND",Kin[0]);
 NetAdd(tmp,"NINNB","ND",1);
}
else  {
 NetAdd(tmp,"NINR","ND",0);
 NetAdd(tmp,"NINB","ND",Kin[0]);
 NetAdd(tmp,"NINNR","ND",0);
 NetAdd(tmp,"NINNB","ND",Kin[0]);
       }
NetAdd(tmp,"NINC","ND",Kin[1]);
NetAdd(tmp,"NINV","ND",Kin[2]);
NetAdd(tmp,"NINO","ND",Kin[3]);
```

```
     NetAdd(tmp,"NINNC","ND",Kin[1]);
     NetAdd(tmp,"NINNV","ND",Kin[2]);
     NetAdd(tmp,"NINNO","ND",Kin[3]);
    }
    if(Count>1)   { nel=nel+1; NinP=outp; }
  }
}
void DAC(char* _FAR typ,int nel,int NOutA,IntVector& _FAR  JIn)
{   int i;   char CName[20];
  TSch *tmp;
  tmp=ElmCopy(typ,nel);
  if(tmp!=NULL)
  {
    NetAdd(tmp,"NOUTA","NA",NOutA);
    for(i=0;i<JIn.Dim();i++)
    {
      sprintf(CName,"NIN%u",i);
      NetAdd(tmp,CName,"ND",JIn[i]);
      sprintf(CName,"NINN%u",i);
      NetAdd(tmp,CName,"ND",JIn[i]);
    }
  }
}
void DAC(char* _FAR typ,int nel,int NOutA,int Nin1,int NinNum)
{   int i;   IntVector JIn(NinNum);
  for(i=0;i<NinNum;i++) JIn[i]=Nin1+i;
  DAC(typ,nel,NOutA,JIn);
}
```

Обобщенный интерфейс САПР COD с конкретной САПР предполагает использование единых моделей компонентов верхнего уровня в файле COMP для различных САПР и видов интерфейсов. Однако возможно использование других моделей компонентов среднего уровня для командного интерфейса. Они отличаются тем, что каждая из них выводит в командный файл команды включения символа компонента и именования цепей. Аналоговые цепи имеют префикс NA, а цифровые – ND. Внутри моделей цифровые цепи имеют положительные номера, аналоговые – отрицательные номера. Примеры моделей компонентов ЦАП и счетчика на языке C++ для командного интерфейса приведены в [27, 31].

Модели компонентов табличного и командного интерфейсов на языке C++ транслируются для всех операционных систем в подкаталогах

LCP4T и LCP4I, загружаются в библиотеки lcdp4t (uipcad4t) и lcdp4i (uipcad4i) для версии PCAD4.5 и в библиотеки lcdp8t (uipcad8t) и lcdp8i (uipcad8i) для версии PCAD8.5. Для других языков программирования и операционных систем имена библиотек и командных файлов составляются в соответствии с табл. 3.9. Библиотеки моделей компонентов рекомендуется создавать с помощью командных файлов, примеры которых приведены ниже.

В операционных системах Windows и OS/2 создаются как статические библиотеки типа lib, так и динамические библиотеки типа dll и дополнительные библиотеки типа lib. Так как типы статических и дополнительных библиотек совпадают, то в соответствии с табл. 4.4 имена статических библиотек начинаются с символа l, а динамические и дополнительные – с символа i. Для создания динамических и дополнительных библиотек необходимо наличие файла типа def, который создается с помощью командных файлов gendefo.cmd для OS/2 и gendefw.bat для Windows.

Файл gendefo.cmd для объединенного интерфейса:

```
REM  Создание *.DEF-файла к объединенному интерфейсу LCGI
REM  1. Компиляция исходных файлов
icc  /Id:\codos\include; /Ge- /C  mcfz.cpp mcimport.cpp tsch.cpp comp.cpp table.cpp codread.cpp mcoi.cpp mcv2.cpp mcp8t.cpp mcp4t.cpp mcp8i.cpp control.cpp pcad4.cpp sch4.cpp slb4.cpp sym4.cpp pcad8.cpp sch8.cpp slb8.cpp sym8.cpp mcr.cpp mcai.cpp > icogi.err
REM  2. Создание *.DEF файла
cppfilt /b /q /p mcfz.obj mcimport.obj tsch.obj comp.obj table.obj codread.obj mcoi.obj mcv2.obj mcp8t.obj mcp4t.obj mcp8i.obj control.obj pcad4.obj sch4.obj slb4.obj sym4.obj pcad8.obj sch8.obj slb8.obj sym8.obj mcr.obj mcai.obj > icogi.de
REM  Этот фрагмент добавляет LIBRARY EXPORTS автоматически.
if exist *.obj del *.obj
if not exist icogi.de goto end
echo LIBRARY EXPORTS >temp
copy temp+icogi.de  icogi.def
del temp
del icogi.de
:end
```

Файл gendefw.bat для объединенного интерфейса:

```
REM  создание *.DEF-файла к объединенному интерфейсу LCGI
REM  1. Компиляция исходных файлов
```

```
icc  /Id:\codos\include; /Ge- /C mcfz.cpp mcimport.cpp tsch.cpp
comp.cpp table.cpp codread.cpp mcoi.cpp mcv2.cpp mcp8t.cpp mcp4t.cpp
mcp8i.cpp control.cpp pcad4.cpp sch4.cpp slb4.cpp sym4.cpp pcad8.cpp
sch8.cpp slb8.cpp sym8.cpp mcr.cpp mcai.cpp > icwgi.er
REM  2. Создание *.DEF-файла
cppfilt /b /q /p mcfz.obj mcimport.obj tsch.obj comp.obj table.obj
codread.obj mcoi.obj mcv2.obj mcp8t.obj mcp4t.obj mcp8i.obj control.obj
pcad4.obj sch4.obj slb4.obj sym4.obj pcad8.obj sch8.obj slb8.obj sym8.obj
mcr.obj mcai.obj > icwgi.de
REM  Этот фрагмент добавляет LIBRARY EXPORTS автоматически.
if exist *.obj del *.obj
if not exist icwgi.de goto end
echo LIBRARY EXPORTS >temp
copy temp+icwgi.de  icwgi.def
del temp
del icwgi.de
:end
```

В результате анализа реализованных интерфейсов УИ САПР с промышленными САПР была показана возможность построения обобщенных интерфейсов с группой САПР. Обобщенный интерфейс позволяет использовать множество моделей компонентов для интерфейса с промышленными САПР.

Примеры моделей компонентов ЦАП (DAC) и реверсивного счетчика (MCTR) для обобщенного интерфейса на языке JAVA приведены ниже:

```
//=     DAC
public static void DAC (String typ,int nel,int nout,int inBeg, int nSize)
{int[] jin = new int[nSize];
for (int i=0; i < nSize; i++)      jin[i] = inBeg + i;
DAC(typ,nel,nout,jin);}
public static void DAC (String typ,int nel,int nout,int[] jin)
{       {       nel=NEL++;
              vexttsch.tsch.ElmCopy(typ, nel);
              vexttsch.tsch.NetAdd(vext.Curr,"NOUTA","A",nout);
              for (int i=0;i<jin.length;i++)
              vexttsch.tsch.NetAdd(vext.Curr,"NIN"+i,"D",jin[i]);
       };
};//end DAC
//=     MCTR
public static void MCTR (String typ, int nel, int lct, int noutmin, int noutp,
int ninmin, int ninp,int ninr, int ninc, int ninv, int nino, int ninb)
```

```
{int[] jin = new int[lct]; int[] jout=new int[lct]; int[] kin=new int[4];
for (int i=0;i<lct;i++) jin[i]=ninmin+i;
for (int i=0;i<lct;i++) jout[i]=noutmin+i;

if (typ.regionMatches(true,0,"IE11",0,4)||
   typ.regionMatches(true,0,"++11",0,4)||
   typ.regionMatches(true,0,"ИЕ11",0,4)   )
{      kin[0]=ninr;  }
else {      kin[0]=ninb; };
kin[1]=ninc; kin[2]=ninv; kin[3]=nino;
MCTR(typ,nel,lct,jout,noutp,jin,ninp,kin);
};
public static void MCTR (String typ,int nel,int lct,int[] jout,
                    int noutp,int[] jin,int ninp,int[] kin)
{   int outp;
          {   int stnum=lct/4;
          if (Math.IEEEremainder(lct,4)!=0.0)stnum++;
          for (int k=0;k<stnum;k++)
          {   nel=NEL++;
 vexttsch.tsch.ElmCopy(typ, nel);
             int NUM=1;
             if (typ.regionMatches(true,0,"IE11",0,4)||
                typ.regionMatches(true,0,"++11",0,4)||
                typ.regionMatches(true,0,"ИЕ11",0,4)    )
             {      vexttsch.tsch.NetAdd(vext.Curr,"NINR","D",kin[0]);
                    vexttsch.tsch.NetAdd(vext.Curr,"NINB","D",1);
             }
             else  {     vexttsch.tsch.NetAdd(vext.Curr,"NINB","D",kin[0]);
                    vexttsch.tsch.NetAdd(vext.Curr,"NINR","D",0);
             }
             vexttsch.tsch.NetAdd(vext.Curr,"NINR","D",kin[0]);
             vexttsch.tsch.NetAdd(vext.Curr,"NINC","D",kin[1]);
             vexttsch.tsch.NetAdd(vext.Curr,"NINV","D",kin[2]);
             vexttsch.tsch.NetAdd(vext.Curr,"NINO","D",kin[3]);
             if(k==(stnum-1)) outp=noutp; else outp=vext.ns+k+1;
             vexttsch.tsch.NetAdd(vext.Curr,"NINP","D",ninp);
             vexttsch.tsch.NetAdd(vext.Curr,"NOUTP","D",outp);
             for (int i=0;k*4+i<jout.length && i<4;i++)
vexttsch.tsch.NetAdd(vext.Curr,"NOUT"+i,"D",jout[k*4+i]);
             for (int i=0;k*4+i<jin.length && i<4;i++)
                    vexttsch.tsch.NetAdd(vext.Curr,"NIN"+i,"D",jin[k*4+i]);
```

```
            ninp=outp;
          };
        };
};//end MCTR
```

При вызове модели компонента добавляется запись в список элементов таблицы варианта схемы. После успешного добавления записи компонента нужно к требуемым выводам подключить цепи. Это происходит с помощью функции добавления цепи (NetAdd), которой передаются параметры: номер цепи, номер компонента, имя вывода в УИ САПР, тип вывода в конкретной САПР. Модели компонентов находятся в файле COMP и используются для всех интерфейсов.

Эффективность обобщенного интерфейса объясняется снижением объемов статических и динамических библиотек, архивных файлов пакетов на языке JAVA, а также использованием общих моделей компонентов, функций работы с таблицами и файлами.

3.5. Автоматизация формирования библиотек моделей компонентов САПР PCAD

Библиотеки символов используются при синтезе функциональных и принципиальных схем, а библиотеки конструктивов – при конструкторском проектировании модулей ЭВМ.

Создание библиотек компонентов отличается трудоемкостью и ответственностью, поэтому автоматизация создания библиотек является актуальной задачей [29, 33, 45, 46, 120–122].

В работе [134] предложено преобразование текстового описания символов компонентов в графический формат без учета выполняемых функций типов выводов и их эквивалентности. В работе автора [33] описывается способ создания символов компонентов с помощью специального файла текстового описания типа def и с помощью таблицы соответствия uipcad.dbt, используемой для интерфейса исследовательской САПР с другими системами. Файл описания компонентов типа def создается в текстовом редакторе и преобразуется в pdf-формат, из которого утилитой PDIFIN формируется символьный образ типа sym. Базовые параметры описания компонента формируются с помощью символов установки среды SYM.sym и PRT.prt, соответствующей версии САПР PCAD. Базовые параметры определяют метрическую систему единиц, информационные слои и параметры. После формирования символов всех компонентов автоматически вызывается программа компоновки библиотек pclib, которая создает библиотеку.

В результате анализа программ автоматического формирования библиотек символов и конструктивов [31, 134] получены многоуровневые модели формирования символов и конструктивов компонентов, приведенные на рис. 3.1–3.3. На рис. 3.1 представлена модель непосредственного формирования символов и конструктивов компонентов. При этом используются функциональные модули интерфейса УИ САПР с промышленными САПР. Подобными модулями являются методы поиска и чтения компонентов в таблице соответствия (файл Table), методы поиска путей и имен формализованного задания (файл CodRead). Специализированными являются функции формирования файла макрокоманд, а также методы чтения, добавления и записи таблиц символов и конструктивов. Метод отличается быстродействием, но требует знания структуры двоичных файлов символов (SYM) и конструктивов (PRT). Результаты проверяются в среде САПР PCAD с помощью текстового редактора символов и компонентов PCCOMP и интерактивных графических редакторов схем и конструктивов.

Модель формирования символов и конструктивов компонентов в текстовом формате обмена информацией САПР PCAD – PDIF (PCAD Data Interchange Format) в файлах типа pdf приведена на рис. 3.2. Файлы типа pdf преобразуются во внутренние двоичные форматы с помощью вспомогательной программы pdifin.exe. Программы pdifin должны соответствовать формату базы данных САПР PCAD. Для версии PCAD 4.5 используется формат базы данных 1.04, а для версии PCAD 8.5 – формат базы данных 2.09. Текстовый формат PDIF применяется для преобразования схем и библиотек компонентов как в САПР PCAD, так и в других САПР электронной аппаратуры: ALTIUM PCAD, ORCAD.

Модель непосредственного формирования библиотек символов и конструктивов компонентов приведена на рис. 3.3. Способ непосредственного формирования библиотек является самым быстрым и редко использующим обращение к жесткому диску. В процессе формирования библиотек во внутреннем формате САПР PCAD используются функции нижних уровней табличного интерфейса, описанного в п. 3.2. Объясняется такой подход общностью структур данных схемы (файл типа SCH) и библиотеки символов (файл типа SLB). Схема является множеством компонентов и соединений между ними, а библиотека – множеством компонентов.

Для формирования библиотек компонентов разработан ряд программ. Символьные библиотеки формируются с помощью таблицы uipcad.dbt и программы symbdbt. Таблица uipcad.dbt является информационным обеспечением формирования символов компонентов. Информационное обеспечение формирования символов и конструктивов компонентов находится в таблице uipcad.dbn и отличается от таблицы uipcad.dbt наличием данных о конструктиве, в соответствии с контактами выводов. Таб-

лицу uipcad.dbn использует программа формирования символов компонентов symbdbn. Имя создаваемой библиотеки запрашивается, а тип slb присваивается автоматически. Таблица типа dbn отличается от таблицы типа dbt только последним столбцом. В текст описания столбца введены секции, разделяемые точкой с запятой, и описание вывода дополняется типом эквивалентности и номером контакта.

Рис. 3.1. Многоуровневая модель непосредственного формирования символов и конструктивов компонентов

Рис. 3.2. Многоуровневая модель формирования символов и конструктивов компонентов с использованием символьного файла типа pdf

Запись имеет следующий формат:

<описание>::="<текст>"
<текст>::=<описание группы выводов(ОГВ)>;[<ОГВ>; ...<ОГВ>]
<ОГВ>::=<описание вывода(ОВ)>,[<ОВ>, ... <ОВ>]
<описание вывода>::=<тип вывода> <группа эквивалентности>

<номер контакта> <имя вывода в САПР PCAD>
<имя вывода в САПР COD>.

Рис. 3.3. Многоуровневая модель
непосредственного формирования библиотек символов и конструктивов

Следует обратить внимание на описание компонентов, для которых функции отдельных секций – логических элементов – имеют различные или эквивалентные функции. В этом случае выводы питания компонента описываются не в каждой секции, а в последнем описании группы выводов. Описания группы выводов отдельных секций разделяются точкой с запятой, и их количество не ограничивается. После последней секции точка с запятой может отсутствовать.

В САПР COD на функциональном уровне выводы питания не используются, но они необходимы для перехода к конструкторскому уровню. Для отличий выводов питания приняты правила обозначений: 0V – цифровой общий проводник; 0VA – аналоговый общий проводник; UP, UP1, UP2 – вывод положительного напряжения питания в порядке возрастания; UN, UN1, UN2 – вывод отрицательного напряжения питания в порядке возрастания по абсолютной величине. Обозначение выводов формируется в соответствии с данными табл. 3.9.

Например, описание выводов микросхемы К561ЛА7 будет иметь следующий вид:

"0 1 1 IN1 NIN1, 0 1 2 IN2 NIN2, 1 0 3 Y NOUTP;
0 2 5 IN1 NIN1, 0 2 6 IN2 NIN2, 1 0 4 Y NOUTP;
0 3 8 IN1 NIN1, 0 3 9 IN2 NIN2, 1 0 10 Y NOUTP;
0 4 12 IN1 NIN1, 0 4 13 IN2 NIN2, 1 0 11 Y NOUTP;
0 0 7 0V 0V, 0 0 14 UP1 UP1"

Таблица 3.9

Таблица имен выводов компонентов

Назначение символа	Значение символа	Семантика значения
Номер вывода компонента	N	Первый символ в имени
Количество выводов одного типа	K	Первый символ в имени
Массив выводов одного типа	M	Первый символ в имени
Отношение направления передачи данных	IN	Вход (input)
	OUT	Выход (output)
	IO	Двунаправленный вывод (input, output)
	ADR	Вход адреса
	0A	Уровень нуля аналогового
	GND	Общий вывод сигналов
	UP	Напряжение питания положительное
	UN	Напряжение питания отрицательное
Отношение разрядов числа, строки	MIN	Минимальный номер входа и цепи соответствует старшему разряду
	MAX	Максимальный номер входа и цепи соответствует младшему разряду
	CP	Старший разряд
	MP	Младший разряд
Символ функции сигнала	R	Начальная установка (Reset)
	S	Установка состояния (Set)
	V	Разрешение работы
	C	Синхронизация
Вид вывода	P	Прямой (Positiv)
	N	Инверсный (Negativ)
Тип сигнала	D	Цифровой (Digital)
	A	Аналоговый (Analog)
Тип канала (цепи) и сигнала	E	Электрический
	O	Оптический
	P	Пневматический
	R	Радиоканал
	M	Механический

Для автоматизации создания библиотек конструктивов компонентов разработана программа PRTBAT. Информационное обеспечение формирования конструктивов находится в таблицах uipcad.dbn и pac.dbt. Таблица uipcad.dbn рассмотрена выше, структура таблицы pac.dbt приведена ниже. Таблица содержит семь столбцов. Первый столбец содержит обозначение корпуса по стандарту, и его содержание совпадает с соответствующим столбцом таблиц uipcad.dbt и uipcad.dbn. Второй столбец определяет количество контактов, третий и четвертый – габарит по длинной (GABX) и короткой (GABY) сторонам. Пятый и шестой столбцы определяют шаг

по длинной (STEPX) и короткой (STEPY) сторонам в миллиметрах. Седьмой столбец содержит имя образа (следа – FP) конструктива на печатной плате. Параметром программы является имя создаваемой библиотеки. Сообщения об обнаруженных ошибках записываются в файл PRTBAT.ERR. Программа работает аналогично предыдущей. С помощью информации из таблиц создается файл типа pdf, который преобразуется утилитой PCAD PDIFIN в файл конструктива типа prt. По завершении создания множества prt-файлов создается командный файл для утилиты PCLIB, а также библиотека, и стираются файлы типа prt. Формирование символов и конструктивов компонентов с использованием текстового описания и последующим преобразованием в формат базы данных САПР PCAD утилитой PDIFIN характеризуется наглядностью, но непосредственное формирование символов и конструктивов в формате базы данных отличается быстродействием.

Алгоритм создания символов и конструктивов компонентов в формате конкретной САПР с последующим объединением в библиотечную структуру приведен на рис. 3.4, алгоритм автоматического создания библиотек символов и конструктивов компонентов в формате конкретной САПР представлен на рис. 3.5.

Для формирования символов и конструктивов компонентов нужно найти пути, открыть файл таблицы соответствия uipcad.dbn и загрузить данные. После загрузки данных производятся поиск и чтение строки таблицы, формируется имя символа или конструктива и заголовок соответствующего файла. После формирования заголовка заполняются таблицы в формате конкретной САПР, открывается выходной файл, производятся запись сформированных таблиц и закрытие файла. После закрытия файла можно перейти на новую строку таблицы данных, и процесс повторяется с поиска и чтения строки таблицы данных. После исчерпания таблицы данных компонентов файлы символов и конструктивов компонентов объединяются в библиотеки.

Программа непосредственного формирования символов и конструктивов в формате базы данных 2.09 САПР PCAD 8.5 состоит из модулей считывания информации о компоненте из таблицы UIPCAD.DBN, модуля формирования структур данных в формате базы данных PCAD 8.5 и модуля записи в файл символов и конструктивов.

Для каждого компонента из таблицы UIPCAD.DBN формируются файлы символов типа SYM и конструктивов типа PRT, которые в последующем объединяются в библиотеки символов типа SLB и конструктивов типа PLB. В процессе работы программы созданный символ отображается на экране для контроля процесса формирования символов и конструктивов.

Рис. 3.4. Алгоритм создания символов и конструктивов компонентов в формате конкретной САПР с последующим объединением в библиотечную структуру

Рис. 3.5. Алгоритм автоматического создания библиотек символов и конструктивов компонентов в формате конкретной САПР

3.6. Автоматизация формирования текстовых описаний информационной поддержки жизненного цикла вычислительных систем

Проектирование начинается с формирования технического задания, в котором должны быть отражены цель создания системы или устройства и множество ограничений. Синтез устройства (системы) в виде формализованного задания является трудноформализуемым творческим этапом, выполняемым инженером. Вид формализованного задания зависит от используемой САПР, хотя информация инвариантна относительно конкретной САПР. В формализованном задании содержится информация о типах и именах компонентов, соединениях между компонентами и формах взаимодействия компонентов между собой и с внешней средой.

Развитие средств обработки, передачи информации и автоматизированного проектирования позволяет создать информационные модели объектов для всего жизненного цикла изделия. Результатом развития информационных, материальных и энергетических технологий стали *виртуальные предприятия* – распределенный комплекс организаций, проектирующих, производящих и обслуживающих промышленные изделия. Эффективность виртуальных предприятий обусловлена пространственным распределением природных, энергетических и интеллектуальных ресурсов и промышленной зоны. Весь жизненный цикл изделия, начиная с проектирования и производства и кончая обслуживанием и утилизацией, должен сопровождаться информационным обеспечением, соответствующим системам автоматизации производства и их интеграции по международному стандарту STEP (Standard for Exchange of Product data) ISO 10303 и ГОСТ Р ИСО 10303-1–99 [104]. Стандарт ISO 10303 дополняют стандарты ISO 15531 MANDATE (Industrial manufacturing management data) представления данных для виртуальных предприятий, ISO 13584 P-LIB (Parts Library) представления данных о стандартных компонентах промышленных изделий, ISO 14959 Parametrics представления данных о параметрах изделий, ISO 15926 (Industrial automation systems and integration) промышленной автоматизации.

Информация об изделии создается в процессе его проектирования при выполнении процедуры синтеза, а используется и дополняется при производстве, эксплуатации и утилизации на протяжении всего жизненного цикла в соответствии с ГОСТ Р ИСО 10303–99 (ISO 10303) [104]. Информация об изделии представляется на языке описания в соответствии с правилами грамматики [146]. Описания изделий должны восприниматься как пользователем, так и формальной системой и называются формализованными заданиями. Формализованное задание (FZ) состоит из множества разделов, синтаксически и семантически однородных [29, 33, 103]. Поэтому

синтез и анализ формализованных заданий выполняют по разделам. Разделы формализованного задания (PFZ) могут описывать начальные значения, внешние воздействия, компоненты и связи между ними, управление процессом анализа. Описания изделия могут соответствовать различным уровням абстракции и этапам проектирования. Формализованные задания САПР PCAD в виде текстового файла типа alt и САПР ПРАМ 5.3 типа prm соответствуют конструкторскому уровню проектирования, а формализованные задания УИ САПР соответствуют функциональному уровню.

Для хранения и передачи описаний лучшей является текстовая форма, а для быстрого восприятия информации пользователем удобнее графическая форма. Поэтому возникает задача автоматического формирования текстовых описаний и преобразования их в графическую форму для быстрого восприятия человеком. Например, в стандарте ИСО 10303–99 предусмотрен язык EXPRESS и соответствующая графическая форма EXPRESS-G, формируемая автоматически [103].

Для комплексной САПР (КСАПР) ПРАМ 5.3 [32, 103] формализованное задание представляется в виде множества разделов, которые хранятся в одном файле. Основными разделами формализованного задания являются: «Перечень элементов», «Соединения», «Паспорт». Синтаксис и семантика разделов формализованного задания описаны в [29, 32, 87, 101] и в отраслевом стандарте ОСТ 4Г 0.091.397–85. Маршрут проектирования подробно описан в работе [30]. Результат проектирования, который в ПРАМе называется архивом, содержит наряду с выходными данными разделы исходного формализованного задания. Полный архив для модуля многоканального аналого-цифрового преобразователя находится в файле R22 подкаталога PRAM. В этом же подкаталоге в файле PRAM53.HLP расположен текст отраслевого стандарта. Описание R22 соответствует реальному модулю, изготовленному малой серией в заводских условиях.

Формализованное задание синтезируется инженером или формируется автоматически в САПР COD при выборе в меню локального или удаленного выполнения (Execute, Remote Execute) подменю текстового интерфейса – PRAM. Раздел формализованного задания описания соединений формируется по компонентам или по цепям. Файл типа prm содержит три раздела: «Паспорт», «Перечень элементов» и «Соединения», причем первый копируется из файла X:\fa\pram\pasport.txt.

Формализованные задания САПР ПРАМ 5.3 и файл типа alt САПР PCAD соответствуют конструкторскому уровню, поэтому необходимо преобразование компонентов произвольной разрядности в фактическую и упаковка по корпусам.

Файл типа alt воспринимается всеми версиями САПР PCAD (4.5, 8.5–8.7, 2000–2006), но в последней не следует указывать путь к библиотеке компонентов, которая выбирается в меню PCAD2000. Файл типа alt содержит

разделы (секции): PATH, BOARD, SHEET. Раздел PATH содержит множество путей поиска файлов конструктива модуля ЭВМ и библиотеки конструктивов компонентов (PATH=<путь 1>; <путь n>;). Раздел BOARD содержит имя файла конструктива модуля ЭВМ (BOARD = <полное имя файла>;). Раздел SHEET содержит описание перечня компонентов и соединений в подразделах PARTS и NETS соответственно (SHEET = <имя листа> PARTS <перечень компонентов> NETS <список соединений> ENDSHEET). Если не указывать имя листа в разделе SHEET, то листы будут именоваться 01, 02, т. е. в порядке возрастания. Подраздел PARTS представляет собой перечисление компонентов схемы, причем одинаковые компоненты группируются и записываются в одной строке через запятую (<тип компонента> = <имя конструктива 1>, < имя конструктива n>;). Подраздел NETS содержит описание соединений в формате <имя цепи> = <имя конструктива 1>/<номер контакта>, <имя конструктива n>/<номер контакта>). В соответствии с маршрутом проектирования в САПР PCAD [31] файл типа alt преобразуется программой PCNLT в двоичный файл типа pkg для конструкторского проектирования, а в САПР PCAD2000 файлы конструктива и описания загружаются пунктом меню File/Open редактора печатных плат [120, 128].

Формат EDIF (Electronic Data Interchange Format) является международным стандартом и используется для обмена описаниями электронной аппаратуры между различными САПР [104]. Формат EDIF поддерживается основными САПР электронной аппаратуры, файл обмена типа EDF создается и воспринимается САПР PCAD и ORCAD. В САПР PCAD включена утилита преобразования файлов типа edf в файлы типа alt. В отличие от описания конструкторского уровня, в формате EDIF возможно описание внешних воздействий. Развитием стандартов EDIF и STEP в области электроники занимается рабочая группа ISO JWG9 [128].

Язык EXPRESS предназначен для представления данных об изделии и обмена этими данными между вычислительными системами. Структура обмена в соответствии со стандартом ИСО 10303-21–99 должна быть последовательным файлом, содержащим две секции, одна из которых является заголовком, а во второй размещены данные. Секция заголовка содержит информацию о всей структуре обмена и начинается с лексемы HEADER и заканчивается лексемой ENDSEC. Секция заголовка должна содержать экземпляры объектов: file_description, file_name, file_schema.

Пример секции заголовка:

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('FILE CONTAINS A  UNIT MODEL'),'2:1');
```

FILE_NAME('COMPUTER UNIT MODEL FP01 #1');
FILE_SCHEMA(('EXAMPLE OF SCHEMA FP01'));
ENDSEC;

Структура секции данных приведена ниже:

DATA;
<Содержание секции данных>
ENDSEC;
END-ISO-10303-21;

Принятие международного стандарта привело к появлению программных продуктов, поддерживающих информационное обеспечение жизненного цикла изделия, в соответствии с CALS-технологией [104, 133]. Программные продукты поддержки жизненного цикла изделия сокращенно называются PLM (Product Lifecycle Management).

Программа ST-DEVELOPER является средой разработки для данных на языке EXPRESS с использованием универсальных языков программирования, ST-WEB PUBLISHER является инструментом представления данных в сети, ST-EXPRESS преобразует текстовое описание в графическое. Комплексы программ ENOVIA фирмы IBM, IMAN фирмы UNIGRAPHICS могут служить примерами PLM-продуктов.

Таблица 3.10

Языки описания, использующие XML

Имя	Полное наименование	Назначение
CKML	Conceptual Knowledge Markup Language	Описание методов представления знаний и результат анализа данных
UDDI	Universal Description Discovery Integration	Инструментальная поддержка универсального каталога
MML	Mathematical Markup Language	Описание математических формул и данных
PGML	Precision Graphics Markup Language	Описание двумерной масштабируемой векторной графики
VML	Vector Markup Language	Описание двумерной векторной графики
SML	Shema Markup Language	Описание электронных схем (СФУ)
TML	Tutorial Markup Language	Описание обучающей информации для ее онлайнового представления и оценки знаний студентов
WML	Wireless Markup Language	Описание общего содержания, предназначенного для передачи по беспроводной сети
RDF	Resource Description Framework	Язык описания ресурсов
OWL	Web Ontology Language	Язык описания онтологий
WSDL	Web Services Description Language	Язык описания сетевых сервисов

Работу с текстовыми описаниями облегчают языки разметки: HTML, SGML и XML [29, 33, 153], причем SGML и XML позволяют описывать структуры документов и являются основой специализированных языков, приведенных в табл. 3.10. Отсутствие языка описания электронных схем на базе XML привело к созданию как языка SML, так и автоматического формирования описания схемы. Создаются два файла с именем формализованного задания: информационный типа xml и стартовый типа htm для загрузки схемы в программу просмотра, поддерживающую язык XML, например Internet Explorer для Windows или многоплатформенная Mozilla. Отображение в программе просмотра обеспечивается библиотекой функций в файле Parser. Стандарт XML и его поддержка упростили отображение схем и временных диаграмм.

		1 Формализованное задание (ФЗ)	
2	Модели поиска путей и имен ФЗ (CodRead)	Модели компонентов (Comp)	Модели управляющих процедур и функций генерации сигналов (Control)
3	Структура данных варианта схемы (TSCH)		
4	Методы чтения, добавления и записи структуры данных файла описания	Методы чтения, записи, поиска в таблице данных варианта схемы (TSCH)	Методы чтения, записи и поиска компонента в таблице соответствия (Table)
5	Методы чтения, добавления и записи структуры данных компонента, его выводов и цепей	Методы упаковки компонентов в корпус в таблице (TSCH)	Методы поиска, чтения и записи информации о контактах компонента и подключенных цепях
6	Файл проекта конкретной САПР или стандартная структура обмена данными		

Рис. 3.6. Многоуровневая структура текстового интерфейса

Рассмотрим структуру и основные функции для формирования всего множества представления текстовых файлов описания функционального уровня вычислительных устройств и систем. Многоуровневая структура текстового интерфейса приведена на рис. 3.6.

3.7. Модели компонентов обеспечения интерфейса САПР COD с САПР CATIA

САПР CATIA является комплексной системой автоматизированного проектирования (КСАПР), охватывающей множество предметных областей. Эффективность КСАПР CATIA объясняется наличием общих проектных процедур и операций и специализированных модулей. Функциональные подсистемы соответствуют основным предметным областям и традиционным направлениям подготовки специалистов: машиностроение, проектирование промышленных установок, электроника, электротехника, дизайн и размещение оборудования в промышленных объектах, оборудование зданий. Работа проводилась с версией 5 САПР CATIA для персональных ЭВМ R16 IBM Order No: 8232SP, в которой автор указан пользователем.

Проектирование начинается с синтеза возможных вариантов схем и их описания в виде формализованных заданий учебно-исследовательской САПР COD. После анализа формализованного задания с описанием нескольких вариантов схем оцениваются производительность и критерии эффективности и производится выбор оптимального варианта. Принципиальная схема может отображаться в стандартных программах просмотра Интернета (Microsoft Internet Explorer (IE), Mozilla). Предварительный образ конструкции модуля в САПР COD отображается в среде виртуальной реальности (файл типа wrl) или X3D (файл типа X3D).

Конструкция модуля проектируется в приведенных САПР электронной аппаратуры (CAD EDA). В CAD EDA производится размещение компонентов и трассировка соединений, и результат проектирования модуля передается в САПР CATIA. Например, при проектировании конструкции модуля в САПР PCAD производится вывод описания в форматах PDIF (файл типа pdf) с преобразованием в формат IDF с помощью вспомогательной программы IDF2PCAD.exe. Файл типа idf воспринимается подсистемой САПР CATIA Circuit Board Design. По завершении проектирования модуля решаются вопросы размещения и сопряжения с другими компонентами комплексного изделия.

Преобразование формализованного задания описания устройства в документ САПР CATIA выполняется автоматическим формированием файла макрокоманд на языке VBSCRIPT с расширением catvbs. Используемое при преобразовании подмножество команд приведено в табл. 3.11. Основные модули интерфейса САПР CATIA представлены в табл. 3.12.

Рекомендуется создать библиотеку компонентов в каталоге объектов САПР CATIA – ObjCat. Компоненты должны соответствовать таблицам uipcad.dbm или uipcad.dbn и таблице геометрии компонентов pac.dbt в ка-

талоге CODOS\BIN. Составные компоненты могут состоять из множества простых компонентов.

Таблица 3.11

Подмножество команд САПР CATIA

Назначение команды	Имя команды	Имя подкоманды	Макрокоманда
Создание нового файла	File	New	Language="VBSCRIPT" Sub CATMain() Set documents1 = CATIA.Documents Set productDocument1 = documents1.Add("Product") End Sub
Открыть файл		Open	Language="VBSCRIPT" Sub CATMain() Set documents1 = CATIA.Documents Set productDocument1 = documents1.Open(FileName) End Sub
Сохранить файл		Save	Language="VBSCRIPT" Sub CATMain() Set productDocument1 = CATIA.ActiveDocument productDocument1.SaveAs FileName End Sub
Копировать в буфер	Edit	Copy	Language="VBSCRIPT" Sub CATMain() Set productDocument1 = CATIA.ActiveDocument Set selection1 = productDocument1.Selection selection1.Clear Set product1 = productDocument1.Product Set products1 = product1.Products Set product2 = products1.Item(Element) selection1.Add product2 selection1.Copy End Sub
Вставка из буфера		Paste	Language="VBSCRIPT" Sub CATMain() Set productDocument1 = CATIA.ActiveDocument Set selection1 = productDocument1.Selection selection1.Paste End Sub

Исходный код общего модуля содержится в файле mci.cpp, исходный код модуля формирования макросов PCB – в файле mcipcb.cpp, модуля формирования макросов NET – в файле mcinet.cpp, прототипы функций – в файле mci.hpp. Принцип работы всех трех модулей – поиск компонентов в таблице вариантов TSCH и запись команд в выходной файл макрокоманд в соответствии с найденными компонентами.

В модуле PCB для поиска микросхем используем функцию FindChip(), в которой формируется список компонентов, определяются их параметры, и эти параметры заносятся в структуру Chip. Запись команд в файл макроса выполняет функция draw_Board().

Таблица 3.12

Основные модули интерфейса

Имя модуля	Назначение модуля
Comp	Исходный текст моделей компонентов верхнего уровня
mci, mpi, mji, mcinet, mpinet, mjinet, mcipcb, mpipcb, mjipcb	Исходный текст модулей на языках программирования CPP, PL1 и JAVA
Control	Управляющий модуль (Iprint, Aprint, Nprint)
Codread	Функции чтения путей и имен файлов УИ САПР COD
Table	Функции для работы с таблицами компонентов uipcad.dbm, uipcad.dbn
Tsch	Функции для работы с таблицей варианта

В модуле NET существуют функции FindSNet() и FindMNet(). Назначение – поиск в таблице вариантов стационарных и мобильных сетевых компонентов. Если сетевые компоненты в формализованном задании найдены, то запускается функция draw_room(), которая записывает в файл макроса команды создания комнаты, компьютеров и точек доступа.

В общем модуле присутствуют все вышеперечисленные функции. Переменная Kchip содержит количество найденных микросхем, переменная Knet – количество сетевых компонентов, равное сумме стационарных (KSnet) и мобильных (KMnet). Сетевым компонентом могут быть стационарные или мобильные серверы или персональные компьютеры, телекоммуникационные компоненты (коммутаторы, маршрутизаторы, точки доступа). Если количество сетевых компонентов не равно нулю, то будут создаваться помещения для сетевых объектов. Если имеются сетевые компоненты и микросхемы, то отображаются сеть и модуль. Может быть задана локальная беспроводная сеть вне здания.

Развитие вычислительных технологий привело к появлению программируемых сетевых вычислительных устройств. Словарь основных переменных и функций приведен в табл. 3.13.

В результате выполнения многовариантного формализованного задания создается множество файлов макросов, количество которых соответствует числу вариантов. Полный путь к библиотеке компонентов ObjCat, приведенной в табл. 3.14, определяется значением переменной окружения CODLOC.

Таблица 3.13

Словарь основных переменных и функций

Имя функции или переменной	Назначение функции или переменной
int mci()	Точка входа, главная функция
void FindSNet()	Функция поиска стационарных сетевых компонентов
void FindMNet()	Функция поиска мобильных сетевых компонентов
void FindChip()	Функция поиска микросхем
void draw_Board()	Функция генерации и записи макроса создания печатной платы
void draw_Room()	Функция генерации и записи макроса создания компьютерного класса
int KSnet	Количество найденных стационарных сетевых компонентов
int KMnet	Количество найденных мобильных сетевых компонентов
int Kchip	Количество найденных схемных компонентов
int KUnet	Количество найденных сетевых устройств
int KURnet	Количество найденных реальных сетевых устройств
int KUVnet	Количество найденных виртуальных сетевых устройств

Таблица 3.14

Таблица компонентов в каталоге ObjCat

Имя файла	Назначение
A1-A4.CATDrawing	Формат чертежа A1, A2, A3, A4
Dip14-Dip48.CATPart	Корпуса микросхем с числом выводов 14, 16, 20, 24, 28, 32, 42, 48
R.CATPart	Резистор
C.CATPart	Конденсатор
IBM.CATPart	Модуль IBM PC
hs_m.CatProduct	Образ мужчины
hs_w.CatProduct	Образ женщины
PC.CATPart	ЭВМ IBM PC
MPC.CATPart	Мобильный IBM PC
TPC.CATPart	Мобильный планшетный PC
PPC.CATPart	Карманный компьютер
ap.CATPart	Беспроводная точка доступа
ap_sphere.CATPart	Активная точка доступа
ap_konus.CATPart	Активная направленная точка доступа

Разработанные модули включаются в библиотеку обобщенного интерфейса посредством перекомпиляции статической и динамической библиотек. Для компиляции статической библиотеки используем командный файл lcwgi.bat для Windows и lcogi.cmd для OS/2. Для компиляции динамиче-

ской библиотеки вначале нужно получить файл определения типа DEF. Файлы определений типа DEF создаются для OS/2 командой gendefo.cmd и для Windows командой gendefw.bat. После формирования файлов определений можно перейти к созданию динамических библиотек типа DLL командными файлами icogi.cmd и icwgi.bat. В результате при выборе пункта меню Execute->Interface CATIA->Command NET and PCB выполнится команда редактора LPEX fti.lx и будет создан файл микрокоманд.

Файл макрокоманд для САПР CATIA выполняется выбором пункта меню Tools и подменю Macro->Macros. В окне Macros выбирают файл и выполняют команду Run. Результат проверяется, и корректируется формализованное задание, компоненты библиотеки или модуль преобразования. Результат можно сохранить в форматах CATIA (CATProduct) или VRML (wrl), STEP (stp).

Образ изделия на начальном этапе проектирования модуля ЭВМ можно получить, используя интерфейс со средой виртуальной реальности. Можно отобразить изменения цифровых сигналов на выводах микросхем и температуру микросхемы. Однако трехмерное представление в САПР CATIA отличается увеличенным объемом, качеством и отображением деталей.

3.8. Модели компонентов обеспечения интерфейса САПР COD со средой виртуальной реальности

В среде виртуальной реальности [3, 33, 136, 152–154] могут быть представлены модули, блоки, ЭВМ и вычислительные системы со стационарными и мобильными объектами, а также технологические процессы их изготовления. Основой представления виртуальной реальности являются описание множества объектов на языке VRML и программы преобразования описаний в изображение, называемые программами просмотра VRML-описаний [29, 33, 55, 56].

Описание множества объектов (представляющих внеший вид модулей, блоков, стоек ЭВМ и вычислительных машин) на языке VRML можно получить путем автоматического преобразования формализованного задания на моделирование. Формализованное задание может быть описано на любом из универсальных языков программирования PL/1, C++, JAVA. Для преобразования формализованного задания в среде OS/2 или Windows в меню нужно выбрать язык описания, ввести имя формализованного задания и в меню выполнения (execute) выбрать пункт VRML. Полученный файл типа wrl следует отобразить с помощью программы просмотра. Тип файла wrl является сокращением от world (мир).

Основой автоматического преобразования формализованного задания в описании на языке VRML являются многофункциональные модели компонентов с общим интерфейсом [6]. Наличие моделей компонентов с общим интерфейсом позволяет преобразовать формализованные задания в результаты для различных приложений. В зависимости от приложения синтезируются модели компонентов и управляющих модулей, которые объединяются в статические и динамические библиотеки и выбираются в зависимости от вида приложения. С целью снижения трудоемкости создания моделей компонентов и управляющих модулей используются модели различных уровней. Модели верхнего уровня передают параметры моделям среднего уровня, которые в свою очередь формируют разделы выходного файла с использованием модели нижнего уровня. Модели верхнего уровня не зависят от вида приложения. Вид приложения определяется моделями среднего и нижнего уровней.

Заголовок файла типа wrl на языке VRML формируется управляющим модулем и записывается в выходной файл. После вызова всеми компонентами процедур верхнего уровня завершается заполнение таблицы варианта схемы и формируется выходной файл в формате VRML. Геометрические модели компонентов выводятся в порядке их описания в формализованном задании. Регулярные структуры вычислительных систем могут быть описаны в цикле, а нерегулярные структуры могут быть представлены в описании конкретных компонентов. Управляющий модуль после завершения формирования выходного файла закрывает все открытые файлы. Образы компонентов находятся в подкаталоге X:\FA\VRML\ObjVRML.

Формализованные задания, подлежащие преобразованию в VRML-формат, не должны содержать функциональных моделей компонентов, так как модели должны формировать VRML-файл. Поэтому функциональные модели нужно заключить в комментарии. Модели компонентов содержат формализованные задания, начинающиеся с сочетания FP03 и модели вычислительных систем (FP08–FP12). В описаниях вычислительных систем содержатся модели оперативной обработки (МКМ) и завершающей обработки данных (MPRINT).

Для создания и отладки новых моделей компонентов преобразования формализованного задания в VRML-файл нужно включить в ФЗ интерфейсную модель компонента верхнего уровня.

3.9. Синтез модели и алгоритмов работы аналого-цифровой подсистемы с USB-интерфейсом

Аналого-цифровые подсистемы с USB-интерфейсом на БИС позволяют вводить аналоговые и цифровые сигналы, производить их обработку и вывод аналоговых и цифровых сигналов. С повышением степени инте-

грации аналого-цифровые устройства приближали к источникам сигналов. Таким образом уменьшали помехи при передаче аналоговых сигналов по линиям связи, длину аналоговых линий и увеличивали длину линий для передачи цифровых сообщений. При этом количество аналоговых сигналов на одно устройство уменьшалось, а количество устройств увеличивалось.

Перспективными являются аналого-цифровые устройства с USB-интерфейсом (ADCUSB) и с радиоканалом (ADCRCA). Устройства ADCUSB отличаются малым удалением от системного блока, но при этом нет необходимости в дополнительном источнике питания. Устройства с радиоканалом допускают значительное удаление от точки доступа (AP), но при этом требуются дополнительный источник питания и сравнительно большие затраты энергии при передаче данных.

Аналого-цифровые устройства с USB-интерфейсом выпускаются различными зарубежными предприятиями. Основные характеристики таких устройств приведены в табл. 3.15.

Таблица 3.15

Основные характеристики аналого-цифровых устройств с USB-интерфейсом

Название	Цена, у.е.	Кол-во входов АЦП	Разряд ряд-ность АЦП	Час-тота АЦП кГц	Кол-во выхо-дов ЦАП	Раз-ряд-ность ЦАП	Час-тота, Гц	Кол-во цифровых входов/ выходов	Стоимость единицы произво-дительно-сти, у.е.
NI USB 6009	285	8/4	13/14	48	2	12	150	8 + 4	0,00044
NI DAQ Pad 6016	1345	16/8	16	200	2	16	300	32	0,00042
LabJack U12	119	8/4	12	8	2	10	50	20	0,001
Eagle Daq	700	16/8	14	250	4	14		8 + 8 +8	0,00022
DAT dt9836	2400	12/6	16	225	2	16	500	16 + 16	0,00066
MCC USB 3110	399	–	–	–	4	16	100	8	–
SuperLogics 9801	575	16/8	12	100	–	–	–	16	0,0047
Phidgets	–	8/4	–	–	–	–	–	8 + 8	–

В качестве примера на рис. 3.7 приведена функциональная схема устройства ADCUSB типа NIDAQ 6009 фирмы Natnional Instruments. Устройство содержит источник питания для дополняющих компонентов, собственно интерфейс USB, микроконтроллер шины USB, АЦП на 8 каналов, 2 ЦАП и 12 цифровых линий ввода-вывода в двух группах (8 и 4).

Модель аналого-цифровой системы содержит цифровую вычислительную подсистему или машину, аналого-цифровые подсистемы, подключенные к цифровой части с помощью стандартных каналов ввода-вывода. Аналого-цифровая подсистема ADCUSB подключается с помощью канала USB, по которому передается управляющая информация и выводимые

данные и принимаются результаты преобразования и цифровые сигналы. Управляющая информация задает конфигурацию подсистемы. Структуру данных можно передавать непосредственно или передавать номера цепей, которым соответствует структура данных. Передача номеров цепей облегчает визуализацию результатов и соответствует принципам высокоуровневой УИ САПР COD. Синтаксис и семантика параметров функции ADCUSB приведены в табл. 3.16. Идентификатор MODE соответствует одиночным или дифференциальным аналоговым входам.

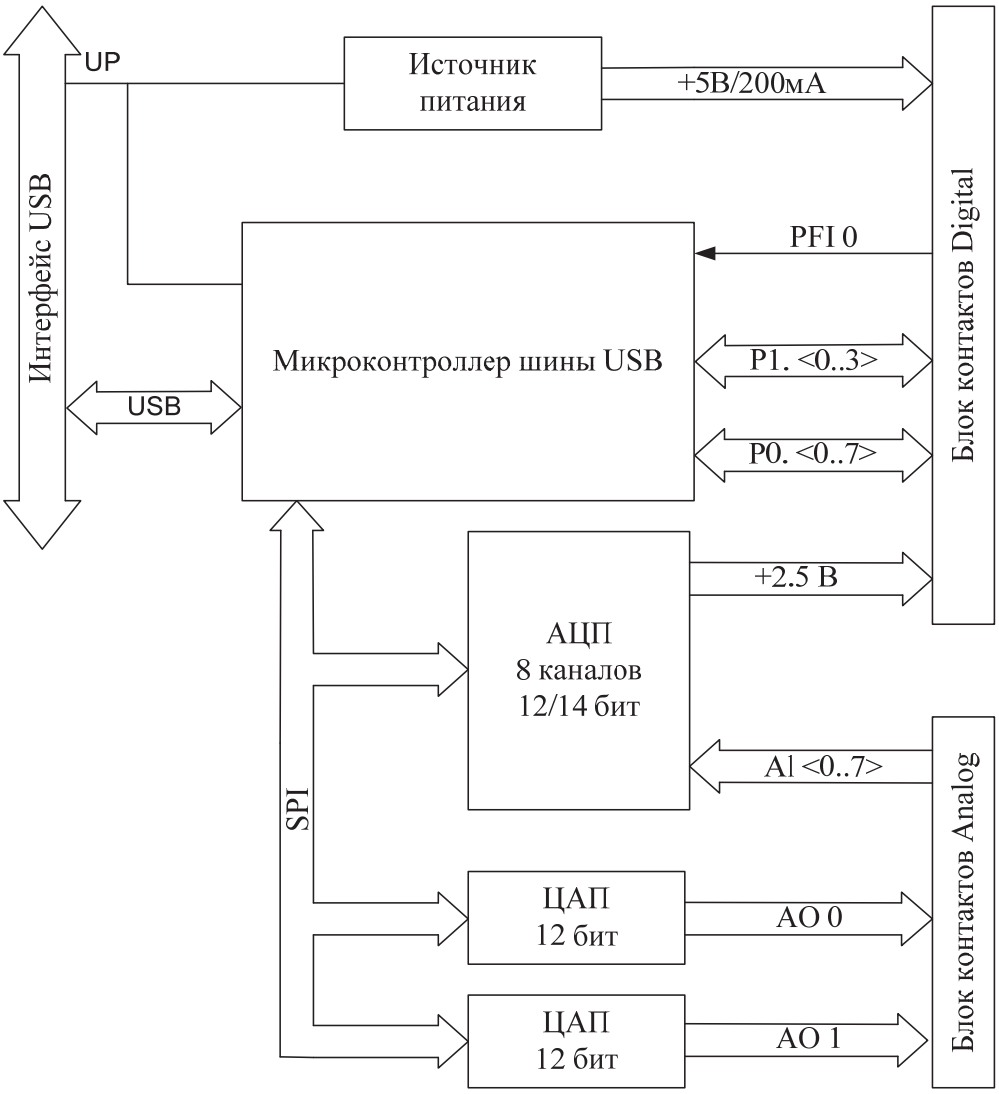


Рис. 3.7. Функциональная схема ADCUSB типа NIDAQ 6009

Формализованное задание fp03nin служит для отладки модели функции ADCUSB, задание fp03ninm служит для отладки модели функций ADCUSB при работе совместно с внешним мультиплексором, расширяющим возможности устройства. Схема считается работоспособной, если это сопротивление находится в заданных пределах. Такой подход проверен в проектах

для министерства геологии и эффективен при дистанционном использовании вычислительных устройств и систем. Формализованное задание fp04nin соответствует fp03nin, но отличается отсутствием исходного текста функции ADCUSB и использованием функциональной модели из библиотек компонентов. Задание fp04ninm соответствует fp03ninm, содержит внешний мультиплексор и благодаря отсутствию исходного текста функции ADCUSB удобнее для использования. Возможно использование нескольких аналого-цифровых подсистем ADCUSB с внешними коммутаторами и другими компонентами. Текст формализованного задания fp04ninm приведен после структурной схемы в разделе 5.

Таблица 3.16

Таблица синтаксиса и семантики модели устройства USB NI6009

№ п/п	Идентификатор	Тип данных PL/I	Тип данных C,C++,JAVA	Тип данных ADA	Семантика	Примечания
1	TYP	CHAR(*)	char(C++) string(JAVA)	Character	Тип элемента	NI6009
2	NEL		Int	Integer	Номер элемента	
3	NINDAC1	DEC FIXED(3)	intVector(C++) int[] (JAVA)		Номер кода DAC1	
4	NINDAC2				Номер входа DAC2	
5	MNOUTADC				Вых. ADC. Массив номеров	
6	MNDIO1				Номер цепи цифр. вх-вых. MDIO1	8 разрядов
7	MNDIO2				Номер цепи цифр. вх.-вых. MDIO2	4 разряда
8	MODE				Режим работы АЦП	10083-один. 10106-дифф.

Модель подсистемы ADCUSB состоит из нескольких разделов [29]. Раздел FTYPE производит проверку допустимости типов компонентов, которые передаются идентификатором TYP и номером компонента NEL. Номер компонента необходим для отличия однотипных компонентов. Функции раздела FS – инициализация переменных и восстановление состояний. Вычисление значений выходов выполняется в разделе FOUT для каждого компонента подсистемы: цифроаналоговых преобразователей (DAC1, DAC2), аналого-цифрового преобразователя ADC и цифрового ввода-вывода DIO. Для каждого компонента выполняется определенная последовательность действий (шагов): чтение состояния компонента, запись конфигурации и состояния, выполнение действия в соответствии с таблицей допустимых команд (табл. 3.17).

Таблица 3.17

Таблица допустимых команд

Назначение команды	Имя команды	Используемые параметры	Возвращаемые значения
Создание задачи	DAQmxBaseCreateTask	**TaskName const char[]** – имя задачи	**taskHandle TaskHandle*** – указатель на созданную задачу
Удаление задачи	DAQmxBaseClearTask	**TaskHandle TaskHandle** – имя задачи	**Status int32** – возвращает код ошибки
Создание канала аналогового ввода (описание)	DAQmxBaseCreateAIVoltageChan	**TaskHandle TaskHandle –** имя задачи **physicalChannel** const char [ ] – номер канала входа, например «Dev1/ai0» **terminalConfig** int32 – конфигурация канала ввода **minVal** float64 – минимальное значение (В) **maxVal** float64 – максимальное значение (В) **units** int32 – размерность, оставлять в DAQmx_Val_Volts (В)	**Status int32** – возвращает код ошибки
Аналоговый вывод	DAQmxBaseWriteAnalogF64	**TaskHandle TaskHandle –** имя задачи **numSampsPerChan** int32 – количество выводимых отсчетов **autoStart** bool32 – всегда FALSE **timeout** float64 – время ожидания **dataLayout** bool32 – указание метода группировки значений **writeArray** float64[ ] – массив 64-битных значений для вывода	**sampsPerChanWritten** int32 * – количество отсчетов, записанных в буфер **Status int32** – возвращает код ошибки

Окончание табл. 3.17

Назначение команды	Имя команды	Используемые параметры	Возвращаемые значения
Чтение восьмиразрядных цифровых отсчетов	DAQmxBaseReadDigitalU8	**TaskHandle TaskHandle –** имя задачи **numSampsPerChan –** int32 – количество отсчетов **timeout** float64 – Время ожидания **fillMode** bool32– указание метода группировки значений **arraySizeInSamps** uInt32 – размер массива, хрянящий считанные значения	**sampsPerChanRead** int32 * – количество отсчетов, записанных в буфер **Status int32** – возвращает код ошибки
Вывод восьмиразрядных цифровых отсчетов	DAQmxBaseWriteDigitalU8	**askHandle TaskHandle –** имя задачи **numSampsPerChan** int32 – количество выводимых отсчетов **autoStart** bool32 – всегда FALSE **timeout** float64 – Время ожидания **dataLayout bool32** – указание метода группировки значений **writeArray** uInt8 [ ] – массив выводимых значений	**sampsPerChanWritten** int32 * – количество отсчетов, записанных в буфер **Status int32** – возвращает код ошибки

Полученные результаты проверяются и, в случае достоверности, записываются в таблицу цифровых или аналоговых сигналов по заданному номеру цепи (сигнала).

Пример модели ADCUSB с параметрами в виде номеров цепей на языке C++ NIDAQ 6009 приведен ниже.

```
//
// model of NI USB 6009 device
//
#include <NIDAQmx.h>
#include <vextcpp.h>
#define DAQmxErrChk(functionCall) if( DAQmxFailed(error=(functionCall)) ) goto Error;

void ADCUSB(char* TYP, int NEL,int nindac1,int nindac2,int mnoutadc[8],
int mdio1[8], int mndio2[4], int mode) {

int error=0;
TaskHandle taskHandle=0,taskHandle2=0,taskHandle4=0;
char errBuff[2048]={'\0'};
int32 read;
float64 datad[1000];
uInt8 mdio3[4]={out[mndio2[0]].New,out[mndio2[1]].New,0,out[mndio2[3]].New};
float64 data[2] = {as[nindac1].ANew,as[nindac2].ANew};
// преобразование данных с цифровых цепей
      for (int i=0; i<4; i++) {
            if(mdio3[i]==3) mdio3[i]=1;  }
// Конфигурация и запуск задачи ЦАП1 и ЦАП2
DAQmxErrChk (DAQmxCreateTask("",&taskHandle));
DAQmxErrChk    (DAQmxCreateAOVoltageChan(taskHandle,"Dev1/ao0","",0.0,5.0,
DAQmx_Val_Volts,""));
DAQmxErrChk    (DAQmxCreateAOVoltageChan(taskHandle,"Dev1/ao1","",0.0,5.0,
DAQmx_Val_Volts,""));
DAQmxErrChk (DAQmxStartTask(taskHandle));
DAQmxErrChk  (DAQmxWriteAnalogF64(taskHandle,1,1,10.0,DAQmx_Val_ GroupBy
Channel, data,NULL,NULL));
DAQmxStopTask(taskHandle);
DAQmxClearTask(taskHandle);
//Чтение с канала АЦП
DAQmxErrChk (DAQmxCreateTask("",&taskHandle2));
DAQmxErrChk  (DAQmxCreateAIVoltageChan(taskHandle2,"Dev1/ai0","",DAQmx_
Val_RSE ,-10.0,10.0,DAQmx_Val_Volts,NULL));
DAQmxErrChk  (DAQmxCfgSampClkTiming(taskHandle2,"",10000.0,DAQmx_Val_
Rising,DAQmx_Val_FiniteSamps,2));
DAQmxErrChk (DAQmxStartTask(taskHandle2));
DAQmxErrChk  (DAQmxReadAnalogF64(taskHandle2,1000,10.0,DAQmx_Val_ Group
ByChannel,datad,1000,&read,NULL));
```

```
        DAQmxStopTask(taskHandle2);
        DAQmxClearTask(taskHandle2);
        // Конфигурация и запуск задачи цифрового выхода
        // используется для управления мультиплексором
        DAQmxErrChk (DAQmxCreateTask("",&taskHandle4));
        DAQmxErrChk (DAQmxCreateDOChan(taskHandle4,"Dev1/port1/line0:3","",
DAQmx_ Val_ChanForAllLines));
        DAQmxErrChk (DAQmxStartTask(taskHandle4));
        DAQmxErrChk (DAQmxWriteDigitalLines(taskHandle4,1,1,10.0,DAQmx_Val_
GroupByChannel,mdio3,NULL,NULL));
        DAQmxStopTask(taskHandle4);
        DAQmxClearTask(taskHandle4);
        // вывод значения со входа АЦП на номер контакта mnoutadc
        as[mnoutadc[0]].ANew=datad[0];
        // вывод сообщения об ошибке при возможной неправильной конфигурации
        Error:
                if( DAQmxFailed(error) )
                        DAQmxGetExtendedErrorInfo(errBuff,2048);
                if( taskHandle!=0 ) {
                        /*********************************************/
                        // DAQmx Stop Code
                        /*********************************************/
                }
                if( DAQmxFailed(error) )
                        printf("DAQmx Error: %s\n",errBuff);
        }
```

Подсистема ADCUSB NIDAQ 6009 является средней по оценкам критерия эффективности – производительности. Передача параметров возможна в виде структуры данных и списка номеров цепей. Использование списка номеров цепей удобнее при описании систем в виде формализованных заданий. Внешние мультиплексоры расширяют функциональные возможности подсистем и повышают эффективность.

* * *

Созданы методология и алгоритмы работы обобщенных многофункциональных моделей компонентов с общим интерфейсом. Новые многофункциональные модели компонентов с общим интерфейсом связаны с общими данными для различных моделей и приложений и позволяют из одного описания объекта получить множество приложений. Многофункциональные модели компонентов с общим интерфейсом значительно снижают трудоемкость проектирования, в особенности на начальных этапах.

4. РЕАЛИЗАЦИЯ МНОГОУРОВНЕВОЙ СИСТЕМЫ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

4.1. Структура и функции программного комплекса

Подсистема выбора формализованного задания и начальных значений параметров **SETSEL** представляется в виде

SETSEL=<SETFZ, SETLD, SETLM, SETLIB, SETCADIN, SETSERV, SETLOC>,

где **SETFZ** – множество имен формализованных заданий;

SETLD – множество допустимых языков описания;

SETLM – множество допустимых языков сообщений;

SETLIB – множество допустимых видов библиотек (статические – lib, динамические – dll);

SETCADIN – множество допустимых описаний для входных САПР;

SETSERV – множество доступных серверов для просмотра материалов или выполнения заданий;

SETLOC – имя диска (буква) размещения каталогов системы (COD, CODOS), примеров и библиотек объектов (FA) комплекса.

Подсистема выбора результатов проектирования **PRJSEL** представляется в виде

PRJSEL=<SETCADOUT, SETINTF>,

где **SETCADOUT** – множество допустимых выходных САПР;

SETINTF – множество используемых интерфейсов для выходных САПР.

Подсистема выбора представления результатов проектирования **RESSEL** представляется в виде

RESSEL=< VIEWTXT, VIEWAD, VIEWSCH, VIEWMOD, VIEWNET>,

где **VIEWTXT** – программа просмотра сообщений для одного или множества вариантов;

VIEWAD – множество программ просмотра диаграмм цифровых и аналоговых сигналов для многовариантного анализа;

VIEWSCH, VIEWMOD, VIEWNET – множество программ отображения схем, модулей и сетевых объектов;

Множество правил проектирования **RPRJ** состоит из следующих подмножеств:

RPRJ=<RNAMEVAR, RMCADIN, RMCADOUT, RFTSCH, RFTAB>

где **RNAMEVAR** – правила образования вариантов имен результатов проектирования;

RMCADIN – правила выбора модулей и функций заполнения таблицы варианта схемы;

RMCADOUT – правила выбора модулей и функций создания варианта схемы конкретной САПР;

RFTSCH – правила выбора модулей и функций извлечения данных из таблицы варианта схемы;

RFTAB – правила выбора модулей и функций извлечения данных из таблицы информации о компонентах схемы.

В процессе проектирования формализованное задание на языке высокого уровня, представляющее множество технических решений, преобразуется в текстовую или табличную форму уровня одного варианта решения, которая может быть преобразована в текстовый, командный или табличный формат конкретной САПР или в формализованное задание [12–14]. Количество вариантов определяется формализованным заданием.

Процесс преобразования формализованного задания в описание для конкретной промышленной САПР с помощью программно-методического комплекса COD происходит следующим образом. Для каждого варианта описания формируется таблица TSCH (см. рис. 4.1). Формализованное задание транслируется и редактируется с соответствующими моделями процедур, объединенными в библиотеку. При выполнении формализованного задания модель конкретного компонента вызывает обращение к информационной таблице и помещение информации о компоненте в таблицу варианта схемы (см. рис. 4.2). Связи или соединения заполняются в таблице TSCH из формализованного задания. Таким образом, в таблице варианта схемы (см. табл. 4.6) оказывается вся информация о компонентах, выводах компонентов и связях с другими компонентами. После формирования TSCH включаются функции формирования описания варианта схемы для конкретной САПР. Функции четвертого уровня (см. рис. 4.1) обеспечивают обработку компонентов, а функции пятого уровня – преобразование информации о выводах компонентов.

Префиксы функций приведены в табл. 4.2, основные функции – в табл. 4.3. Например, функция для открытия таблиц соответствия обозначается DOpen или DbnOpen, а для их закрытия – DClose или DbnClose. Функция открытия файла библиотеки компонентов обозначается LOpen или LibOpen, закрытия – LClose или LibClose. Функция добавления цепей – NetAdd, функция добавления элементов – ElmAdd, функция дополнения схемы – SchAdd. Правила образования имен команд оболочки COD представлены в табл. 4.4, таблица конкретных процедур и функций интерфейса приведена в табл. 4.5.

	1. Формализованное задание (ФЗ)	
2. Модели поиска путей и имен ФЗ (CodRead)	Модели компонентов (Comp)	Модели управляющих процедур и функций генерации сигналов (Control)
3. Структура данных варианта схемы (TSCH)		
4. Методы чтения библиотеки компонентов добавления и записи структуры данных схемы или файла макрокоманд	Методы чтения, записи, поиска в таблице данных варианта схемы (TSCH)	Методы чтения, записи и поиска компонента в таблице соответствия (Table)
5. Методы чтения, добавления и записи структуры данных компонента, его выводов и цепей	Методы чтения, добавления и записи таблиц символов и конструктивов	Методы поиска, чтения и записи информации о контактах компонента и подключенных цепях
Файл проекта конкретной САПР или стандартная структура обмена данными		

Рис. 4.1. Обобщенная структура многоуровневой САПР

Рис. 4.2. Формирование таблицы варианта схемы

Интерфейс TSCH (таблица варианта схемы) представляет собой вывод структуры описания схемы в файл <имя ФЗ>.tsh в текстовом формате. Этот файл может потребоваться в случае отладки модулей, а также если требуется полная информация о компонентах схемы. Аналогичная структурированная в формате XML информация выводится в файл типа xml. Файл типа XML используется для графического отображения схем.

Таблица 4.1

Таблица основных файлов

Имя модуля	Назначение модуля
uiorcad, uipcad	Процедуры и функции нижних уровней
control	Модули управления
Comp	Модели компонентов верхнего уровня
Codread, codini	Функции чтения путей и имен файлов УИ САПР
table	Функции для работы с таблицами компонентов uipcad
Tsch	Функции для работы с таблицами варианта схемы Tsch
Errors	Функции вывода сообщений

Таблица 4.2

Таблица префиксов функций

Префикс функций	Объект
P, Pif	Интерфейс САПР <COD> с промышленными САПР
L, Lib	Библиотека компонентов
D, Dbt, Dbn	Таблица соответствия
TS, TSch	Таблица схемы
TE, Telm	Таблица элементов
Name	Имя объекта
Typ	Тип объекта
S, Sch	Схема
E, Elm	Элемент
N, Net	Цепь
C, Con	Вывод
Pin	Контакт
Msg, Err	Сообщения об ошибках
PinList	Список контактов
NetList	Список цепей

Таблица 4.3

Таблица основных функций

Имя функции	Назначение функций обработки объектов
Open	Открытие файлов
Close	Закрытие файлов
Srch, Find	Поиск файла, поиск в таблице
Copy	Копирование файлов, строк в таблице
Update, Upd	Обновление файлов, строк в таблице
Add	Добавление записей в файл, строки в таблицу, цепи к компоненту
Del	Удаление записей, контактов, цепей
Draw	Добавление проводника, шины

Таблица 4.4

Правила образования имен команд оболочки COD

Назначение символа	Номер	Значение	Семантика значения
Способ выполнения формализованного задания	1	F	Локальное
		R	Сервер OS/2
		P	Сервер P390
		H	Сервер S390
		W	Сервер WWW
		L	Создание LIB
		I	Создание DLL
		M	Модуль общего интерфейса
Язык описания формализованного задания	2	P	PL/1 (PLI)
		C	C++ (CPP)
		J	JAVA
		A	ADA
		H	VHDL
Тип операционной системы	3	D	DOS
		O	OS/2
		W	WINDOWS
		V	VM
		H	MVS (OS390)
		U	UNIX
		L	LINUX
Пакет САПР для конкретной сферы	4	F	Модели функциональные
		G	Общий интерфейс
		R	PRAM5.3
		P	PCAD
		O	ORCAD
		C	CADDY
		A	AUTOCAD, EAGLE
		V	VRML
		E	EDIF
		EI	Eagle командный интерфейс
		S	STEP (EXPRESS)
		X	XML
		Z	TSCH
		V	VRML
		I	CATIA
		L	ALTIUM
Версия САПР	5, 6	Цифры	Номер версии
Тип моделей (интерфейс с САПР)	5, 6 или 7	F	Модели функциональные
		I	Модели командного интерфейса
		T	Модели табличного интерфейса
		C	Описание по компонентам
		N	Описание по цепям
		Z	Синтез FZ COD

Таблица правил образования имен команд оболочки COD (табл. 4.4) используется для локальных и удаленных комплексов и относится к базовой модели [87], хотя используется и в информационной модели. Локальное выполнение функционального моделирования и оценки ресурсов в среде Windows производится командами fcwf.bat, fpwf.bat и fjwf.bat для языков описания C++, PL/1 и JAVA, для среды OS2 или ECS – fcof.cmd, fpof.cmd, fjof.cmd. Сервер приложений использует локальные команды, например fcop8t.cmd и fcop8i.cmd, для табличного и командного интерфейса с САПР PCAD8.5 соответственно. Для САПР CATIA v5 командные файлы формирования файлов макрокоманд будут fcoi.cmd, fpoi.cmd и fjoi.cmd.

Команды формирования статических и динамических библиотек моделирования – lcof.cmd, icof.cmd, команды формирования библиотек объединенного интерфейса – lcogi.cmd, icogi.cmd.

Таблица 4.5

Таблица процедур и функций интерфейса

Имя объекта	Назначение объекта	Вызов объектов (процедур, функций)	Возвращаемое значение
Формализованное задание (ФЗ)	Описание одного или множества технических решений	Модели внешних воздействий	–
		Модели компонентов	
		Управляющие модули	
Модели внешних воздействий	Описание внешних воздействий	ElmCopy	–
		NetAdd	
Модели компонентов	Описание моделей компонентов	ElmCopy	–
		NetAdd	
Модели управляющих процедур	Описание модулей управления для приложений	DbnClose	Номер раздела ФЗ
		PifClose, PClose	
PifOpen, POpen	Начальная установка путей и объектов при первом вызове (comp, signal)	–	–
PifClose	Проверка завершения создания таблицы схемы и вывод результатов	TSClose	–
TSClose	Создание тела файла макрокоманд	CodRead	–
TSOpen	Создание таблицы схемы	–	–
TSAdd	Добавление компонентов в таблицу схемы	–	–

Окончание табл. 4.5

Имя объекта	Назначение объекта	Вызов объектов (процедур, функций)	Возвращаемое значение
CodRead	Поиск имени задания, шаблона чертежа, пути и типы библиотек	CodSrch	0 – не найдено 1 – найдено
ElmCopy	Создание элемента в таблице схемы	DbnOpen ElmSrch ElmAdd AbsName	0 – не создан N – номер элемента в таблице схемы
ElmAdd	Добавление компонента в таблицу схемы	–	0 – не добавлен N – номер компонента в таблице
DbnOpen	Поиск файла таблицы соответствия последовательно uipcad.dbn, uipcad.dbt и загрузка	GetPin DbnAdd	0 – файл не найден 1 – файл найден
DbnClose	Завершение работы с таблицей соответствия и удаление таблицы из памяти	–	–
ElmSrch	Поиск компонентов в таблице соответствия по его имени	–	0 – не найден N – номер записи
PinSrch	Поиск контакта элемента	–	0 – не найден I – индекс
AbsName	Проверка типа компонента на абстрактный тип	–	0 – тип конкретный 1 – тип абстрактный
ErrOpen	Открывает файл сообщений типа msg	–	0 – не открыт 1 – открыт
ErrClose	Закрывает файл сообщений типа msg	–	–
ErrAdd	Вывод сообщений в файл типа msg	–	–
GetPin	Чтение информации о контакте из таблицы соответствия	–	N – позиция контакта
DbnAdd	Увеличение объема таблицы в памяти	–	–

Таблица 4.6

Таблица варианта схемы

Имя	Уровень	Назначение	C++, JAVA	PL/1	Примечание
DbnRec	1	Структура таблиц типа dbm	–	–	–
Nel	2	Номер элемента	Int	Dec Fixed(3)	–

Продолжение табл. 4.6

Имя	Уровень	Назначение	C++, JAVA	PL/1	Примечание
ElmNamR ElmNamL ElmNamA	2	Имя компонента из русских или латинских букв и цифр или имя аналога	char (C++) String (JAVA)	Char(*)	*
DOCNamR DOCNamL	2	Имя документа на компоненты из русских или латинских букв и цифр			*
DOCNamA	2	Имя документа аналога			*
P0	2	Мощность статическая	Float	Dec Float	–
P01	2	Энергия переключения компонента			–
S	2	Площадь компонента			–
Massa	2	Масса компонента			–
Price Rel	2	Цена относительно вентиля двухвходового			–
TYPEElm FunName	2	Тип функции и имя функции компонента	char (C++) String (JAVA)	***Char(*)***	*
FunCode	2	Код функции компонента	Int	Dec Fixed(3)	–
KElmPack	2	Количество элементов в корпусе		Dec Fixed(3)	–
TYPEPack	2	Тип корпуса	char (C++) String (JAVA)	Char(*)	–
KPinPack	2	Количество контактов в корпусе	Int	Dec Fixed(3)	–
NElmPack	3	Номер элемента в корпусе		Dec Fixed(3)	–
MPinPack	2	Массив контактов		Dec Fixed(3)	**
KPinIn	2	Количество входных контактов		Dec Fixed(3)	**
KPinOut	2	Количество выходных контактов		Dec Fixed(3)	**
MPinIn	2	Массив входов		Dec Fixed(3)	**
MPinOut	2	Массив выходов		Dec Fixed(3)	**
PinListE	2	Список выводов элемента	EPinList		–

Окончание табл. 4.6

Имя	Уровень	Назначение	C++, JAVA	PL/1	Примечание
PinType	3	Тип контакта	Int	Dec Fixed	–
PinLeq	3	Тип логической эквивалентности		Dec Fixed(3)	–
PinPCAD	3	Имя вывода в САПР PCAD	char (C++) String (JAVA)	Char(*)	–
PinCod	3	Имя вывода в COD		Char(*)	–
PinN	3	Номер контакта	Int	Dec Fixed(3)	–
PinName	3	Имя контакта	char (C++) String (JAVA)	Char(*)	*
NetTYP	3	Тип цепи, соединенной с контактом		Char(*)	*
NetNum	3	Номер цепи	Int	Dec Fixed(3)	–

* Отсутствию компонента соответствует слово null (строчными буквами).

** Поле в файл не записывается и допустимо только в оперативной памяти.

Входными цифровыми сигналами управляют с помощью параметров процедуры формирования SIGNAL или SIGNALD, вычислением текущих битовых строк или массива строк. Целью управления входными сигналами может быть оценка максимальной частоты работоспособности устройства. Цель содержит противоречивые требования максимальной производительности (частоты сигналов) и достоверности результатов. Поэтому возможна реализация алгоритма управления изменением частоты сигналов, начиная с максимальной, и анализом массива оценок результатов в соответствии с предполагаемыми значениями. Знания представляются соответствием правил формирования сигналов и предполагаемых результатов. Следовательно, соответствие фактических и предполагаемых результатов является условием достижения цели и завершения процесса. Если условие достижения цели не выполнено, то анализ вариантов завершается принудительно по заданному максимальному значению. Для оценки достоверности результатов лучше пользоваться различными видами входных сигналов.

Можно оперативно управлять текущей битовой строкой генерируемого сигнала или заполнять массив битовых строк. Использование внешних моделей сигналов возможно с однократным или многократным вводом из файла. Однократный ввод снижает время, но увеличивает объем буфера моделей. Примеры работы с буферами моделей приведены в примерах.

Входными аналоговыми сигналами управляют с помощью параметров процедуры формирования SIGNALA, а сравнивают фактические и предполагаемые аналоговые сигналы с помощью процедуры SIGNALAC. Первым параметром процедуры SIGNALA указывается номер цепи аналогового сигнала. Второй параметр типа float может быть значением или выражением. Третий и четвертый параметры определяют начало и конец изменения аналогового сигнала. При отсутствии дополнительных ограничений в качестве

третьего параметра используют номер такта минимальный NTMIN, в качестве четвертого параметра – номер такта максимальный NTMAX.

Номер анализируемого варианта можно использовать в качестве одного из индексов двумерного массива моделей сигналов, что обеспечит удобную форму многовариантного анализа. Для одновариантного и многовариантного анализа можно использовать любой тип монитора: IPRINT, CPRINT или APRINT. Управлять структурой системы можно статически (до начала ее работы) и динамически (в процессе функционирования). Модульная структура вычислительных машин и систем позволяет изменять конфигурацию до начала работы. Для изменения связей в процессе работы необходимо вводить специальные компоненты, осуществляющие коммутацию, – мультиплексоры и демультиплексоры. В многопроцессорных вычислительных системах коммутационные компоненты являются основой согласования структуры системы с алгоритмом решения задачи.

Средства языков высокого уровня позволяют в качестве типов компонентов и связей вводить переменные с множеством допустимых значений. Таким образом можно реализовать представление дерева технических решений.

В процессе проектирования необходимо управлять выходной информацией САПР. В рамках исследовательской системы в разделе управления анализом для вывода цифровых сигналов используется управляющий модуль IPRINT или CPRINT с параметрами минимального и максимального номеров цепей и границами временного интервала, для вывода цифровых и аналоговых сигналов – модуль APRINT с параметрами нижних и верхних границ цифровых цепей, временного интервала и аналоговых цепей. Вывод цифровых или аналого-цифровых сигналов управляется форматом при обращении к любому управляющему модулю. Переменная LRES определяет варианты выходных документов. Значение 0 соответствует графическому выводу результатов, 1 – формированию символьной временной диаграммы работы и таблице оценки ресурсов, 2 – дополнительно выводится таблица сравнения сигналов, 3 – дополнительно формируются спецификации всех вариантов, 4 – дополнительно выводится таблица состояний и 5 – дополнительно вводится возможность оптимизации.

Модель пользователя представлена структурой LU с различными значениями: начальным, старым и новым. Значению 0 соответствует запрет управления процессом проектирования, 1 – возможность одновариантного анализа, 2 – возможность многовариантного анализа, 3 – управление внешними воздействиями, 4 – изменение структуры и 5 – оптимизация.

Основной задачей при разработке исследовательских САПР было однократное описание объекта для различных приложений. Различные приложения выбираются из меню и выполняются при неизменном основном описании. Таким образом экономится труд инженера в процессе проекти-

рования. Примерами различных приложений являются: моделирование с оценкой ресурсов, формирование спецификаций, преобразование описания в схемный файл системы PCAD, импорт описаний из различных САПР. Маршруты проектирования в САПР COD в процессе выполнения различных приложений приведены на рис. 4.3.

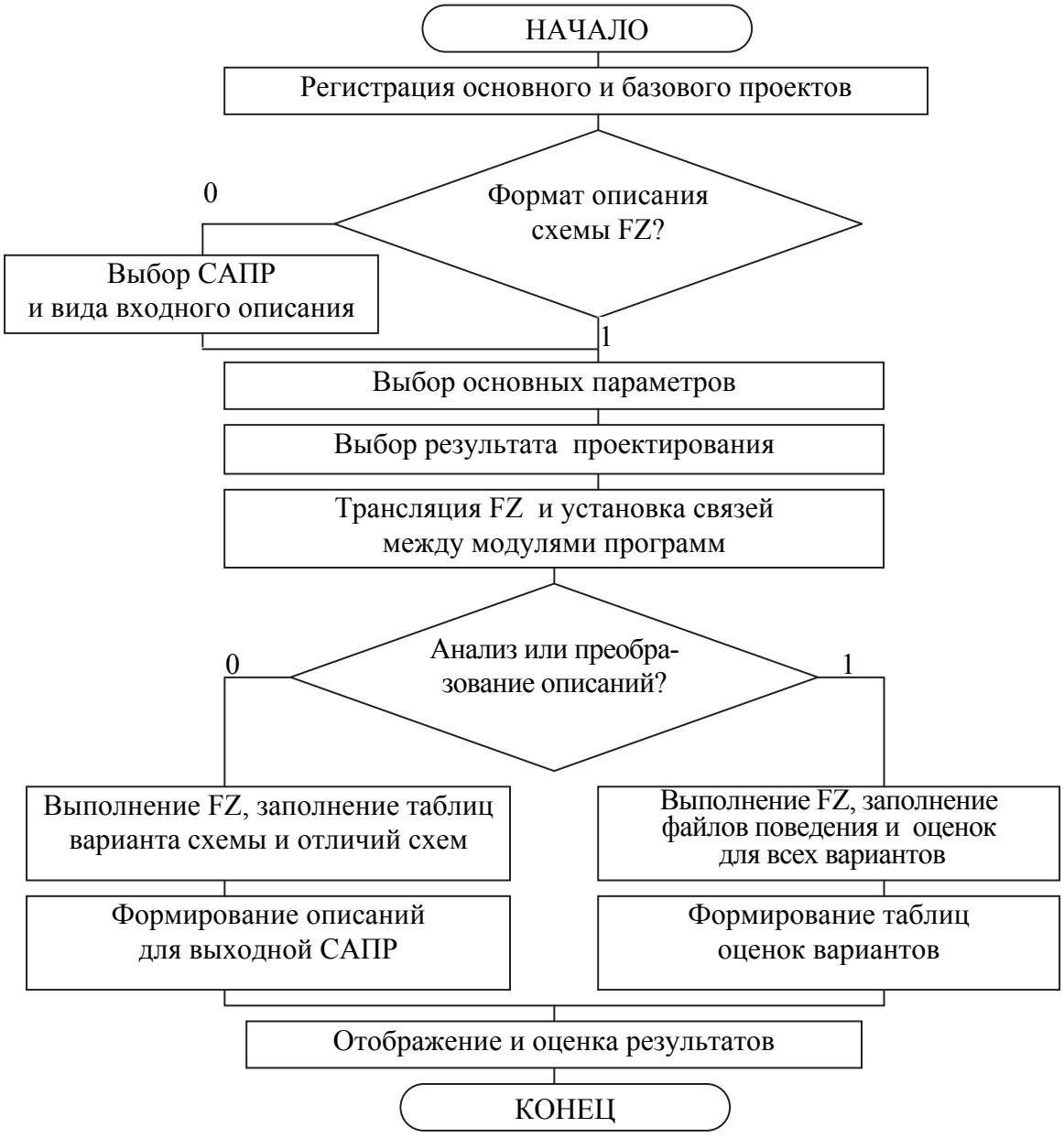


Рис. 4.3. Маршруты проектирования в САПР COD
(FZ – формализованное задание)

Важным является анализ вычислительных устройств с неисправностями компонентов в различных вариантах.

Синтез описаний неоднородных вычислительных систем и выполнение различных приложений обеспечивается различными компонентами САПР, в том числе программным и информационным.

Рассмотрим основы информационного обеспечения на примере исследовательской САПР.

4.2. Информационное обеспечение моделирования и проектирования вычислительных устройств и систем

Состояния сигналов в процессе анализа представляются значениями структур данных. Тип данных зависит от мощности множества состояний. Аналоговым сигналам соответствуют действительные числа, двоичным сигналам – один бит. Однако двоичные модели сигналов используют для анализа только на логическом уровне. С увеличением мощности множества состояний сигналов подробнее анализируется переходный процесс, но значительно увеличивается время анализа. Поэтому используют минимальные по сложности модели сигналов, достаточные для достижения цели анализа. Известно использование пятизначных и даже девятизначных моделей цифровых сигналов [31, 84], но чаще всего используют троичные.

Двоичные модели сигналов в синхронных моделях служат для обнаружения неправильной работы устройства, вызванной грубыми ошибками синтеза. Троичные модели позволяют [31, 113, 114] обнаружить статический риск сбоя, заключающийся в ошибочном изменении сигнала на выходе элемента за счет задержек во входных сигналах.

Динамический риск сбоя [31, 113] соответствует многократному изменению выходного сигнала вместо однократного и может вызываться неблагоприятными сочетаниями задержек входных сигналов или резонансом во входной цепи на уровне порога срабатывания. Пятизначное представление значений сигналов позволяет обнаружить динамический риск сбоя. Примеры решения системы логических уравнений в двоичном и троичном базисах приведены в [31]. Пример системы логических уравнений дан для схемы на рис. 4.4. Уравнения составлены в порядке, обратном распространению сигналов:

```
NEW(010)=NEX(008)!NEX(009);
NEW(009)=^(NEX(007)&NEX(005));
NEW(008)=NEX(006)&NEX(007);
NEW(007)=NEX(004)&NEX(005);
NEW(006)=NEX(002)!NEX(003);
```

Глубина логической схемы KDEL, или ранг, равен трем. Поэтому максимальное число итераций не превышает KDEL+1, т. е. четырех. Дополнительная итерация необходима для проверки устойчивости результатов. Структура данных представления цифровых и аналоговых сигналов содержит поля новых (NEW) значений и результатов предыдущих итераций (NEX). Разделение полей значений сигналов позволяет реализовать принцип единственного присваивания и исключить влияние числа итераций на результат. Запись производится в поля новых значений (NEW), а чтение из полей (NEX) – в соответствии с номером цепи. Поэтому описание любой логической схемы может быть составлено в произвольном порядке.

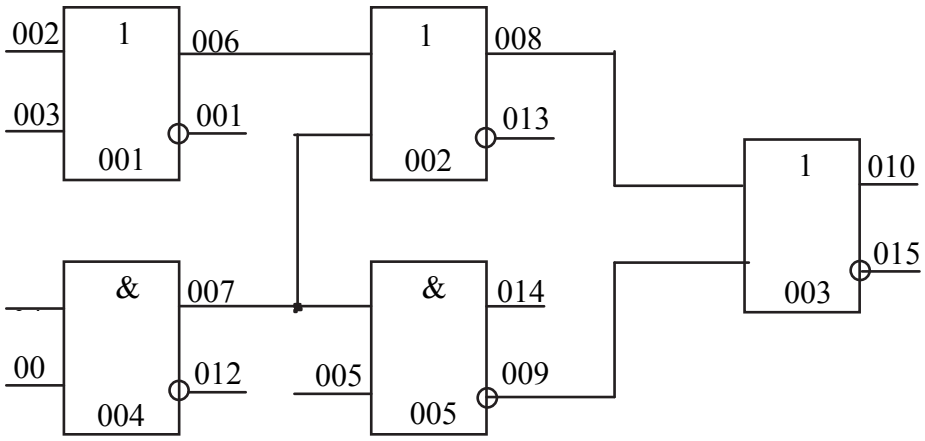


Рис. 4.4. Логическая схема для расчета итерационного процесса

С целью сокращения времени анализа следует стремиться описывать схемы по ходу распространения сигналов, для чего достаточна обычная практика оформления схем слева направо и сверху вниз.

В многоуровневых САПР на верхних уровнях абстракции используются обобщенные модели компонентов, которые конкретизируются по мере перехода к нижним уровням. Для перехода от абстрактных моделей к конкретным требуется дополнительная информация. Рассмотрим преобразование описаний при переходе с функционального уровня к конструкторскому.

На функциональном уровне основной является функция, выполняемая компонентом или узлом вычислительного устройства, а на конструкторском уровне – геометрическая модель компонента, которая включает модель корпуса, пространственное расположение и нумерацию контактов. На функциональном уровне результат преобразования информации, подходящей по множеству входных цепей, поступает на выходные цепи. При этом пространственное расположение цепей не имеет значения, но уровень помех влияет на надежность преобразования цифровой информации и определяет погрешность работы аналоговых и аналого-цифровых узлов.

Для автоматического преобразования описаний вычислительных устройств функционального уровня в описания конструкторского уровня конкретной САПР необходимо программное и информационное обеспечение. Информационное обеспечение может быть представлено в форме таблиц или баз данных. В УИ САПР и пособии [33] файлы баз данных имеют тип db или dbf, а файлы таблиц – тип dbt, dbn или dbm. В файле типа dbm наряду с кодом функции компонента для САПР PCAD содержится имя функции и тип функции, который соответствует имени процедуры модели компонента. Базы данных и таблицы содержат идентичную информацию о компонентах, но отличаются форматом данных [28]. Таблицы свойств компонентов предназначены для информационного обеспечения формирования символов компонентов, автоматического преобразования варианта формализованного задания в схему конкретной САПР, импорта и экспорта структуры обмена [1]. Поэтому таблицы типа dbt в описании выводов содержат только тип вывода, имя вывода в САПР PCAD и имя вывода в УИ САПР COD. Формальное описание вывода (ОВ) представляется в виде

<ОВ>::=<тип вывода> <имя вывода в САПР PCAD>
< имя вывода в УИ САПР COD>.

Таблицы типа dbn или dbm предназначены для информационного обеспечения формирования символов и конструктивов компонентов и автоматического преобразования варианта формализованного задания в схему или описание конструкторского уровня конкретной САПР. Формальное описание вывода для таблицы типа dbn представляется в виде

<ОВ>::=<тип вывода> <группа эквивалентности> <номер контакта>
<имя вывода в САПР PCAD>
< имя вывода в УИ САПР “COD”>.

Формальное описание вывода для таблицы типа dbm имеет вид

<ОВ>::=<тип вывода> <группа эквивалентности><номер контакта>
<имя вывода в САПР PCAD>
< имя вывода в УИ САПР “COD”>
<тип цепи> <номер цепи>.

Все вышеприведенные таблицы формально описываются следующим образом:

<Таблица>::=<текст описания множества компонентов (ОМК)>
<ОМК>=::<описание компонента (ОК)>[<ОК>]
<ОК>::=<описание параметров компонента>
<описание выводов компонента (ОВК)>
<ОВК>::=<описание группы выводов (ОГВ)>;[<ОГВ>;…<ОГВ>]
<ОГВ>::=<ОВ>, [<ОВ>,…<ОВ>]

Имена параметров и свойств контактов и цепей приведены в табл. 4.6. Примеры таблиц типов dbt, dbn, dbm находятся в подкаталогах X:\COD\BIN и X:\CODOS\BIN, а также могут находиться в подкаталогах с библиотеками компонентов для промышленных САПР. Для САПР ORCAD версии 9 таким подкаталогом будет X:\FA\O9, для САПР PCAD8.5 подкаталогом будет X:\FA\P8. Две первые строки таблиц отмечены знаком комментария (/), и в них размещены заголовки столбцов.

Следует обратить внимание на описание компонентов, для которых функции отдельных секций логических элементов имеют различные или эквивалентные функции. В этом случае выводы питания компонента описываются не в каждой секции, а в последнем описании группы выводов. Описания группы выводов отдельных секций разделяются точкой с запятой, и их количество не ограничивается. После последней секции точка с запятой может отсутствовать.

В САПР COD на функциональном уровне выводы питания не используются, но они необходимы для перехода к конструкторскому уровню. Для отличий выводов питания приняты правила обозначений: 0V – цифровой общий проводник; 0VA – аналоговый общий проводник; UP, UP1, UP2 – вывод положительного напряжения питания в порядке возрастания; UN, UN1, UN2 – вывод отрицательного напряжения питания в порядке возрастания по абсолютной величине.

Описания вычислительных устройств на языках высокого уровня [9, 14] могут содержать абстрактные и конкретные типы компонентов. Конкретные типы компонентов характеризуются выполняемой функцией, ресурсами и параметрами, упакованы в корпус. Абстрактные типы компонентов характеризуются обобщенной или конкретной функцией. Примером абстрактного компонента является мультиплексор аналоговых и цифровых сигналов, последовательный или параллельный сумматор произвольной разрядности. Существующие промышленные САПР, как правило, работают только с конкретными типами компонентов. Примерами таких САПР являются ПРАМ5.3 и PCAD (MD). Объем описаний вычислительных устройств на языках низкого уровня в графической или текстовой форме значительно превышает аналогичное описание на языках высокого уровня. Поэтому эффективно описывать технические решения на более высоком уровне абстракции, на языках высокого уровня, и автоматически преобразовывать их в описания для промышленных САПР. Рассматривается реализованный интерфейс исследовательской САПР с промышленными САПР, особенности программного и информационного обеспечения.

4.3. Формализованное задание для автоматизированного анализа

Формализованное задание для автоматизированного анализа состоит из следующих разделов:

INIT – объявление и установка начальных значений параметров и компонентов;

INPUT – ввод внешних сигналов;

UNIT – описание схемы;

MOD – внутренние процедуры – модели компонентов;

CTRL – управляющий модуль.

Особенностью формализованного задания является обязательная нумерация всех цепей, которые могут быть объявлены. Одиночные или групповые цепи разрешено именовать.

Раздел объявления и установки начальных значений параметров содержит общую часть, которая включается препроцессором с помощью операторов %INCLUDE – разделы TXTPAR. Индивидуальные параметры для конкретного задания можно изменять с помощью операторов присваивания, начиная с метки INIT. Например, записывают необходимое количество сигналов – тринадцать и максимальное число тактов – шестнадцать:

INIT: NS=13; NTMAX=16;

затем объявляют массивы структур аналоговых и цифровых сигналов, регистров – %INCLUDE TXTDCL. В этом же разделе объявляют все одиночные и групповые сигналы, индивидуальные для конкретного задания. Например, объявляют битовую строку – модель входного сигнала VXOD – DCL VXOD BIT(16). Началом раздела ввода внешних сигналов является метка INPUT, а внешние сигналы формируются с помощью специальных процедур SIGNAL и SIGNALA.

Если изменение сигнала (разность между четвертым и третьим параметром) превышает длину битовой строки, то сигнал периодически повторяет последовательность, заданную битовой строкой. Генерация сигналов подробно объясняется в отдельном разделе. Аналоговые сигналы генерируются с помощью стандартных функций.

Раздел описания схемы начинается с метки UNIT. Описание схемы производится покомпонентно путем вызова обобщенных процедур класса компонентов. Имена процедур для всех мультиплексоров – MX, для логических схем – LO. Конкретный компонент идентифицируется типом – символьной строкой. Тип мультиплексора – '564КП1', тип логической схемы – '564ЛА7', или их аналогов, соответствующих библиотеке. Связи описывают

покомпонентно путем задания значений параметров цепей. Например, описания компонента и соединений на языке PL/1 имеют вид:

```
UNIT: CALL LO('564ЛЕ10',001,006,009,007,003);
      CALL LO('564ЛА7' ,002,007,011,004,002);
```

на языках С и С++:

```
UNIT: LO('564ЛЕ10',001,006,009,007,003);
          LO('564ЛА7' ,002,007,011,004,002);
```

на языке JAVA:

```
case 3: { //UNIT:
comp.LO('564ЛЕ10',1,6,9,7,3);
comp.LO('564ЛА7' ,2,7,11,4,2);}
```

различные примеры приведены в [27, 31].

Раздел внутренних процедур – моделей компонентов – необходим при включении новых элементов или неадекватности существующих моделей. Параметрами процедуры являются номера цепей. Программная модель компонента выполняет действия над структурами данных с индексом массива, соответствующим номеру цепи. В процедурах предусматривают входной и выходной контроль запрещенных кодовых комбинаций значений сигналов, например '10'B [31, 112–114]. Каждую внутреннюю процедуру проверяют тестовыми примерами.

Раздел управляющего модуля состоит из процедур IPRINT для вывода цифровых сигналов и модуля APRINT для вывода цифровых и аналоговых сигналов. Параметры процедуры IPRINT:

1) минимальный номер цепи, начиная с которой сигнал выводится на печать;

2) максимальный номер цепи, сигнал которой предполагается вывести на печать;

3) минимальный номер такта, с которого сигнал выводится на печать;

4) максимальный номер такта, для которого сигнал выводится на печать.

С целью снижения времени трансляции управляющий модуль подключается редактором из библиотеки модулей.

Включаемый текст W является условным оператором: он используется для управления итерационным процессом автоматизированного анализа и передает управление на ввод внешних сигналов (INPUT), начальную установку (INIT) или описание схемы (UNIT). В библиотеке исходных включаемых текстов имена операторов выбора точек входа соответствуют име-

нам процедур. Например, при анализе логических схем используют оператор %INCLUDE WLO. Для языка PL/1 все объявления точек входа процедур описаны в файле WALL.INC, а для языка C++ все определения функций и процедур описаны в файле UICPP.H.

4.4. Параметры САПР и структуры данных

С целью использования минимальных ресурсов оперативной памяти и машинного времени введены параметры, управляемые пользователем. Текст параметров (TXTPAR) при использовании только статических библиотек следующий:

```
DCL(NS INIT(64),NAS INIT(09),LRG INIT(8), KAPT INIT(8),
NM INIT(1), NRG INIT(16))EXT DEC FIXED(3);
DCL(NT INIT(0),NPT,NTMIN INIT(1),
NTMAX INIT(32))EXT DEC FIXED(6);
DCL(TTIME INIT(0),DELT,DELTP INIT(400),
DELTN INIT(600))EXT DEC FLOAT;
DCL(M_S INIT('1'B),PPT)BIT(1)EXT,ERC CHAR(1)EXT;
DCL LAB(3) LABEL INIT(INIT,INPUT,UNIT), (NL INIT(0),
LRES INIT(1), LU INIT(2)) DEC FIXED(3) EXT;
DCL (C0 INIT('00'B), CF INIT('01'B), CZ INIT('10'B),
C1 INIT('11'B)) EXT BIT(2);
DCL(XMAX, YMAX) EXT DEC FLOAT;
```

Текст включается в формализованное задание с помощью препроцессорного оператора %INCLUDE TXTPAR или TEXTPAR, который должен находиться до текста объявлений, включаемого оператором %INCLUDE TXTDCL или TEXTDCL.

Текст объявлений на языке PL/1 при использовании статических библиотек имеет вид

```
DCL RG(NRG)EXT CTL BIT(LRG);
ALLOCATE RG;RG='0'B;
DCL RGM(NRG,NM)EXT CTL BIT(LRG);
ALLOCATE RGM;RGM='0'B;
DCL 1 OUT(0:NS)EXT CTL,2((NEW,NEX,OLD,OLS,FUT)
BIT(2), DELD DEC FIXED(3)); ALLOCATE 1 OUT;
```

```
OUT='00'B;OUT(001)='11'B; OUT.DELD=0;
DCL 1 AS(0:NAS)EXT CTL,2((ANEW,ANEX,AOLD,AOLS,AFUT)
DEC FLOAT,DELD DEC FIXED(3));
DCL 1 AM(0:NAS,KAPT)EXT CTL,2((ANEW,ANEX,AOLD,
AOLS,AFUT) DEC FLOAT,DELD DEC FIXED(3));
ALLOCATE 1 AS, 1 AM; AS=0; AM=0;
DELT=DELTP+DELTN; NT=0;TTIME=0;
DCL SIGNAL ENTRY (DEC FIXED(3),BIT(*),
DEC FIXED(6), DEC FIXED(6)) EXT;
DCL TCC(KCC,KCT)EXT CTL DEC FIXED(3);
ALLOCATE TCC;TCC=0; TCC(*,3)=NTMIN;TCC(*,4)=NTMAX;
DCL TPC(NS,KCT)EXT CTL DEC FIXED(3);
ALLOCATE TPC;TPC=0;
DCL(GG,GC,GS,GM,GP0,GP01,GX,GY)
INIT(0) EXT DEC FLOAT;
GG=0;GC=0;GS=0;GM=0;GP0=0;GP01=0;GX=0;GY=0;PP=-1
```

При использовании как статических, так и динамических библиотек все объявления выполняются в тексте параметров (TXTPAR.INC), а текст объявлений TXTDCL.INC состоит из единственной строки вызова процедуры размещения объектов в памяти:

```
CALL ALLOUT;
```

Текст процедуры находится в файле ALLOUT.INC и приведен ниже:

```
/* ТЕКСТ ОБ'ЯВЛЕНИЙ ДЛЯ МНОГОВАРИАНТНОГО АНАЛИЗА */
 ALLOUT:PROC;
 ALLOCATE RG; RG='0'B;
 ALLOCATE RGM;RGM='0'B;
 ALLOCATE 1 OUT; OUT='00'B;OUT(001)='11'B;OUT.DELD=0;
 ALLOCATE 1 AS,1 AM;AS=0;AM=0;
 PPT=^M_S;NPT=1+^M_S;DELT=DELTP+DELTN;
 NT=0;TTIME=0;XT=TTIME/DELT;JCC=1;
 ALLOCATE TCC;TCC=0;
 TCC(*,3)=NTMIN;TCC(*,4)=NTMAX;
 ALLOCATE TPC;TPC=0;
 DCL (MPP(NVARMAX),MKEFC(NVARMAX),
 MKEFM(NVARMAX),MKEFP(NVARMAX))
 CTL EXT DEC FLOAT;
 IF NVAR=1 THEN ALLOCATE MPP,MKEFC,MKEFM,MKEFP;
 GG=0;GC=0;GS=0;GM=0;GP0=0;GP01=0;GX=0;GY=0;PP=-1;GPF=0;
 END ALLOUT;
```

Текст параметров TXTPAR.INC при использовании как статических, так и динамических библиотек имеет вид

```
/* Текст объявлений для внешних процедур DLL */
 DCL(NS,NAS,NRG,NM,KAPT,LRG,ERC,NL,NPPD,NPPA)EXT   DEC
FIXED(3)
 RESERVED (IMPORTED);
 DCL(NT,NPT,NTMIN,NTMAX)EXT DEC FIXED(6) RESERVED
 (IMPORTED);
 DCL(TTIME,XT,DELT,DELTP,DELTN,PP,LRA,ERA)EXT
 DEC FLOAT RESERVED (IMPORTED);
 DCL(NVAR,NVARMAX,KCT,KCC,JCC,KPC,LRES,LU)EXT
 DEC FIXED(3) RESERVED (IMPORTED);
 DCL(M_S,PPT)EXT BIT(1);
 DCL RG(NRG)EXT CTL BIT(LRG);
 DCL RGM(NRG,NM)EXT CTL BIT(LRG);
 DCL ( C0 INIT('00'B),CF INIT('01'B),CZ INIT('10'B),
    C1 INIT('11'B)) EXT BIT(2);
 DCL 1 OUT(0:NS)EXT CTL,2((NEW,NEX,OLD,OLS,FUT)BIT(2),
    DELD DEC FIXED(3));
 DCL 1 AS(0:NAS)EXT CTL,2((ANEW,ANEX,AOLD,AOLS,AFUT)
    DEC FLOAT,DELD DEC FIXED(3));
 DCL  1  AM(0:NAS,KAPT)EXT  CTL,2((ANEW,ANEX,AOLD,AOLS,
AFUT)  DEC FLOAT,DELD DEC FIXED(3));
 DCL(TCC(KCC,KCT),TPC(NS,KCT))EXT CTL DEC FIXED(3);
 DCL(GC,GS,GG,GM,GP0,GP01,GX,GY,XMAX,YMAX)EXT      DEC
FLOAT ;
 DCL (GPF,KEFC,KEFM,KEFP)  EXT DEC FLOAT;
```

Текст объявлений для многовариантного анализа на языке C++ находится в файле TXTDCL и приведен ниже:

```
   if (nrg!=0)
     rg = new BitString[nrg];
   {
 register int i;
     for ( i=0 ; i<nrg ; i++ )
       { rg[i].value=NULL;
         rg[i].Alloc(lrg);  }
}
   if (nrg!=0 && nm!=0)
     rgm = new BitString[nrg,nm];
```

```
    {
register int i,j;
       for ( i=0 ; i<nrg ; i++ )
        for ( j=0 ; j<nm ; j++ )
         { rgm[i,j].value=NULL;
           rgm[i,j].Alloc(lrg);     }
}
   out = new DigOut[ns+1];
   memset(out,0,sizeof(DigOut)*(ns+1));
   out[1].New = out[1].Nex = out[1].Old = out[1].Ols = out[1].Fut = C1;
   as = new AnalogOut[nas+1];
   memset(as,0,sizeof(AnalogOut)*(nas+1));
   am = new AnalogArray[nas+1];
   memset(am,0,sizeof(AnalogArray)*(nas+1));
   tcc = new tstr[kcc];
   memset(tcc,0,sizeof(tstr)*kcc);
   tpc = new tstr[ns+1];
   memset(tpc,0,sizeof(tstr)*(ns+1));
   tcas = new tsas[kcas];
   memset(tcas,0,sizeof(tsas)*kcas);
   ppt = !m_s;    npt = 1;
   delt = deltp + deltn;
   jcc=0;// index of table tcc in proc. SignlDC
   gs=gc=gm=gpf=gp0=gp01=gx=gy=0;
   pp=-1;
   if (nvar==1)
   {
mpp=new float[nvarmax];mkefc=new float[nvarmax];
     mkefm=new float[nvarmax];mkefp=new float[nvarmax];
     memset(mpp,0,sizeof(float)*nvarmax);
     memset(mkefc,0,sizeof(float)*nvarmax);
     memset(mkefm,0,sizeof(float)*nvarmax);
     memset(mkefp,0,sizeof(float)*nvarmax);
    }
```

Приведенный выше текст объявлений размещает объекты в оперативной памяти. В формализованное задание входит текст UICPP.H, включающий необходимые заголовочные файлы, прототипы функций моделей компонентов и управляющих модулей, а также тексты объявлений внешних переменных и структур данных TXTPAR и VEXTCPP.H, которые находятся в подкаталоге INCLUDE.

Текст объявлений внешних переменных и структур данных, а также методы их размещения в памяти SetOut() на языке JAVA находятся в файле VEXT.JAVA, который после трансляции помещается в подкаталог COD и в пакеты приложений, находящихся в подкаталоге JAVA:

```
package cod;
import cod.outstruct;
import cod.asstruct;
import java.util.Vector;
public class vext
{
 public static Vector outlist = new Vector();
 public static int lres=-1, nvar = 1, nvarmax = 1, nl = 1, lu, lun, nppd=0,
nppa=0;
 public static boolean m_s=true, ppt;
 public static int nt, ntmax=8, ntmin=0, npt, erc=0, niter=0;
 public static int kct = 8, kcc = 1,jcc=0, ns=0, nm=1, nas=0, lrg=8, nrg=0;
 public static float delt, deltp=1, deltn=1, ttime, xt, pp , lra=-1, era=-1;
 public static outstruct out;
 public static asstruct as;
 public static int tcc[][], tpc[][];
 public final static byte C0 = 0, C1 = 3, CF = 1, CZ = 2;
 public static BitString rg[], rgm[][];
 public static float[] mpp, mkefc, mkefm, mkefp;
public static String bin(byte i)
 {  if(i == C0) return "00";    if(i == C1) return "11";
   if(i == CF) return "01";    return "10"; }
 public static void SetOut()
 {  int n = ns + 1;
  out.New = new byte[n];    out.Nex = new byte[n];
  out.Old = new byte[n];    out.Ols = new byte[n];
  out.Fut = new byte[n];     out.Deld = new int[n];
  out.New[1] =  C1;          out.Nex[1] =  C1;
  out.Old[1] =  C1;          out.Ols[1] =  C1;
  out.Fut[1] =  C1;
  if(nas > 0)
    SetAout();
    SetRg();
  delt = deltp + deltn;   ppt = !vext.m_s;  npt = 1;
  tcc=new int[kcc][kct];  tpc=new int[ns+1][kct];
  mpp = new float[vext.nvarmax+1]; mkefc= new float[vext.nvarmax+1];
```

```
mkefm = new float[vext.nvarmax+1];mkefp= new float[vext. nvar-
max+1]; }
public static void SetAout()
{  int n = nas + 1;
as.ANew = new float[n];  as.ANex = new float[n];
as.AOld = new float[n];   as.AOls = new float[n];
as.Deld = new int[n]; } public static void SetRg()
{   int i,im;
rg = new BitString[nrg+1];
for(i=1;i<nrg+1;i++) rg[i] = new BitString(vext.lrg);
rgm = new BitString[nm+1][nrg+1];
for(im=1;im<nm+1;im++) for(i=1;i<nrg+1;i++)
 rgm[im][i] = new BitString(vext.lrg); }
}
```

Текст объявлений размещает в оперативной памяти массивы структур требуемого объема. Максимальный размер массивов ограничен форматом параметров и реальным объемом оперативной памяти.

Выделяется память для регистров RG длиной LRG и объемом NRG. Количество цифровых сигналов задается параметром NS аналоговых сигналов NAS, длительность установившегося подтакта определяется DELTP, переходного – DELTN. Длительность такта вычисляется как сумма подтактов. Имя NT определяет номер текущего такта, а границы интервала названы NTMIN и NTMAX. Текущее время названо TTIME, номер подтакта – NPT. Троичной модели цифровых сигналов соответствует единичное значение переменной M_S, а двоичной – нулевое. Байт ERROR применяют для индикации уровня ошибки при использовании различных программ. Общий словарь моделей приведен в табл. 4.7.

Состояния цифровых и аналоговых сигналов хранятся в структурах OUT, AS и AM соответственно. Для выполнения правила единственного присваивания разделены имена: новых (NEW), следующих (NEX), старых установившихся (OLD) и будущих (FUT) значений. В процедурах (функциональных моделях компонентов), в левых частях операторов присваивания располагают имена новых значений выходных сигналов, а в правых частях – имена следующих значений для комбинационных схем и различные сочетания имен для автоматов.

Массив меток LAB содержит значения допустимых меток констант INIT, INPUT, UNIT с возможностью расширения, а текущий номер метки определяется переменной NL.

Таблица 4.7

Общий словарь моделей

№ п/п	Имя	Назначение	Форматы данных	
			PL/1	C, CPP, JAVA
1	ANEW	Новое значение аналогового сигнала	Dec Float	Float
2	ANEX	Следующее значение аналогового сигнала		
3	AOLD	Старое значение аналогового сигнала в установившемся подтакте		
4	AFUT	Будущее значение аналогового сигнала в установившемся подтакте		
5	AS	Массив структур аналоговых сигналов	(0:*)EXTCTL STRUCTURE	Analog Out
6	DELT	Длительность такта	Dec Float	Float
7	DELTN	Длительность переходного подтакта		
8	DELTP	Длительность установившегося подтакта		
9	INIT	Описание начальных значений параметров	LABEL CONSTANT	–
10	INPUT	Начало описания внешних сигналов	LABEL CONSTANT	
11	KCC	Количество сравниваемых сигналов	DEC FIXED (3,0)	Int
12	KCT	Размерность таблицы		
13	LAB	Массив меток	(3)AUTOMATIC, INITIAL, LABEL	–
14	LRG	Длина регистра	DEC FIXED (3,0)	Int
15	NAS	Количество аналоговых сигналов	DEC FIXED (3,0)	
16	NEW	Новое значение сигнала	IN OUT(0:*) BIT(2)	unsigned int:2
17	NEX	Следующее значение сигнала		
18	NITER	Номер итерации	DEC FIXED (3,0)	Int
19	NITERMAX	Максимальное количество итераций		
20	NRG	Число регистров		
21	NS	Количество цифровых сигналов		
22	NT	Номер такта	DEC FIXED (6,0)	
23	NTMAX	Номер такта максимальный		
24	NTMIN	Номер такта минимальный		
25	NVAR	Номер варианта		
26	NVARMAX	Номер варианта максимальный		
27	OLD	Старое значение сигнала в установившемся подтакте	IN OUT(0:*) BIT(2)	unsigned int:2
28	FUT	Будущее значение сигнала		

Продолжение табл. 4.7

№ п/п	Имя	Назначение	Форматы данных	
			PL/1	C, CPP, JAVA
29	OUT	Массив структур цифровых сигналов	(0:*),EXT,CTL STRUCTURE	DigOut
30	PPT	Вид подтакта	IN OUT(0:*) BIT(1)	unsigned short int
31	TCC	Таблица сравнения сигналов	(*)CTL,EXT, DEC, FIXED(3,0)	Tstr
32	TPC	Таблица оценки параметров сигналов	(*)CTL,EXT, DEC,	
33	UNIT	Метка описания схемы	LABEL CONSTANT	LABEL
34	C0	Нулевое значение сигнала	C0 BIT(2) INIT('00')	const unsigned C0 = 0
35	C1	Единичное значение сигнала	C1 BIT(2) INIT('11')	const unsigned C1 = 3
36	CF	Значение фронта сигнала	CF BIT(2) INIT('01')	const unsigned CF = 1
37	CZ	Запрещенное значение сигнала	CZ BIT(2) INIT('10')	Const unsigned CZ = 2
38	GC	Стоимость, отн. ед.	DEC FLOAT	FLOAT
39	GS	Площадь, мм кв.		
40	GM	Масса, г		
41	GP0	Мощность на низкой частоте, мВт		
42	GP01	Энергия переключения, мВт · с		
43	GX	Текущая абсцисса		
44	GY	Текущая ордината		
45	GPF	Мощность, мВт		
46	PP	Производительность		
47	KEFC	Критерий «стоимость – производительность»		
48	KEFM	Критерий «масса – производительность»		
49	KEFP	Критерий «мощность – производительность»		
50	DELD	Временная задержка выходного сигнала	DEC FIXED(3,0)	Int
51	AM	Массив структур аналоговых сигналов	(0:*)EXT CTL STRUCTURE	Analog Out
52	NPPD	Номер цифрового сигнала оценки PP	DEC FIXED(3,0)	Int
53	NPPA	Номер аналогового сигнала оценки PP		
54	LRA	Эффективная разрядность		

Окончание табл. 4.7

№ п/п	Имя	Назначение	Форматы данных	
			PL/1	C, CPP, JAVA
55	ERA	Погрешность	DEC FLOAT	Float
56	CADNAME	Выходная САПР	CHAR (*)	*char
57	CADIN	Входная САПР		
58	CODLOC	Имя диска с САПР	CHAR (1)	*char[1]
59	CODE	Имя задания (ФЗ)	CHAR (*)	*char
60	FRAME	Тип формата (А1–А4)	CHAR (2)	*char[2]
61	Language	Язык описания ФЗ	CHAR (*)	*char
62	Langmsg	Язык сообщений		

Объявление точки входа в процедуру генерации сигналов SIGNAL необходимо для использования битовых строк – моделей цифровых сигналов неопределенной длины.

Процедуры модели компонентов и генераторы сигналов вычисляют, значения заносят в элементы массивов структур цифровых и аналоговых сигналов в соответствии с номерами цепей, заданных при вызове процедур. Таким образом моделируется работа взаимосвязанных компонентов различной сложности.

4.5. Интерфейсы исследовательской САПР COD с системой автоматизации проектирования PCAD

Формат данных PDIF системы PCAD (PCAD Data Interchange Format) является символьным аналогом внутренних форматов файлов типа SYM, SCH, PRT, PCB, PS. Форматы файлов PDIF подробно рассмотрены в [134] и используются для преобразования файлов схем и компонентов при переходе от младших версий PCAD к старшим, а также для интерфейсов с другими системами автоматизации проектирования. Например, в САПР PCAD 2000–2006 схемные файлы PCAD версии 8.5 загружаются непосредственно, а схемные файлы PCAD версии 4.5 после преобразования утилитой PDIFOUT.EXE – в формат PDIF в файл типа PDF. Возможно использование PDIF-формата для автоматизированного формирования библиотек. Автоматизация формирования библиотек компонентов позволяет снизить трудоемкость создания библиотек и облегчает использование библиотек для автоматизированного интерфейса исследовательской САПР с PCAD и другими САПР.

Интерфейс исследовательской САПР COD с системой PCAD может осуществляться путем формирования командных файлов для интерактив-

ного графического редактора PCCAPS с последующим его выполнением. Выполнение командных файлов удобно для наблюдения за процессом ввода и размещения символов компонентов и ввода соединений выводов компонентов. Выводы компонентов соединены, если к различным выводам компонентов подключены одноименные цепи. Символы компонентов должны находиться в соответствующей библиотеке типа SLB в директории X:\FA\P4 или X:\FA\P8. Запуск командного файла на выполнение производится из меню исследовательской САПР или с помощью командной строки: PCCAPS @ <имя файла>.CMD. Командный файл содержит команду ввода файла установки среды PCCAPS, включающей рамку одного из стандартных форматов.

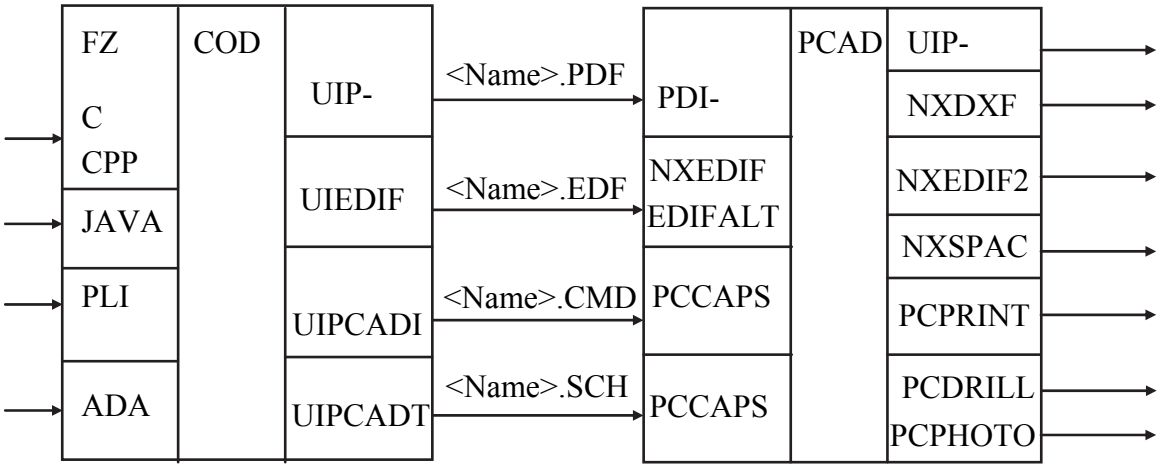


Рис. 4.5. Интерфейсы исследовательской САПР COD и PCAD

Интерфейс УИ САПР с PCAD может выполняться с использованием функций различных уровней. Функции нижних уровней работают со структурами данных конкретной САПР нижнего уровня, например PCAD. Функции верхних уровней не зависят от конкретной САПР нижнего уровня и от способа реализации интерфейса. На рис. 4.6 приведена многоуровневая структура табличного интерфейса с САПР PCAD, а на рис. 4.7 – многоуровневая структура командного интерфейса с различными САПР нижнего уровня. Командный интерфейс УИ САПР с одной САПР нижнего уровня, например PCAD, можно реализовать при использовании функций даже одного уровня. Однако в этом случае модели компонентов будут различными для каждой конкретной САПР. Многоуровневая структура интерфейса позволяет использовать общие модели компонентов, находящиеся в файле comp, и выделить множество функций интерфейса для различных САПР нижнего уровня. Функции поиска путей к ресурсам и имена формализованных заданий находятся в файле codread. Поиск производится сначала в переменных окружения, затем в файле cod.ini.

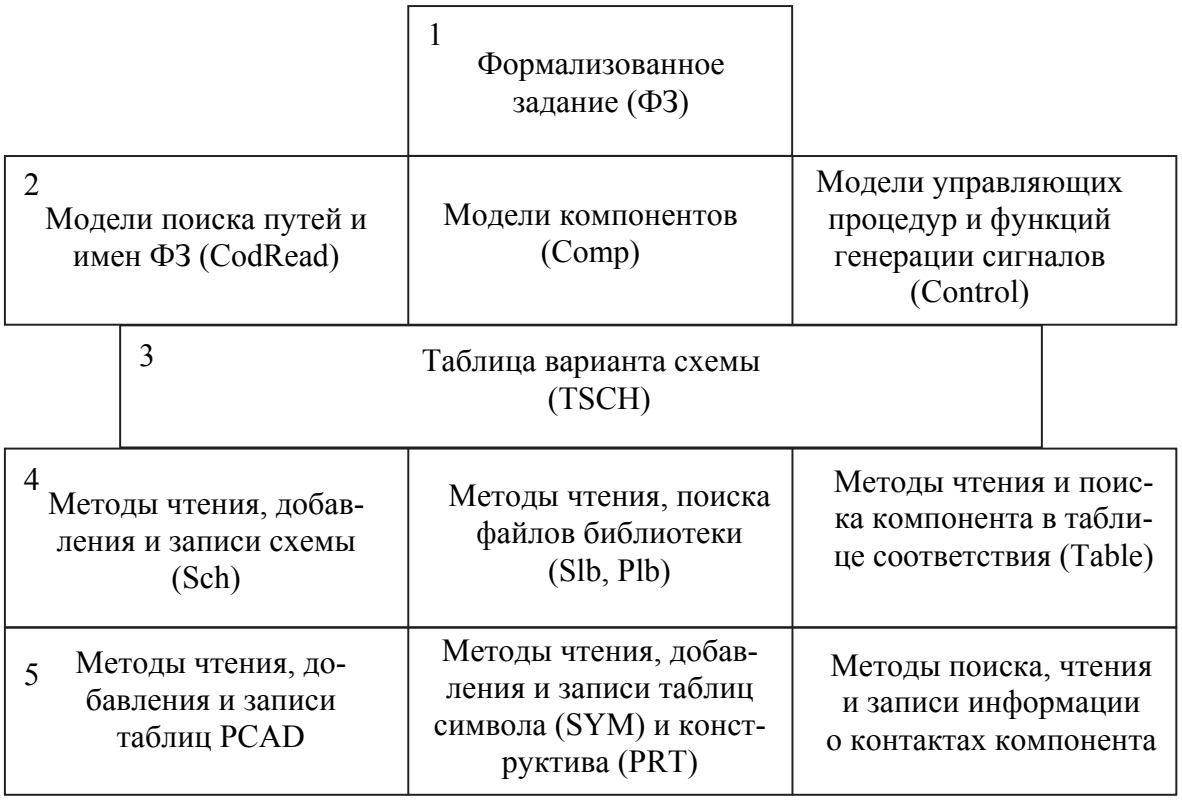


Рис. 4.6. Многоуровневая структура табличного интерфейса УИ САПР с САПР PCAD

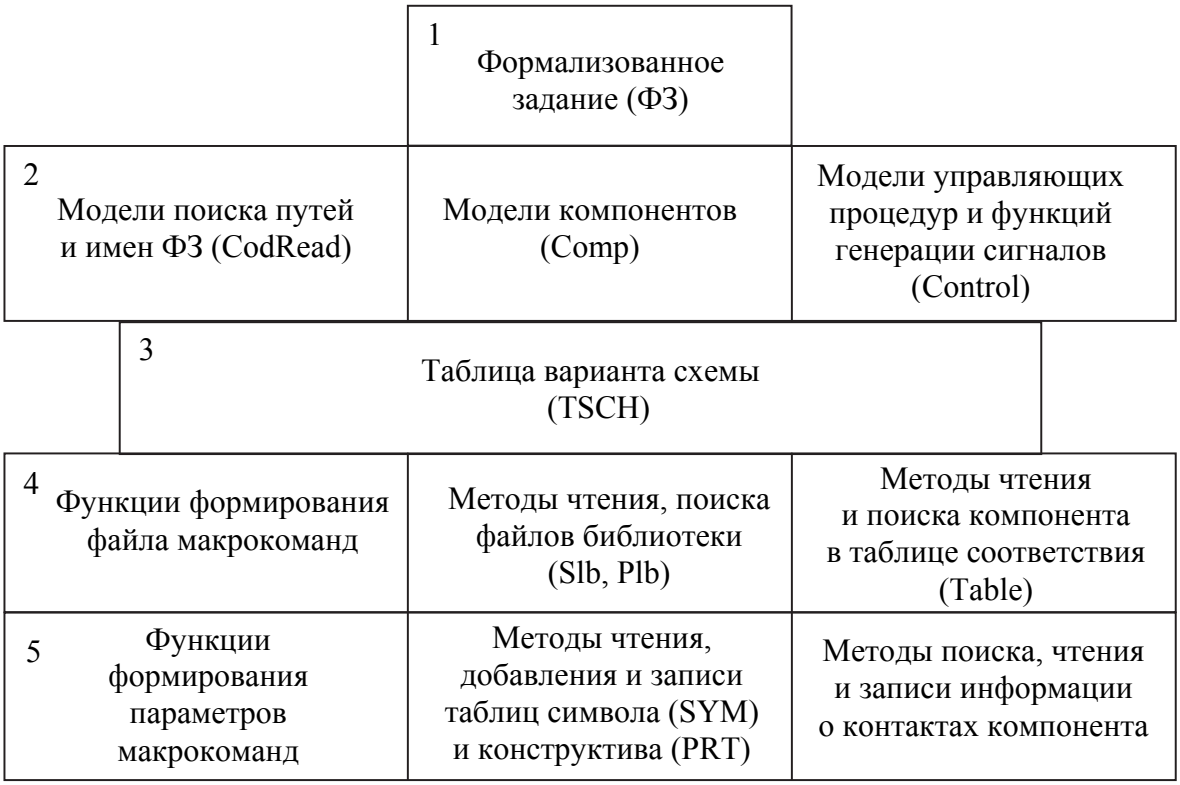


Рис. 4.7. Многоуровневая структура командного интерфейса УИ САПР с САПР PCAD, ORCAD

Такой подход позволяет использовать общее программное обеспечение интерфейса на рабочей станции и на сервере. На рабочей станции информация находится в файле cod.ini, а на сервере регистрация не имеет смысла и информация располагается в переменных среды выполняемого задания. Модели управляющих процедур и функций генерации сигналов больше зависят от САПР нижнего уровня и объединены в файл control. В этот же файл помещаются функции модели компонентов среднего уровня (C, Component), которые обеспечивают независимость моделей компонентов верхнего уровня от вида приложений. Между функциями третьего и четвертого уровней формируется структура данных, полностью описывающая компоненты и соединения между ними. Структура данных называется *таблицей схемы* и обозначается TSCH. Структура данных TSCH подобна таблице соответствия uipcad.dbn, функции работы с которой сведены в файл table. Таблица схемы TSCH, по сравнению с записями uipcad.dbn, дополнена именами цепей, подключенных к соответствующим выводам компонентов. Нижний (пятый) уровень функций образует методы чтения, добавления и записи данных в таблице САПР PCAD, приведенных на рис. 4.6. Методы чтения, добавления и записи таблиц символов (SYM) и конструктивов (PRT) обеспечивают работу с компонентами в среде САПР PCAD. К этому же уровню относятся методы поиска, чтения и записи информации о контактах компонента. На базе методов работы с символами и конструктивами компонентов создаются методы поиска и чтения файлов библиотек символов типа slb и библиотек конструктивов plb, составляющие функции четвертого уровня. К четвертому уровню относятся функции формирования файлов макрокоманд командного интерфейса УИ САПР и САПР нижнего уровня.

Автоматическое формирование описаний устройства в формате базы данных PCAD отличается быстродействием и отсутствием наглядности. Схема алгоритма реализации интерфейса САПР с PCAD приведена на рис. 4.8. Описание устройства в формате базы данных PCAD представляет собой множество отношений в форме таблиц с перекрестными ссылками, схематически приведенных на рис. 4.9. Следует отличать форматы таблиц для версии САПР PCAD до 4.5 включительно и для версий от 6 и выше. Первые являются 16-разрядными версиями САПР PCAD и имеют версию базы данных 1 (1.04 для PCAD 4.5), а вторые – 32-разрядными версиями с базой данных 2 (2.09 для PCAD 8.5). Начиная с версии 8.5 добавлена таблица списка компонентов (complist), содержащая порядковый номер компонента, наименования компонента и дополнительные данные.

Интерфейс исследовательской САПР с промышленными реализуется путем переопределения функций моделей компонентов. Особенностями УИ САПР являются многократное использование единого формализован-

ного задания для различных приложений и многофункциональные модели компонентов с общим интерфейсом. Модели компонентов объединяются в библиотеки в соответствии с приложением и именуются в соответствии с табл. 4.4. Для моделирования на языке C++ в среде MS DOS используется библиотека моделей компонентов LCDF.LIB, для интерактивного интерфейса – библиотека моделей компонентов LCD8I, а для реализации табличного интерфейса – библиотека моделей компонентов LCD8T. В зависимости от версии PCAD к имени библиотеки добавляется цифра в соответствии с табл. 4.4. Например, для PCAD 4.5 библиотека моделей табличного интерфейса имеет имя LCD4T, а для версии 8.5 – LCD8T. Все библиотеки моделей должны находиться в каталоге X:\COD\LIB для среды MS DOS, в каталоге X:\CODOS\LIB для среды OS/2 и в каталоге X:\CODOS\LIBWIN для среды Windows. Если на машине установлено несколько версий системы PCAD, то используемая должна иметь корневой каталог PCAD. В среде OS/2 и Windows оболочка позволяет выбирать текущую версию из PCAD4.5 и PCAD8.5. Библиотеки символов компонентов типа SLB и конструктивов типа PLB должны находиться в подкаталоге P4 или P8 каталога FA, в которых дополнительно размещаются схемные файлы форматов A1–A4 и конструктивы модуля ЭВМ типа BRD.

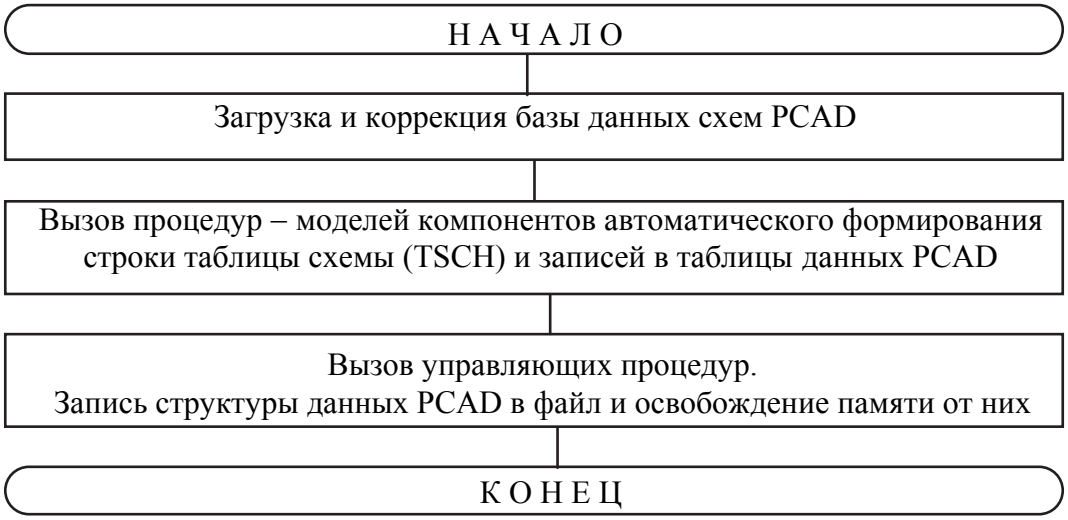


Рис. 4.8. Блок-схема интерфейса САПР с PCAD путем автоматического формирования символов компонентов и связей в формате базы данных PCAD (табличный интерфейс)

Каждому виду интерфейса соответствует своя библиотека моделей компонентов с единым интерфейсом, определенным в файле UICPP.H для языка C++, в файле WALL.INC для языка PL/1 и в файле COMP.JAVA для

языка JAVA. Как и для моделирования, все процедуры моделей компонентов имеют две точки входа, отличающиеся для языка PL/1 использованием массивов входов или выходов – окончание имени процедуры NIN, или указанием минимального номера цепи и их количества – окончание имени процедуры MIN. Основная точка входа использует массив цепей, а в процедуре со второй точкой входа производится заполнение массивов номеров цепей и вызов основной процедуры. Объявление общего имени и выбор конкретной точки входа для языка PL/1 производится оператором GENERIC. Примеры объявлений приведены в файле WALL.INC для сред OS/2 и Windows, а для среды VM – в файле WALL.COPY и макробиблиотеке ADI.MACLIB.

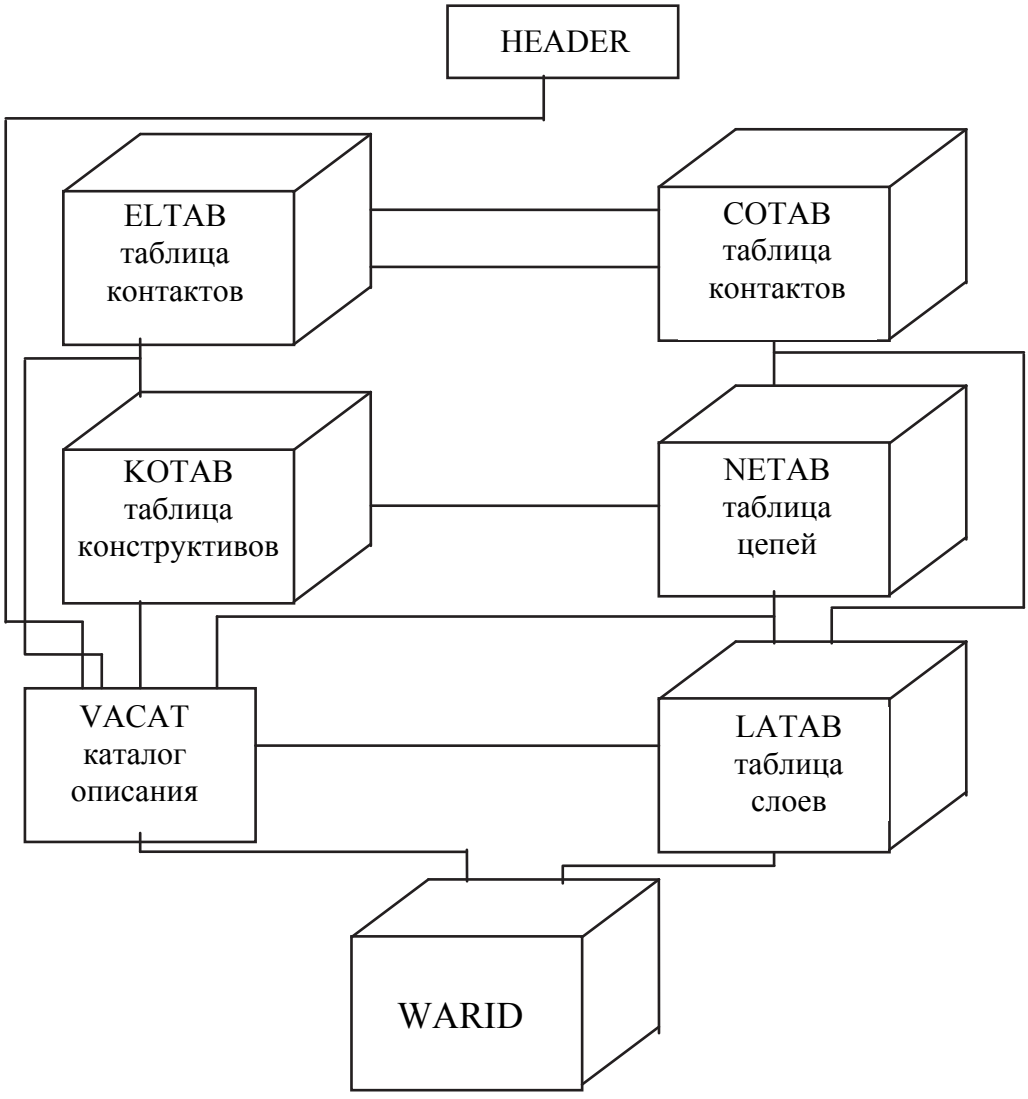


Рис. 4.9. Основные таблицы описания схемы в PCAD

Для ознакомления с автоматизированным интерфейсом САПР PCAD следует выбрать формализованное задание с конкретными типами компо-

нентов, содержащихся в таблицах UIPCAD.DBT, UIPCAD.DBN; провести моделирование и, используя табличный интерфейс, получить принципиальную схему для выбранной версии PCAD; проверить полученный файл типа *.SCH с помощью интерактивного редактора PCCAPS и провести моделирование схемы в соответствии с маршрутом. Результаты моделирования в исследовательской системе сравнить с результатами, полученными в PCAD. Сравнить трудоемкость текстового и графического ввода описания схем.

При использовании одной версии САПР технического проектирования, например PCAD, достаточны следующие подкаталоги в каталоге исследовательской САПР COD:

BIN – исполняемые модули, таблицы UIPCAD.DBT, UIPCAD.DBN, PAC.DBT;

LIB – для среды MS DOS библиотеки функций с битовыми строками lcdbits.lib (bitscpp.lib), функции для работы с массивами целых чисел lcdintv.lib (intvect.lib), функциональных моделей компонентов lcdf.lib (uicpp.lib), интерфейсных моделей компонентов и библиотеки для других приложений в соответствии с табл. 4.4. Для среды OS/2 библиотеки для работы с битовыми строками lcobits.lib, с массивами целых чисел lcointv.lib и дополнительные библиотеки в соответствии с табл. 4.4;

LIBWIN – для среды Windows библиотеки функций для работы с битовыми строками lcwbits.lib, с массивами целых чисел lcwintv.lib и дополнительные библиотеки в соответствии с табл. 4.4;

LIBLIN – для среды LINUX, UNIX библиотеки функций для работы с битовыми строками lclbits, с массивами целых чисел lclintv и дополнительные библиотеки в соответствии с табл. 4.4;

INCLUDE – включаемые тексты txtpar, txtdcl, vextcpp, vextc, заголовочные файлы uicpp.h, uic.h, доступ к таблице table.h и другие заголовочные файлы;

INCPLI – включаемые тексты txtpar.inc, txtdcl.inc, vext.inc, текст перехода на метку раздела формализованного задания w.inc и файл выбора точек входа в процедуры wall.inc;

DOC – документация;

EFF – программы, таблицы и схемы для оценки эффективности и выбора варианта вычислительного устройства или системы. Интерактивные примеры параметрической и структурной оптимизации;

JAVA – архивы классов моделирования и объединенного интерфейса, архивы классов отображения XML-файлов и апплет отображения временных диаграмм. В отдельном каталоге находятся архивы классов для отображения языка виртуальной реальности.

Интерфейс с САПР PCAD может выполняться как на рабочей станции, так и на сервере. На рабочей станции пользователь регистрирует формализованное задание, определяет формат чертежа и выбирает локальное или удаленное его выполнение. Поэтому на рабочей станции вся информация о задании находится в файле COD.INI, в каталоге X:\COD\BIN или X:\CODOS\BIN. На сервере регистрация не имеет смысла, так как задания на выполнение поступают в случайные моменты времени, и источником информации о формализованном задании должно быть полученное сообщение. Полученное сообщение устанавливает переменные среды CODE с именем задания и FRAME с форматом чертежа. Поэтому поиск информации о задании производится сначала в переменных операционной среды и только при отсутствии в файле COD.INI. Формализованное задание передается на сервер, выполняется, и результаты принимаются на рабочую станцию и удаляются на сервере. Отображение результатов в графической форме производится на рабочей станции. Такое разделение функций обеспечивает эффективное решение задач САПР в сети ЭВМ. В качестве операционных систем на сервере могут быть установлены Windows, OS/2, Linux, VM/S390 и zVM.

4.6. Маршруты проектирования модулей ЭВМ в системе PCAD200x

Система PCAD200x [29] работает в среде Windows и используется для проектирования модулей ЭВМ и обучения благодаря наглядным графическим редакторам схемотехнического (SCHEMATIC) и конструкторского (PCB) уровней.

САПР PCAD200x является развитием САПР PCAD [17, 18, 22] в среде Windows. Поэтому файлы модулей ЭВМ в формате PCAD v 8.5 и выше воспринимаются в системе PCAD200x. Файлы в формате PCAD 4.5 воспринимаются после преобразования в формат PDIF утилитой PDIFOUT. Файлы PCAD200x импортируются САПР ALTIUM DESIGNER.

САПР PCAD200x состоит из трех интерактивных редакторов: принципиальных схем (Schematic), конструктивов модулей ЭВМ (PCB) и менеджера библиотек (Library Manager). Открыв папку PCAD2001, можно увидеть три пиктограммы, соответствующие интерактивным редакторам. Можно вызвать любой из них. Нужно только обеспечить соответствие путей к подкаталогам и установок в конфигурационных файлах sch.ini, pcb.ini и comp.ini САПР PCAD2001. Изменение настроек выполняется в меню группой команд Options. При конструкторском проектировании в файле pcb.ini должна быть информация о программе трассировки. Если используется программа [QuickRoute], то оператор следует заменить на RouterExe =

C:\<dirn>\QROUTE.EXE. В описанной секции <dirn> соответствует подкаталогу САПР PCAD200x, а <dirw> – подкаталогу, в который установлен Windows.

Работа в САПР PCAD2001 производится в оконной среде с типичными группами команд, представленными выпадающими меню и пиктограммами «быстрых» кнопок. Соответствие команд и макрокоманд, используемых для автоматизации синтеза схем, приведено в табл. 4.8.

Таблица 4.8

Подмножество команд PCAD2001 – PCAD2006 для интерфейса

Назначение команды	Имя команды	Имя подкоманды	Макрокоманда (v. 15)
Открыть новый файл	File	New	SendKeys“{alt+f}{n}”
Открыть файл		Open	SendKeys“{alt+f}{o}{filename*}{enter}”
Сохранить файл		Save	SendKeys“{alt+f}{s}”
Установка единиц измерения	Options	Configure	SendKeys“{alt+o}{c}{m}{enter}”
Установка библиотеки	Library	Setup	SendKeys“{alt+l}{s}{tab}{a}{libname}{enter}{o}"
Вырезать	Edit	Cut	SendKeys“{alt+e}{t}”
Копировать		Copy	SendKeys“{alt+e}{c}”
Вставить		Paste	SendKeys“{alt+e}{p}”
Отменить		Undo	SendKeys“{alt+e}{u}”
Именовать цепь (разместить порт)	Place	Port	SendKeys“{alt+p}{o}{space}{space}{portname}{enter}”
Ввести символ в PCAD2000		Part	SendKeys“{alt+p}{p}{space}{space}{partname}{enter}” В PCAD2001 {space} удалить
Ввести провод		Wire	SendKeys“{alt+p}{w}
Ввести шину		Bus	SendKeys“{alt+p}{b}{enter}”

Примечание. Вместо libname и filename пишется полное имя, включая путь. Расшифровка специальных символов: {:} – {shift+semicolon}, {\} – {backslash}, {.} – {period}.

Рассмотрим маршрут проектирования модулей ЭВМ с использованием САПР PCAD2001, приведенный на рис. 4.10. Основной маршрут проектирования в системе PCAD2001 использует графическое представление описаний модулей ЭВМ, создаваемых с помощью графического редактора Schematic. Для работы с графическим редактором должны быть подготовлены библиотеки, содержащие все используемые компоненты, которые хранятся в файлах типа LIB.

Синтез схемы происходит в редакторе Schematic.

Возможен один из следующих вариантов получения схемы:

- создание схемы вручную в редакторе Schematic;
- получение схемы, созданной в PCAD 8.5, из файла типа SCH.

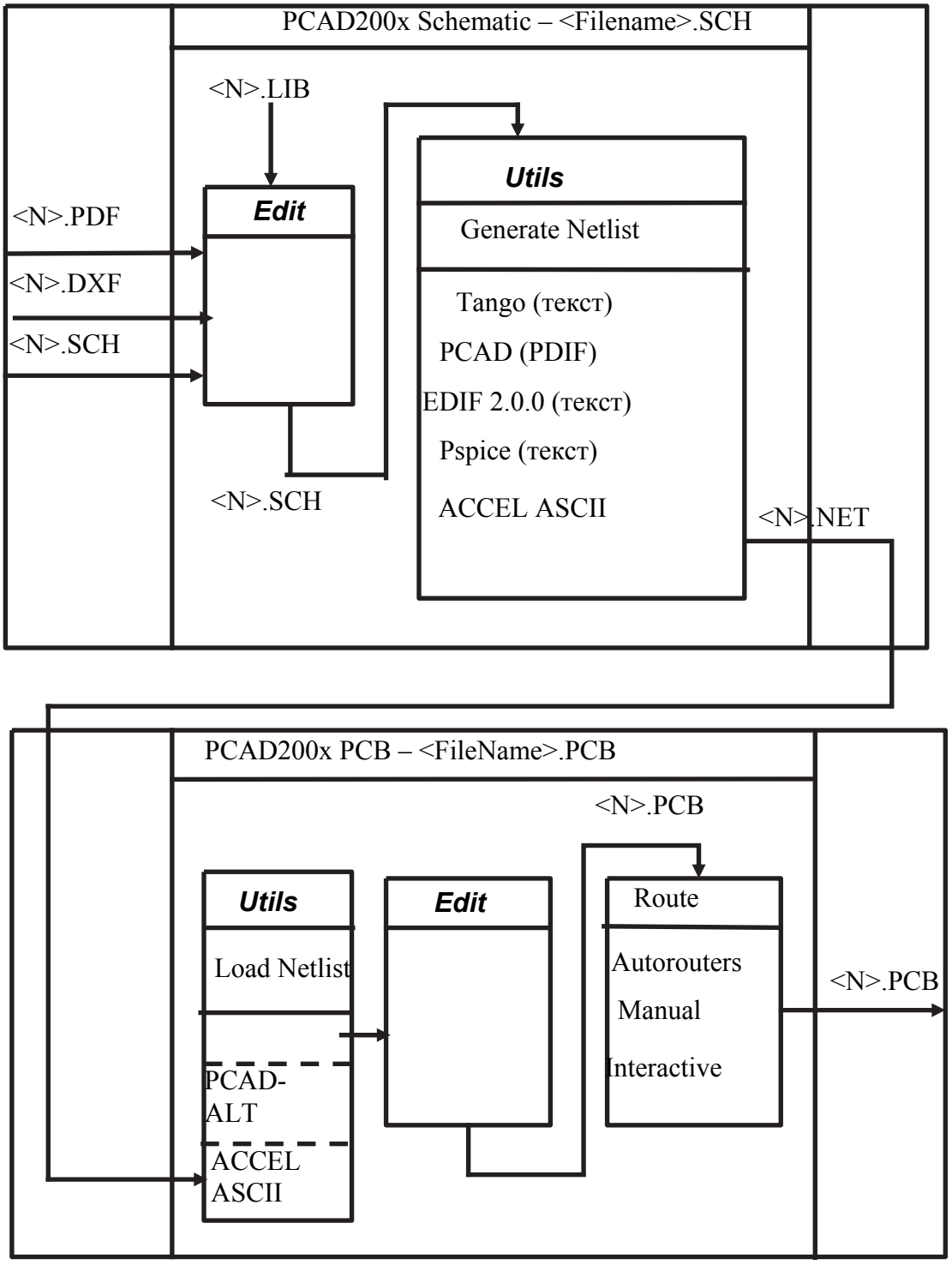


Рис. 4.10. Маршрут проектирования модулей ЭВМ
с использованием САПР PCAD2000, PCAD2001 – PCAD2006

После того как схема синтезирована, необходимо создать файл списка цепей типа NET (команда Utils Generate Netlist). Создав файл списка цепей, можно переходить к этапам конструкторского проектирования и трассировки, которые выполняются в редакторе PCAD200x PCB. Для загрузки схемы нужно выбрать команду Utils Load Netlist, затем разместить компоненты на плате. Трассировка выполняется автоматически (команда Route Avtorouters) с графическим отображением процесса.

Входом программы трассировки является описание модуля с размещенными компонентами и правила (стратегия) трассировки в файле типа STR. Если результаты трассировки неудовлетворительны по числу неразведенных соединений, длине отдельных проводников или другим параметрам, следует произвести повторное размещение компонентов и повторить процесс трассировки до получения удовлетворительных результатов.

4.7. Результаты выполнения формализованных заданий

Примером оптимального варианта узла АЦП является микромодель груботочного интервального АЦП в файле fp06rgt.cpp. При добавлении 15 % аппаратуры производительность АЦП повышается на порядок за счет интервального метода работы. Два дополнительных компаратора определяют положение входного сигнала внутри или вне интервала [17, 18, 22]. Вне интервала du от среднего значения разрешена работа только старшей (грубой) секции счетчика, а при нахождении входного сигнала внутри интервала разрешена работа и точной секции. Таким образом повышается производительность АЦП. В многоканальном режиме через установочные входы в счетчик заносится предполагаемое значение, и время преобразования на канал определяется отклонением от него. На рис. 2.6 приведена схема, а на рис. 4.12, 4.13 – временные диаграммы второго и третьего вариантов груботочного АЦП в файле fp06rgt.cpp.

```
/* Груботочный АЦП FP06RGT.CPP */
#include <uicpp.h>
void main() {
  int lct = 4, ninp  = 14, noutp  = 15, ninp1=17,noutp1=27;
  BitString b01("01");
  /* Объявление цепей выходов и входов */
  IntVector ctout1 (4,6,7,8,9), ctout(4,10,11,12,13),
```

```
ctin (4,0,0,0,0), kin (4,2,3,0,4),
      mloin(2,19,20), mp(2,14,30), m15(2,15,15), m26(2,26,26);
  #include <txtpar>

INIT: /* Начальная установка */
 lres=-1; ns=32; ntmax=40; nas=5;nppd=4;nppa=3;lra=8;
 nvarmax=3; ttime=0; deltn=1000; deltp=9000; //ns
  #include <txtdcl>

INPUT: /* Внешние воздействия */
  float du=0.2*(nvar-1);
  SignalA(2,2*cos(xt/40+1)+2,ntmin,ntmax);
  SignalA(4,as[3].ANex+du,ntmin,ntmax);
  SignalA(5,as[3].ANex-du,ntmin,ntmax);
  Signal(3,b01,ntmin,ntmax);
  Signal(2,b01,ntmin,ntmin+2);

UNIT: /* Описание устройства */
  CA   ("К554СА3" ,1,1,19,22,2,4); // UDAC+du
  CA   ("К554СА3" ,2,1,20,23,5,2); // UDAC-du
  CA   ("К554СА3" ,3,1,4,21,2,3);
  LO   ("К561ЛЕ5" ,5,5,26,mloin); // Режим  грубого АЦП – асинхронный
  LO   ("К561ЛЕ5" ,7,29,28,m26);  // Режим  грубого АЦП – асинхронный
  MTJK ("К561ТВ1" ,4,14,16,0,0,3,28,26);//Синхронный выход режима
  MCTR ("К561ИЕ11",6,lct,ctout1,noutp1,ctin,ninp1,kin);//Грубая секция
  LO   ("К561ЛЕ5" ,10,18,30,m15);  // Разрешение переноса
  LO   ("К561ЛЕ5" ,11,31,17,mp);   // Разрешение переноса 17
  MCTR ("К561ИЕ11",8,lct,ctout,noutp,ctin,ninp,kin);//Точная секция
  DAC  ("К572ПА1" ,9,3,6,8);
  APrint (2,23,ntmin, ntmax,2,5);
  #include <w>
  }
```

Ниже приведен файл fp06rgt.htm для отображения первого варианта принципиальной схемы:

```
<html> <body>
<applet code=CODXML.SCHView.class
 archive=..\..\CODOS\JAVA\CODXML.jar width=1000 height=2000>
<param name=xmlname value="D:\FA\CPP\fp06rgt.xml">
</applet> </body> </html>
```

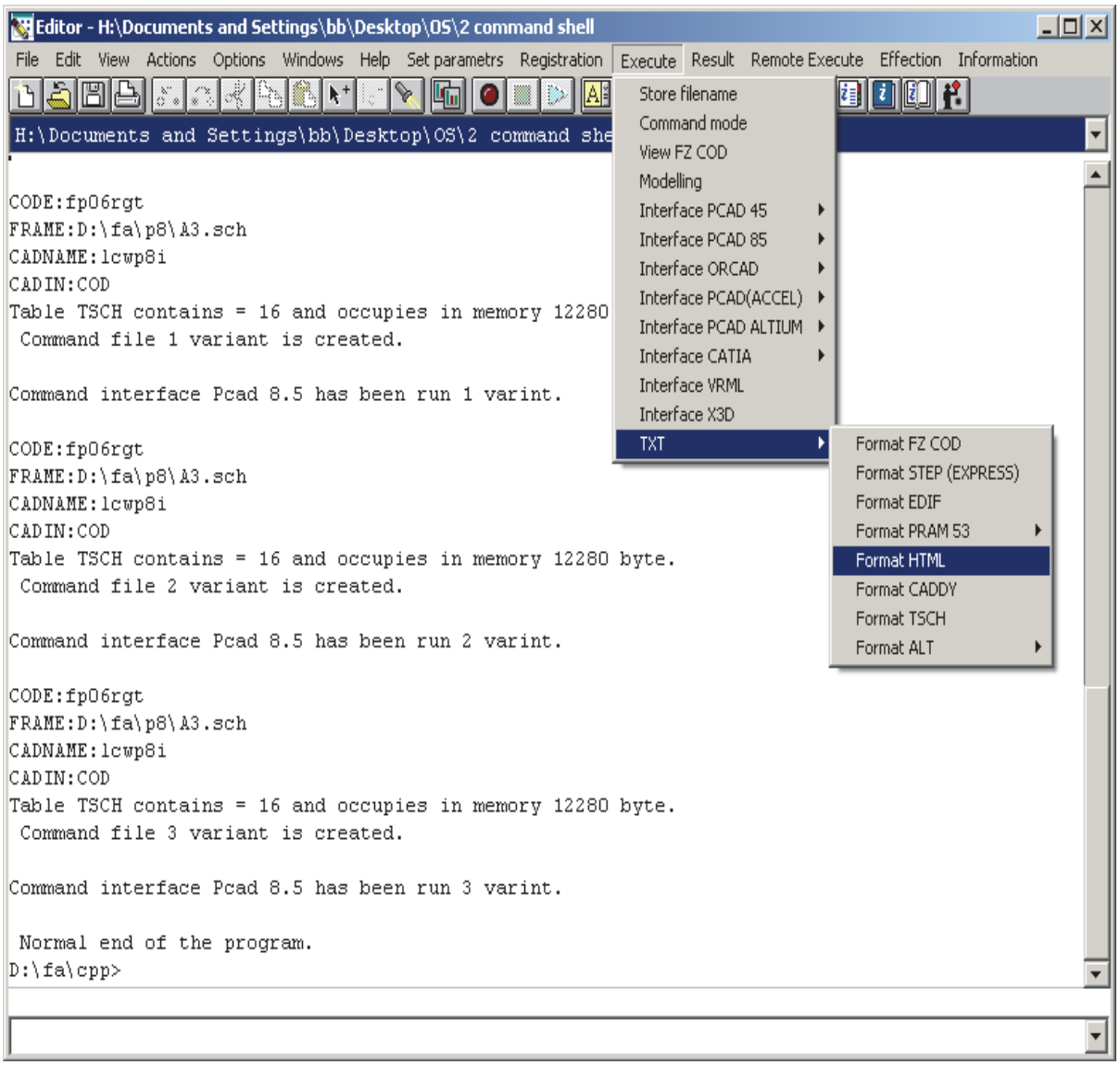


Рис. 4.11. Меню САПР COD СФУ в редакторе LPEX IBM Visual Age

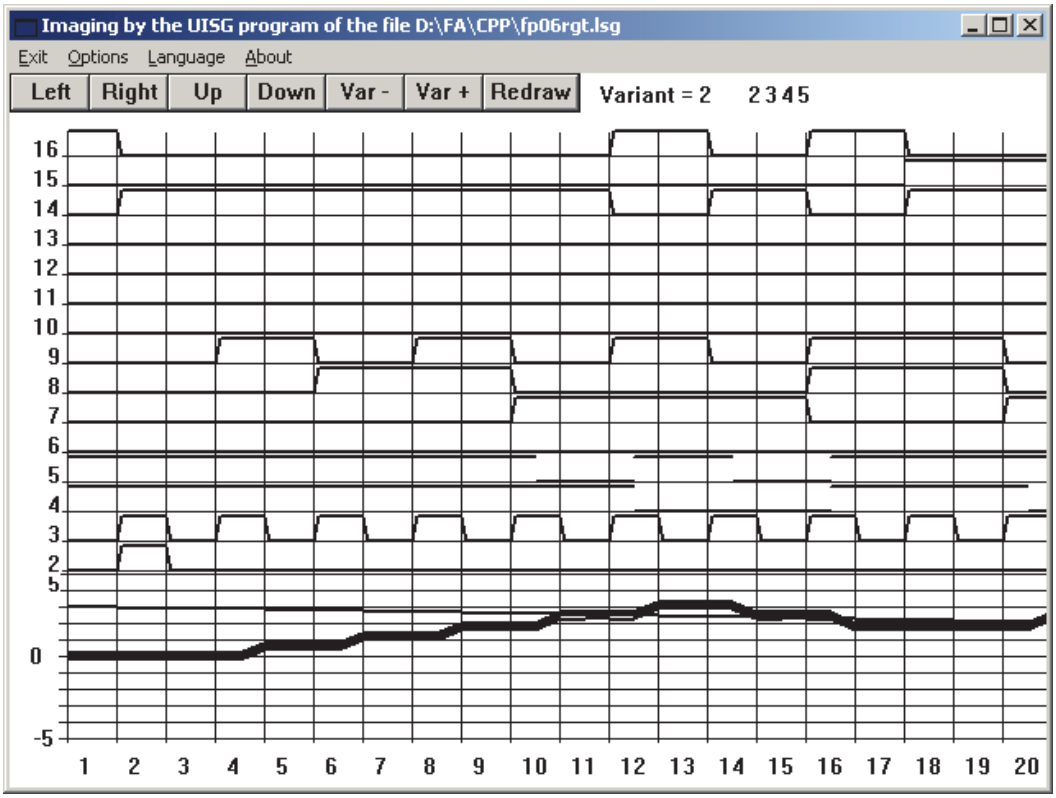


Рис. 4.12. Временная диаграмма работы второго варианта АЦП в файле fp06rgt

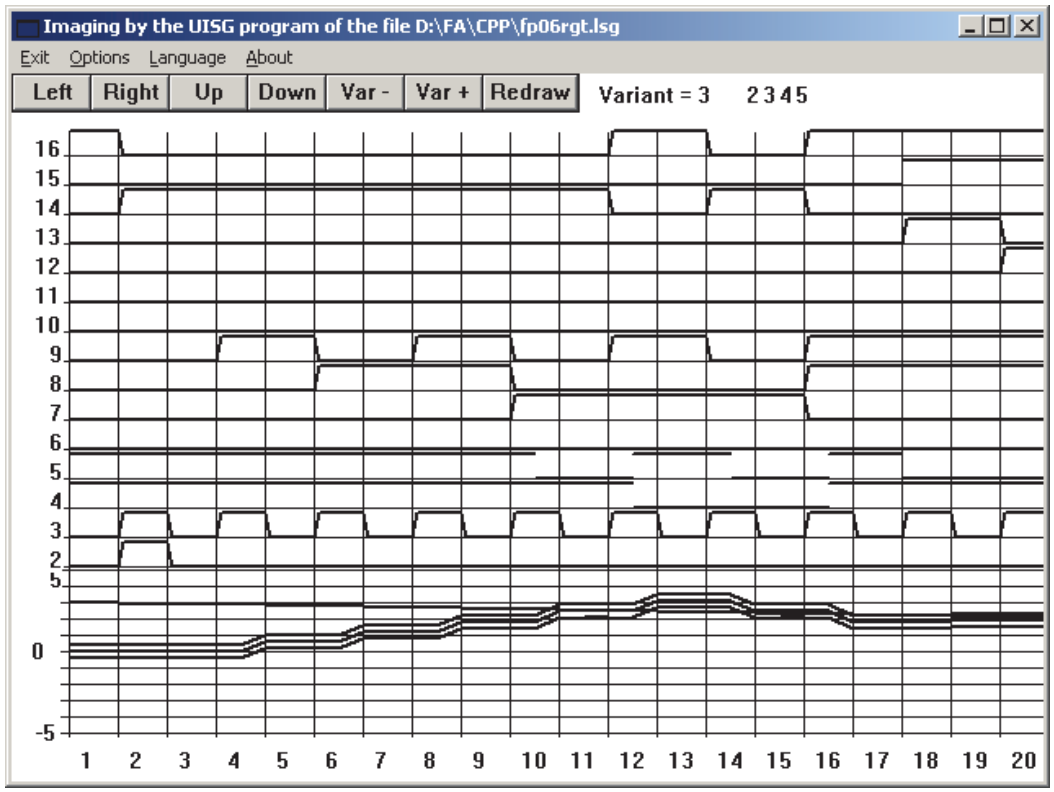


Рис. 4.13. Временная диаграмма работы третьего варианта АЦП в файле fp06rgt

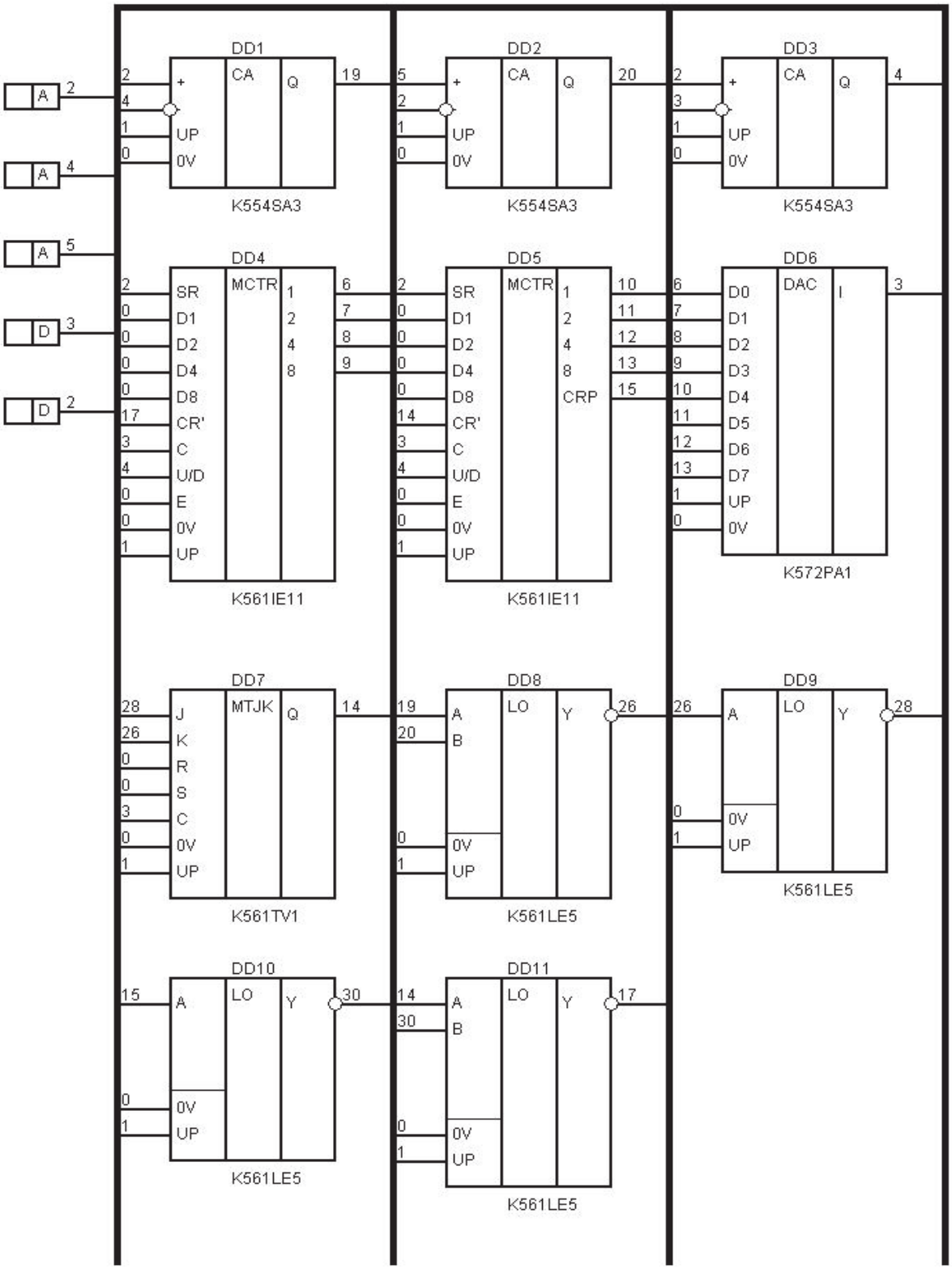


Рис. 4.14. Принципиальная схема груботочного АЦП fp06rgt, автоматически построенная в сетевой программе просмотра (Internet Explorer, Mozilla)

Фрагмент xml файла fp06rgt.xml для создания принципиальной схемы первого варианта груботочного АЦП fp06rgt.cpp представлен далее:

```
<!--DOCTYPE root SYSTEM "D:\CODOS\BIN\PARSER.DTD"-->
<s_schema>
 <s_title>Schema fp06rgt</s_title>
 <s_components>
  <s_component
   ElmNamR="K554CA3"
   ElmNamL="K554SA3"
   ElmNamA="LM111"
   DOCNameR="БК0.348.279ТУ1"
   DOCNameL="BK0.348.279TU1"
   DOCNameA="NONE"
   P0="120.000000"
   P01="0.000100"
   T01="100.000000"
   PNOM="150.000000"
   S="162.000000"
   Massa="1.000000"
   TPmin="-10.000000"
   TPmax="70.000000"
   Rt="41.200001"
   Tau="0.000000"
   PriceRel="4.000000"
   TYPEElm="CA"
   FunName="CA"
   FunCode="49"
   KElmPack="1"
   TYPEPack="201.14-1"
   NumTYPE="0"
   NumPack="1"   >
   <s_outputs KPinOut="2">
    <s_pin
     NElmPack="0"
     PinLeq="0"
     PinN="2"
     PinPCAD="Q"
     PinCod="NOUTP"
     NetNum="19"
     NetName="ND19"
     NetTYPE="ND"      />
   </s_outputs>
   <s_inputs KPinIn="6">
    <s_pin
     NElmPack="0"
     PinLeq="0"
     PinN="3"
     PinPCAD="+"
     PinCod="NINP"
     NetNum="2"
     NetName="NA2"
     NetTYPE="NA"      />
    <s_pin
     NElmPack="0"
     PinLeq="0"
     PinN="4"
     PinPCAD="-"
     PinCod="NINN"
     NetNum="4"
     NetName="NA4"
     NetTYPE="NA"      />
    <s_pin
     NElmPack="1"
     PinLeq="0"
     PinN="10"
     PinPCAD="UP"
     PinCod="UP"
     NetNum="1"
     NetName="ND1"
     NetTYPE="ND"      />
    <s_pin
     NElmPack="1"
     PinLeq="0"
     PinN="14"
     PinPCAD="0V"
     PinCod="GND"
     NetNum="0"
     NetName="ND0"
     NetTYPE="ND"      />
   </s_inputs>
  </s_component>
```

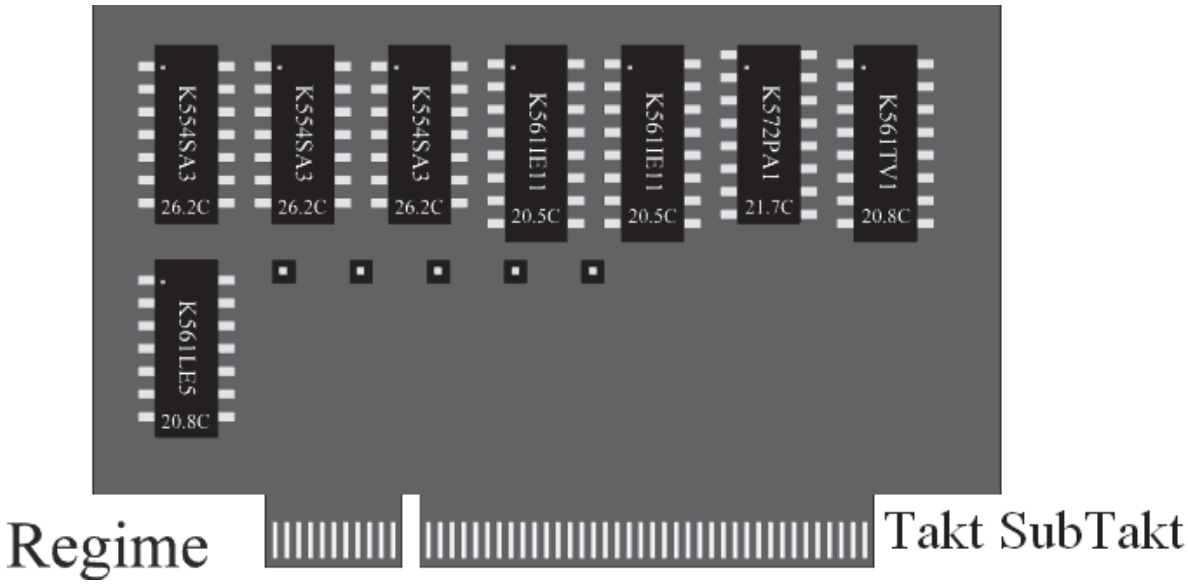


Рис. 4.15. Трехмерная модель модуля АЦП fp06rgt
с отображением температуры компонентов без изменения цифровых сигналов

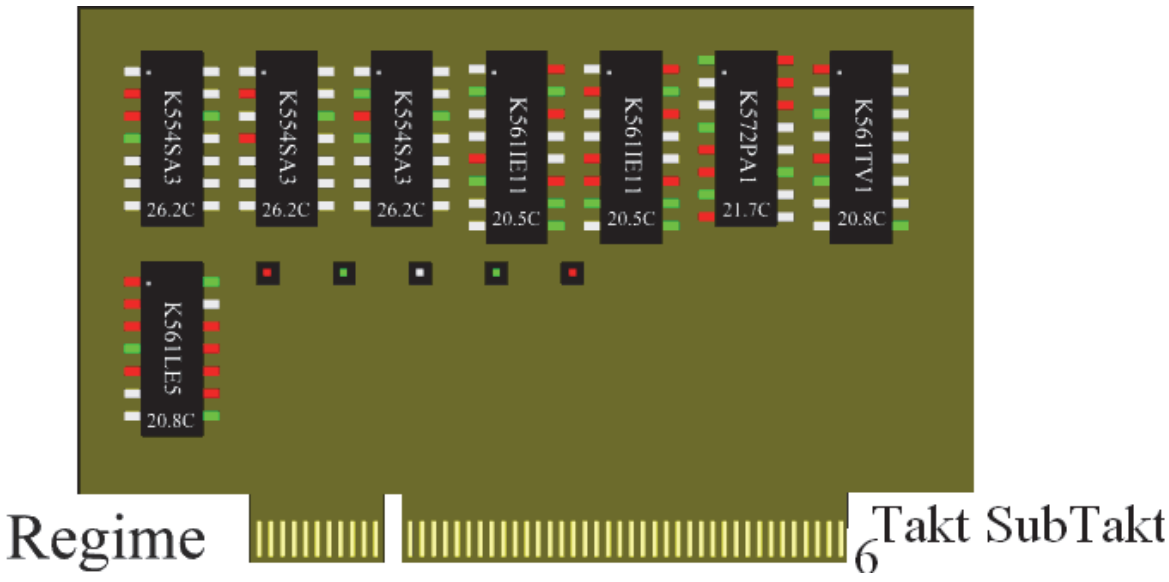


Рис. 4.16. Трехмерная модель модуля АЦП fp06rgt
с отображением температуры компонентов с изменением цифровых сигналов

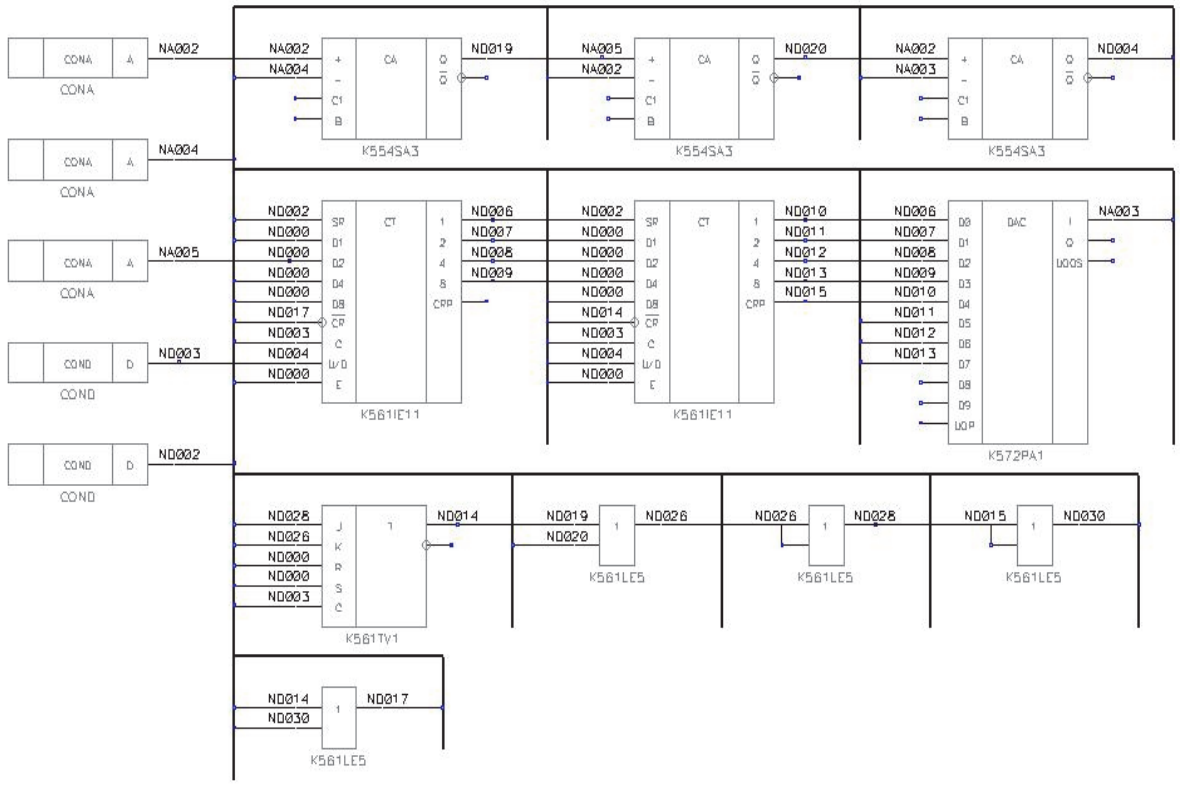


Рис. 4.17. Принципиальная схема узла груботочного АЦП,
полученная с помощью табличного интерфейса в PCAD8.5 из файла fp06rgt

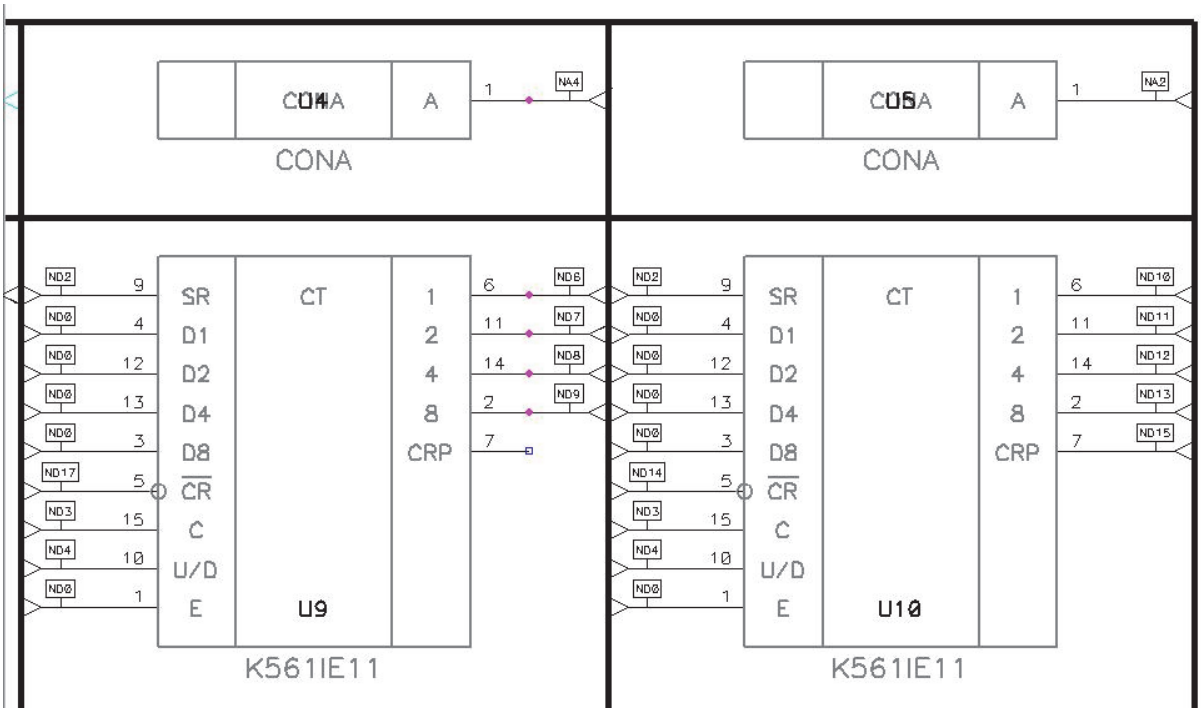


Рис. 4.18. Фрагмент принципиальной схемы узла груботочного АЦП,
полученный с помощью макрофайла (fp06rgt.mac) в PCAD2004

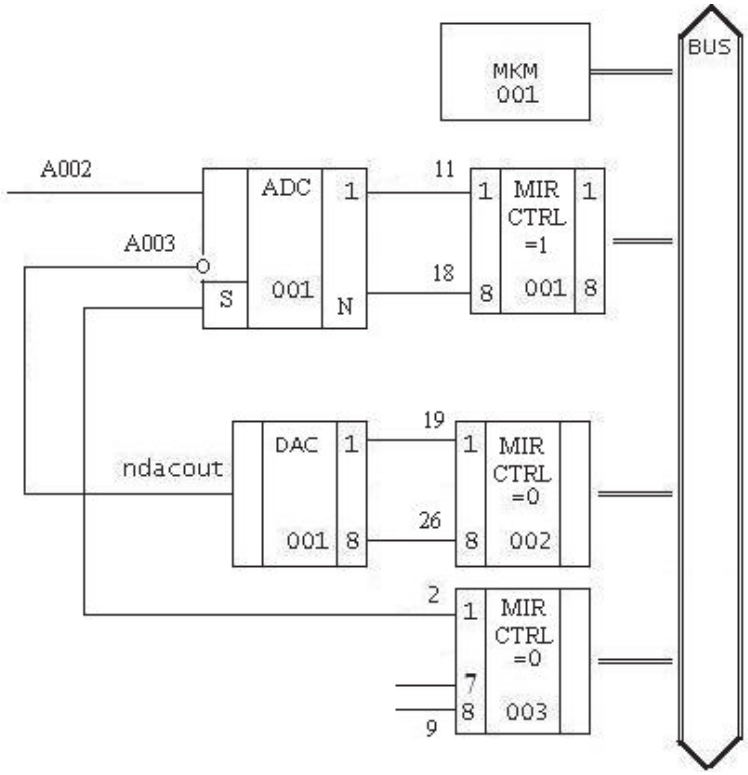


Рис. 4.19. Схема вычислительной системы на базе микроЭВМ
с вводом отклонений от предполагаемого сигнала
с помощью АЦП одного отсчета

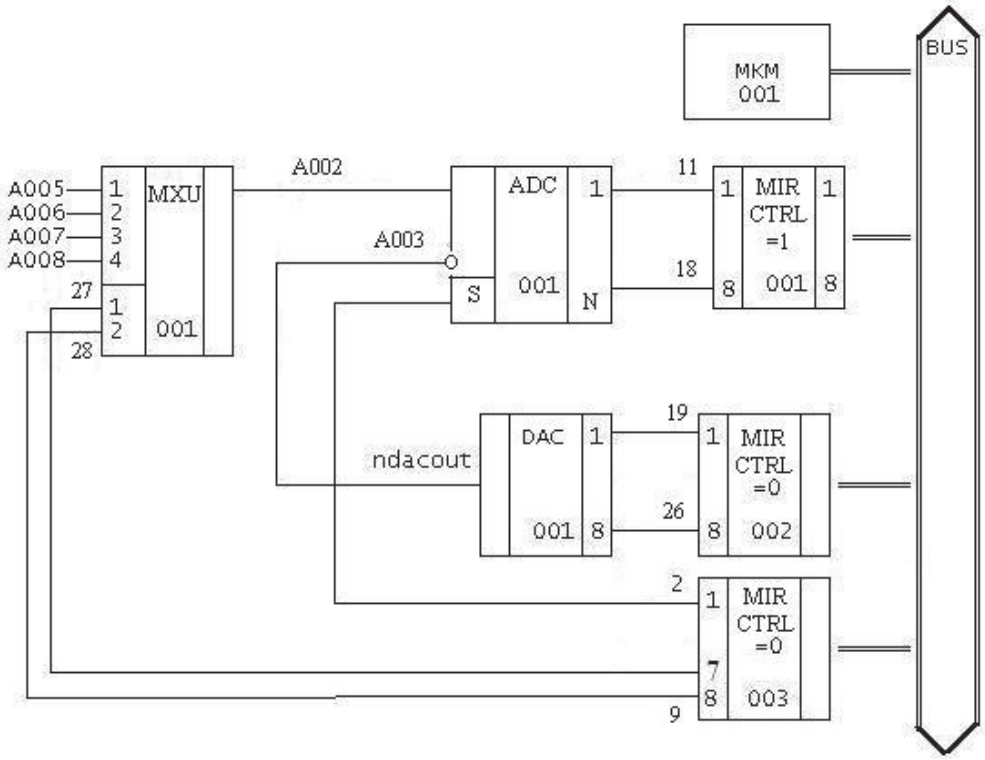


Рис. 4.20. Схема системы с коммутацией каналов на базе микроЭВМ
с вводом отклонений от предполагаемого сигнала
с помощью АЦП одного отсчета

Системы на базе микроЭВМ позволяют отказаться от цифровых компонентов и уменьшить число переключений цифроаналоговых компонентов. На рис. 2.8 приведена схема подсистемы на базе микроЭВМ с интервальным прогнозированием и оценкой отклонений сигнала, а в файле fp08adcp.cpp – соответствующее формализованное задание.

```
#include <uicpp.h>
void main() {   // файл fp08adcp.cpp
  BitString b01("01");
  int ninc = 6, sign=2, dacout=3, noutca=4, noutpca=15,noutmca=16,
  pdu=4, mdu=5;   float du=0.75;
  IntVector noutct(8, 7,8,9,10,11,12,13,14);
  IntVector mirtmp(8, 4,noutpca,noutmca,0,0,0,0,0);
 #include <txtpar>
 INIT:/* Начальная установка */
  nvarmax = 1;  lres = -5;    ns = 40;    ntmax = 1800; nrg = 4;   lrg = 8;
  deltn = 1000; deltp = 9000; //ns
 #include <txtdcl>
 INPUT:/* Внешние воздействия */
  Signal(ninc,b01,ntmin,ntmax);
  SignalA(2,cos(xt/50)+7,ntmin,ntmax);
  SignalA(pdu,as[3].ANex+du,ntmin,ntmax);
  SignalA(mdu,as[3].ANex-du,ntmin,ntmax);
 UNIT:/* Описание устойства */
  MIR ("MIR", 1, 1, ninc, mirtmp);
  MIR ("MIR", 2, 0, ninc, noutct);
  CA  ("К554СА3", 3, 1, noutca, 5, sign, dacout);
  CA  ("К554СА3", 4, 1, noutpca, 5, sign, pdu);
  CA  ("К554СА3", 5, 1, noutmca, 5, mdu, sign);
  DAC ("К572ПА1", 6, dacout, noutct);
  MKM ("М686", 7);
  APrint(2, 20, ntmin, ntmax, 2, 5);
 #include <w>
}
```

Диаграммы работы аналогичны приведенным на рис. 4.12–4.13.

Эквивалентная схема с использованием АЦП одного отсчета с малым числом разрядов для оценки отклонений от предполагаемого сигнала для одного канала приведена на рис. 4.19 и для многих аналоговых сигналов – на рис. 4.20. Основное внимание нужно уделить системам с прогнозом интервала сигналов или с моделью объекта. С накоплением знаний о сигна-

лах и объектах в процессе жизненного цикла систем, уменьшается поток входной информации и увеличивается поток прогнозных оценок сигналов.

* * *

Предложен модульный многоуровневый программно-методический комплекс многовариантного моделирования и проектирования неоднородных вычислительных систем. Новым является единое представление варианта объекта, позволяющее существенно снизить количество модулей и объем программного кода. Сложная обработка формализованных заданий требует значительных затрат энергии, поэтому рекомендуется локализация обработки с визуализацией на рабочем месте.

Созданы модели и алгоритмы реализации модульных интерфейсов с системами схемотехнического и конструкторского проектирования. Единое представление варианта объекта преобразуется модульным комплексом в представления, воспринимаемые множеством разнообразных специализированных и комплексных систем схемотехнического и конструкторского проектирования. Созданы табличный, командный и текстовой варианты моделей интерфейсов. Наибольшим быстродействием отличается табличный интерфейс, наглядностью – командный. Особо следует выделить автоматическое создание формализованных заданий в различных синтаксических средах.

5. АВТОМАТИЗАЦИЯ ПРОЕКТИРОВАНИЯ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ ОБЪЕКТОВ И ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

5.1. Информационные модели объектов в САПР

Основой предлагаемого подхода является однократный ввод описаний объектов в виде формализованных заданий для различных уровней анализа и конструкторско-технологического проектирования. Привычные инженеру графические документы в виде временных диаграмм, схем и сборочных чертежей должны получаться автоматически в результате интерпретации формализованных заданий и решений. При этом повышается производительность труда студентов и инженеров и становится реальным активное обучение в проектной команде. Основное внимание специалистов уделяется принятию решения при автоматизированной оценке вариантов. Символические модели систем, содержащие существенную информацию о функциях или структуре, занимают важное место в процессах обучения и проектирования. Информационная модель описана в ГОСТ 34.003–90 и затем многократно уточнялась. Интерпретация модели или ее изменений во времени зависит от предметной области и облегчает восприятие [1–3, 6].

Символические модели систем могут существовать в среде объекта, обеспечивающего систему комфортными условиями, энергией и сетью передачи информации. Объект может представляться комплексной (междисциплинарной) информационной моделью или множеством вложенных комплексных информационных моделей.

На первом этапе выбирается система измерения и определяются уровни этажей, создаются все необходимые компоненты, отсутствующие в библиотеках. На втором этапе проектирования разрабатывается модель объекта. Существует три степени проработки модели.

Построение генплана. На этом этапе модель представлена в качестве системы коробок, но не имеет в своем составе детальной проработки каждого отдельно взятого объекта.

Каждый отдельный объект системы прорабатывается более детально, создается планировка комнат, этажей, лестниц, задается расположение дверей и окон.

Моделирование помещений. На этом этапе создаются несколько видов помещений, имеющих одинаковую планировку в здании и часто повторяющихся на плане.

Объект может создаваться в среде виртуальной реальности или X3D без ограничений на тип объекта. Вычислительная сеть может находиться в здании или на открытом пространстве. Для неподвижных объектов, например зданий, подходит среда Autodesk REVIT с возможностью автоматического размещения в помещениях сетевых рабочих мест в соответствии с нормами. REVIT допускает создание пользователем дополнительных инструментов, которые должны быть описаны в файле revit.ini в секции ExternalCommands. Файл revit.ini загружается однократно при вызове REVIT. Дополнительные команды размещаются в меню «инструменты» (Tools). Например, дополнительная команда **Space Naming Utility** присваивает имена и номера помещений в MEP-пространстве.

Все компоненты в REVIT находятся в библиотеках в виде семейств (Famili) класса параметризованных объектов. Конкретные объекты обладают значениями параметров, их файлы имеют расширение rfa. Обычно библиотеки компонент находятся в свободном доступе. Примерами семейств являются двери, окна, рабочие места с компьютерами и креслом. Рекомендуемые компоненты для проекта лучше представить в форме таблицы с внешним видом, наименованием на русском и английском языках и размерами в сантиметрах. Аналого-цифровая подсистема с датчиками (АЦВС) размещается ближе к источникам сигналов, а передача и обработка производится только в цифровой форме.

Например, оптимальное размещение точек доступа в здании можно произвести только после оценки поглощающей способности всех конструкций. Проект в REVIT имеет расширение rvt.

Множество стен MWall состоит из объектов различного назначения: Exterior, Interior, Retaining, Foundation и Curtain wall. Внешние стены (Exterior Walls) являются несущими и должны обладать тепловым сопротивлением в соответствии со сводом правил и нормами. Внутренние стены (Interior Walls) используются как внутреннее разделение объема объекта и имеют ненесущий характер. Подпорные стены (Retaining Walls) должны удерживать землю и формировать рельеф. Фундаментные стены (Foundation Walls) формируют фундамент здания. Навесные стены (Curtain Walls) состоят из панелей (panels). В кампусе СФУ2 большинство зданий имеют несущие стены и только корпус Б радиотехнического факультета построен с использованием промышленных панелей. Проектирование здания начинается с создания сетки (Grid) и определения уровней (Level). Сетка осей распределяется в плане (Floor plan), а уровни – с фасада (Elevation).

Кампус университета состоит из множества зданий, дорог и подземных коммуникаций и является объектом более высокого уровня со средой Autodesk INFRAWORKS. Он является частью объекта «город». Процесс проектирования моделей начинается с нижнего уровня и заканчивается верхним, но является итерационным до удовлетворения требований задания. Возможно проектирование множества вариантов объекта для выбора оптимального.

Международный стандарт ISO 15926 установил трехуровневую архитектуру моделей. Внешний (External) уровень моделей соответствует запросам пользователей, внутренние (Internal) модели соответствуют таблице варианта УИ САПР COD. Концептуальный уровень в программно-методическом комплексе представлен формализованным заданием на проектирование (ФЗ-FZ). Основные понятия стандарта ГОСТ Р ISO 15926 приведены в табл. 5.1.

Таблица 5.1

Основные понятия ISO 15926

Уникальное название	Описание	Примечание	Superclass	I ISO 15926-2 entity
Auxiliary system (вспомогательная система)	Система, являющаяся частью большой системы и выполняющая функции, необходимые для работы более крупной системы, но которые не являются частью функций большой системы		System	class_of_inanimate_physical_object
Component (компонент)	Физический объект, который используется как часть большого объекта		physical object class_of_inanimate_physical_object	
Connection (связь)	Физический объект, который предназначен для соединения		physical object	class_of_inanimate__physical_object
Connector (соединитель)	Соединение, состоящее из двух частей	Заглушка и гнездо	connection class_of_inanimate_physical_object	Connector (соединитель)

Окончание табл. 5.1

Уникальное название	Описание	Примечание	Superclass	I ISO 15926-2 entity
End (конец)	Функция, которая является окончанием события		feature	class_of_feature
Plant (объект)	Физические объекты: земля, здания, машины, инструменты, используемые в торговле и промышленности		physical object class_of_ inanimate_ physical_ object	
Plug (пробка)	Физический объект, позволяющий герметично закрыть отверстие		physical object	class_of_ inanimate_ physical_object
Section (часть объекта)	Физический объект, который является одной из основных частей большого объекта		physical object	class_of_ inanimate_ physical_object
Start point (начальная точка)	Функция, которая обеспечивает начало события		feature	class_of_feature
Support (опора)	Физический объект, который в состоянии поддерживать некоторые другие вещи от падения, погружения или дефекта		physical object	class_of_ inanimate_ physical_object
System (система)	Функциональный объект, являющийся набором функциональных объектов, образующих схему для предоставления услуг или служащих для выполнения одной задачи	Примеры: система отопления, система шоссе, системы обработки данных.	functional physical object	class_of_ inanimate_ physical_object

В соответствии со стандартом был проведен анализ процесса проектирования различных объектов в разных областях. Основные результаты представлены в табл. 5.2. При проектировании устройств вычислительных систем моделируются варианты и выбирается оптимальный по критерию

эффективности [1–4]. Для наземных систем это стоимость единицы производительности. При проектировании конструкции печатной платы компоненты размещаются по критерию минимума суммы длин проводников, а при трассировке соединений учитывается штраф за переходные отверстия и возможный уровень помех. Моделируется реальная схема устройства при определенных внешних воздействиях и проверяется соответствие заданию.

Таблица 5.2

Соответствие операций при проектировании объектов различных уровней

Операция	Печатные платы	Здания	Инфраструктура	Организация персонала для совместной работы
Концептуальное моделирование	Концептуальное моделирование и критерии эффективности	Концептуальное моделирование и критерии эффективности	Концептуальное моделирование и критерии эффективности	Концептуальное моделирование и критерии эффективности
Ограничение площади	Контур платы	План участка	План участка (кампуса)	План размещения рабочих мест
Установка слоев	Определение слоев	Назначение этажей	Коммуникационные слои	Уровни обмена сообщениями
Размещение объектов	Установка компонентов	Установка оборудования	Установка зданий и объектов	Размещение персонала
Трассировка соединений (связей)	Трассировка соединений	Системная трассировка	Трассировка надземных и подземных коммуникаций	Правила обмена сообщениями
Моделирование работы системы с оценкой критериев эффективности	Моделирование реальной схемы с оценкой критериев эффективности	Моделирование работы системы с оценкой критериев эффективности	Моделирование работы комплекса (кампуса) с оценкой критериев эффективности	Моделирование работы коллектива в различных условиях с оценкой критериев эффективности

Проектирование здания начинается с анализа участка и возможных внешних воздействий. В соответствии с основной функцией и критериями эффективности выбираются площадь и этажность здания. Проектируется конструкция здания с учетом энергоэффективности. Размещается основное оборудование здания в соответствии с нормами. Затем производится трассировка соединений для всей системы и ее моделирование.

Информационная модель объекта (BIM-Building Information Model) может быть построена в среде Autodesk REVIT с помощью команд при отсутствии или наличии планов этажей. Лучшие результаты получаются при автоматическом построении объектов с помощью программных модулей. Схема алгоритма автоматического построения объектов, при наличии поэтажных планов в форме dwg-файлов, приведена на рис. 5.1. Для программных модулей в САПР используется NetScriptCAD [3–6]. Размещенное оборудование разделяется по назначению на электроснабжение, коммуникационное, технологическое, жизнеобеспечения и управления зданием [4–6]. Программные модули позволяют разместить оборудование в соответствии с нормами, места в лекционных аудиториях, рабочие места в лабораториях и офисах. Трудоемким является процесс соединения оборудования.

Каталог проекта должен содержать подкаталог dwg для поэтажных планов, каталог rvt – для файлов Autodesk REVIT для возможности относительной адресации. Программные модули лучше создавать для многократного использования путем передачи параметров. Примером является модуль загрузки планов этажей.

САПР Autodesk REVIT предполагает работу трех различных специалистов: архитекторов, конструкторов-строителей и специалистов по инженерным системам – системотехников (SYSTEM). Специалисты по инженерным системам должны владеть базовыми знаниями в различных областях и основами программирования для эффективного использования комплексных САПР, подобных REVIT, и создания информационной модели здания (BIM) в процессе его жизненного цикла. Состояние оборудования, помещений и конструкций в соответствии с единым протоколом управления объектом, например BACNET, поступает диспетчеру кампуса для оперативного управления и прогноза поведения объектов. Информационные сети должны быть построены на различных принципах для сохранения работоспособности при чрезвычайных ситуациях. Традиционной является отраслевая подготовка специалистов в узких предметных областях. Нужна корректировка учебных планов для подготовки специалистов по инженерным системам с учетом информационной поддержки комплексными САПР и соответствующими сетевыми сервисами для стационарных и мобильных вычислительных устройств [4–6]. Табл. 5.1 и 5.2 подтверждают общий подход для систем различного уровня.

Интерпретация моделей систем и результатов моделирования выполняется специалистом предметной области непосредственно или после программной обработки. Обработка результатов позволяет снизить время восприятия данных специалистами в разных предметных областях.

5.2. Проектирование информационных моделей объектов на различных уровнях иерархии

Основные понятия в области информационных технологий и автоматизированных систем приведены в ГОСТ 34.003–90. Понятие функциональных компонентов и соответствующие им символы на принципиальных схемах определены в стандарте ЕСКД ГОСТ 2.743–91, а понятие «модель изделия» – в ГОСТ 2.052–2006. Отношения между функциональными компонентами (TREL) составляют вычислительный модуль, который может рассматриваться как часть формализованного задания (FT, FZ), раздел описания модуля (UNIT) и представляться (FVIEW) принципиальной схемой или объемной информационной моделью (BIM).

Использование онтологий при проектировании и сопровождении объектов рекомендуется международным ISO 15926 и российским стандартом Р ИСО 15926, принятым в 2008 г. В связи с большим числом конкретных компонентов и терминов используется многоуровневая модель с небольшим числом базовых понятий на верхнем уровне. По мере уточнения условий работы объекта конкретизируются компоненты и увеличивается их разнообразие. Многоуровневое представление позволяет выровнять сложность принятия решений на различных уровнях.

Среда объекта может создаваться в виртуальной реальности, X3D или в среде конкретной САПР [1–4] без ограничений на тип объекта. Вычислительная сеть может находиться в здании или на открытом пространстве. Для неподвижных объектов, например зданий, подходит среда Autodesk REVIT [3–6] с возможностью автоматического размещения в помещениях сетевых рабочих мест в соответствии с нормами. Кампус университета состоит из множества зданий, дорог и подземных коммуникаций и является объектом более высокого уровня со средой Autodesk INFRAWORKS. Кампус университета является частью объекта «город» [3–6]. Для мобильных объектов используется САПР CATIA [1–4]. Проектирование систем на кристалле может производиться в среде свободной FSF (Free Software Foundation) САПР Electric v.9, написанной на JAVA, или в промышленных системах. Можно использовать биполярные или полевые приборы. Предусмотрено проектирование и печатных плат. Процесс проектирования моделей начинается с нижнего уровня и заканчивается верхним, но является итерационным до удовлетворения требований задания. Минимальный объем описания объекта представляется в символьной форме, а максимальный – в графической. Поэтому минимальной трудоемкостью отличаются ввод символьных описаний и автоматическое преобразование описаний в формы, удобные для восприятия человеком.

Для логического уровня характерны абстракции пространственного размещения компонентов и их физической реализации. Однако при согласовании вычислительной системы (ВС) с внешними каналами необходима конкретизация физической реализации компонентов. На физическом уровне оценивается энергия изменения состояния для двоичного компонента – энергия переключения, которая определяет рассеиваемую объектом мощность для конкретной частоты и ограничивает степень интеграции компонентов [1–4]. На пространственном уровне размещения элементов стремятся к максимальной объемной или поверхностной плотности компонентов, ограниченной технологическими нормами и допустимой рассеиваемой мощностью. Пространственное размещение учитывается только при конструкторском проектировании объектов со стационарными компонентами [5]. Уровень неоднородной физической среды обеспечивает устойчивую обработку информации при возможных внешних воздействиях.

На каждом уровне абстракции рассматривают функциональные модели систем в целом, которые называются макромоделями и описываются функциями выходов и переходов [3]. На следующем уровне абстракции находятся структурные модели, в которых отражается внутренняя структура обьектов и компонент.

Такие модели будем называть микромоделями. Результаты использования макромоделей всегда повторяются. При применении микромоделей можно моделировать нормальную и неисправную работу системы. Анализ несоответствия предполагаемых и фактических результатов позволяет принять решение о модификации правил синтеза класса объектов. Правила синтеза представляются таблицей решений и могут описываться в формализованном задании с помощью условных операторов или условий выбора. Для структурной оптимизации необходимы критерии эффективности [1–4]. Для стационарных вычислительных систем критерием эффективности является стоимость единицы производительности. Для мобильных объектов таким критерием может быть масса единицы производительности или масса вычислительной системы и источника энергии на единицу производительности [3–5].

5.3. Создание объекта в САПР REVIT

DWG (английское *drawing* – чертеж) – бинарный формат файла, используемый для хранения двухмерных (2D) и трёхмерных (3D) проектных данных и метаданных. Является основным форматом для некоторых САПР: AutoCAD, nanoCAD, IntelliCAD. Формат DWG поддерживается функциями импорт-экспорт. Форматы *.dws* (*drawing standards* – стандарты

чертежа), *.dwt* (*drawing template* – шаблон чертежа) также являются форматом DWG. В проекте было использовано пять dwg-файлов SFU2H8L1-5. На рис. 5.2 представлен dwg-файл второго этажа SFU2H8L2.

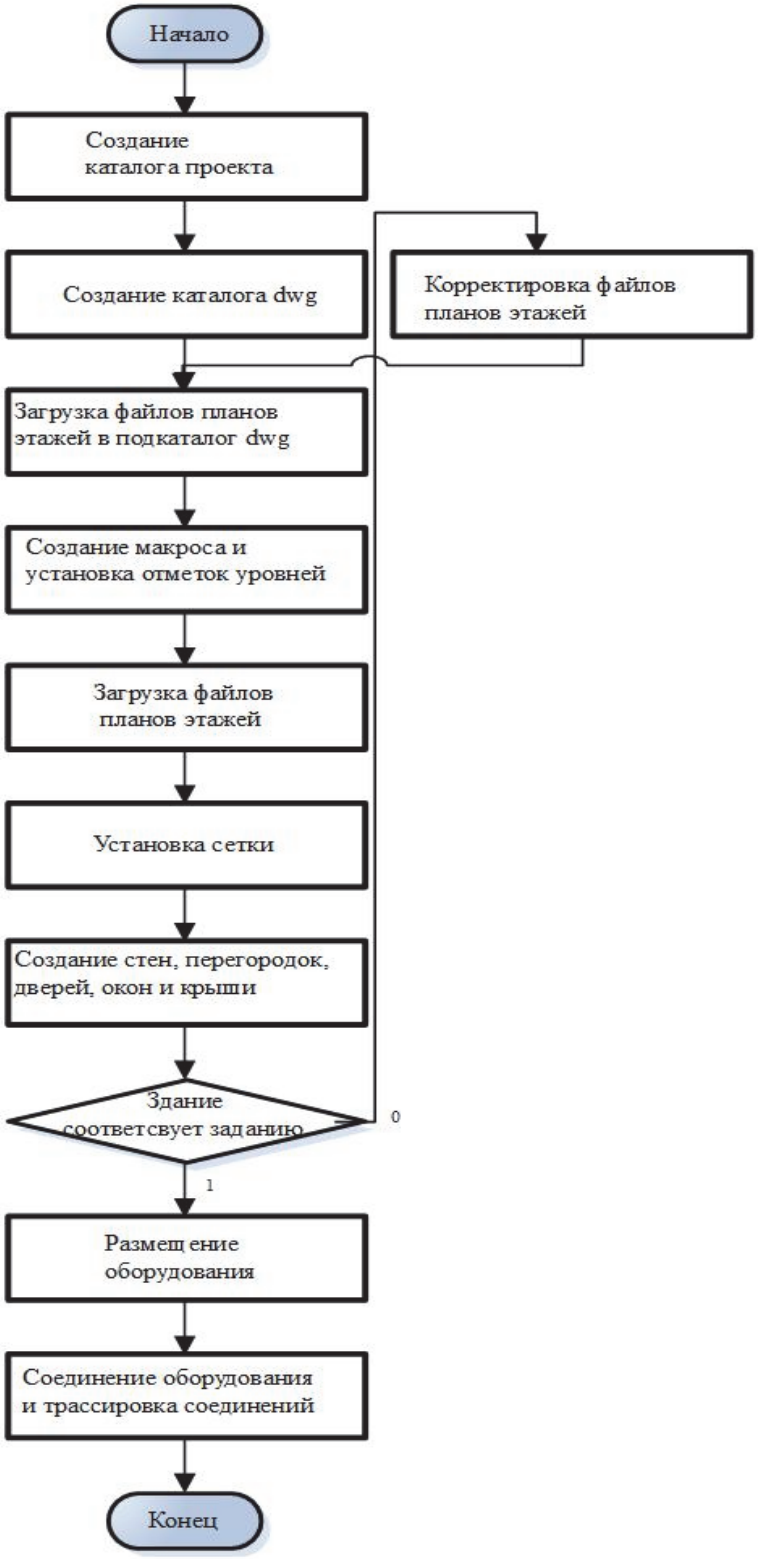


Рис. 5.1. Схема алгоритма автоматического построения объектов

2-й этаж

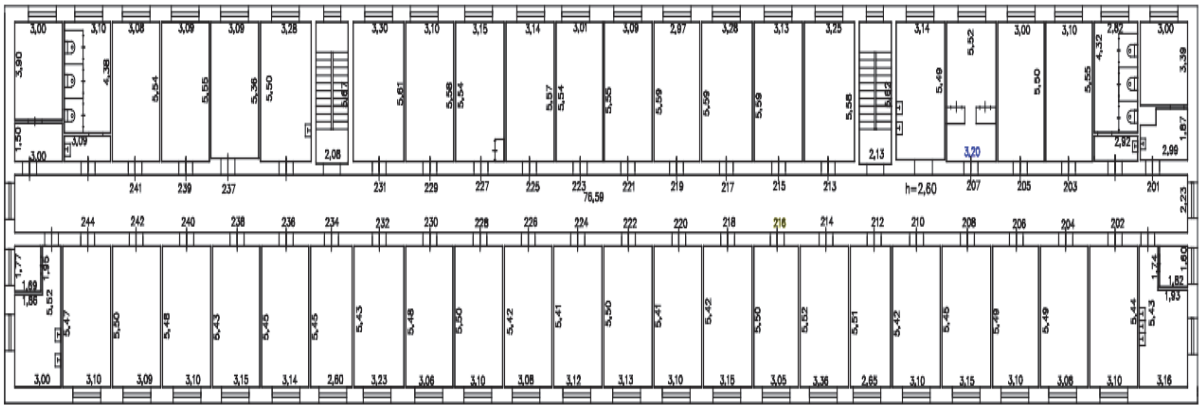


Рис. 5.2. Dwg-файл с планом второго этажа общежития 8 СФУ2

Рис. 5.3. Вид реального общежития SFU2H8

```
using System; // Программа поэтажной загрузки файлов.
using System.Collections.Generic;
using System.Linq;
using Autodesk.Revit.DB;
using Autodesk.Revit.DB.Architecture;
using Autodesk.Revit.UI;
using Autodesk.Revit.UI.Selection;
using Autodesk.Revit.ApplicationServices;
```

```
using Autodesk.Revit.Attributes;
using Autodesk.Revit.Extension;
public class LevelCreation
   {
      public static List<Level> CreateLevels(RvtDocument doc, double[]
elevations, string[] names = null, string[] pathes = null, )
      {
         List<Level> list = new List<Level>();
         for (int i = 0; i < elevations.Length; i++)
         {
            double elevation = elevations[i];
            Level level = doc.Create.NewLevel(elevation);
            if (names != null && names.Length >= i + 1)
               level.Name = names[i];
if (pathes != null && pathes.Length >= i + 1)
{
               level.Import_path = pathes[i];
                  level.SAPRImport(pathes[i], NULL, NULL);
}
            list.Add(level);
         }
         return list;
      }
}
CachedDoc.Delete(new FilteredElementCollector(CachedDoc)
               .WherePasses(new ElementClassFilter(typeof(Level), false))
               .Select(e => e.Id)
               .ToList());
List<Level> levels = LevelCreation.CreateLevels(
      CachedDoc,
      //В кавычках указываем путь к файлу
      StreamReader objReader = new StreamReader("path");
      string sLine="";
      ArrayList arrText = new ArrayList();
      while (sLine != null)
        {
          sLine = objReader.ReadLine();
          if (sLine != null)
        arrText.Add(sLine);
        },
```

```
        //Загружаем в цикле множество dwg-файлов из подкаталога dwg
          new string[]
        {
          for (int i = 1; i < 6; i++)
          {
" \dwg\L" + i + ".dwg" ;
          }
        }
    );
```

Следует отметить ошибки в исходных dwg-файлах, но после исправлений из информационной модели SFU2H8.rvt можно экспортировать файлы без ошибок. Подобные оценки для множества решений представляют трудоемкую задачу, для реального решения которой требуются инструментальные средства и информационное обеспечение.

Подобным образом были построены все модели объектов SFU2, созданы файлы с центрами координат объектов в системе LL84. Для построения дорог требуется файл с центрами координат точек излома и файл точек излома коммуникаций инженерных сетей информационных, электроснабжения, водоснабжения и канализации.

5.4. Проектирование кампуса в INFRAWORKS

Для проектирования кампуса или комплекса зданий необходим анализ участка и возможных внешних воздействий. Средой проектирования может служить САПР Autodesk INFRAWORKS. Границы участка и объектов задаются в угловых координатах, что создает возможность размещения кампуса даже на другой планете. Здания и объекты кампуса могут размещаться в двух вариантах. В виде символов в виде коробок или в виде полноценных зданий, созданных в САПР REVIT. Размещение символов требует меньше ресурсов и рекомендуется для проверки координат и взаимного расположения объектов. После проверки координат можно размещать полноценные здания из файлов типа rvt. Для проекта кампуса нужно создать каталог проекта и в нем подкаталоги dwg и rvt, где размещаются все необходимые объекты и возможна относительная адресация. Подобным образом были построены все модели объектов SFU2, созданы файлы с центрами координат объектов в системе LL84. Для построения дорог требуется файл с центрами координат точек излома и файл точек излома коммуникаций инженерных сетей (информационных, электроснабжения, водоснабжения и канализации).

В САПР Autodesk INFRAWORKS проектирование возможно с помощью команд или программных модулей. Командный режим рекомендуется для оригинальных отличий одного кампуса от другого. Использование программных модулей рекомендуется для итерационного проектирования кампуса или создания подобных кампусов.

Приведем пример функции добавления объектов САПР REVIT к модели второго кампуса Студгородок Сибирского федерального университета в САПР Autodesk INFRAWORKS и ее использование

```
function AddRvtModel(name)
{
	var model = buildingsCoords[name];

	var table = app.ActiveModelDb.Table('BUILDINGS');
	table.BeginWriteBatch();
	var w = table.GetWriteRow();

	w.NAME = model.name;
	w.DATA_PATH = "..//rvt" + model.name + ".rvt"
	w.DATA_SOURCE_ID = -1;
	w.MODEL_SCALE_X = model.sx;
	w.MODEL_SCALE_Y = model.sy;
	w.MODEL_SCALE_Z = 1;
	w.MODEL_ROTATE_X = 0
	w.MODEL_ROTATE_Y = 0;
	w.MODEL_ROTATE_Z = 0;
	w.ROOF_HEIGHT = model.sz * floorHeight;
	w.ROOF_MATERIAL = roofMaterial;

	var geom = new adsk.Geometry({
		"type": "Polygon",
		"coordinates": [[
			model.x, model.y
		]]
	});
	w.GEOMETRY = geom;
	table.Insert(w);
	table.CommitWriteBatch();
};
// Добавление моделей зданий
AddRvtModel('SFU2G2014full');
```

```
AddRvtModel('SFU2J2015');
AddRvtModel('SFU2H82015');
```

Объединяют кампус дороги различных классов, которые задаются по точкам излома. Класс дороги можно выбрать и изменить с помощью меню. Коммуникации кампуса проводятся на разной глубине и отображаются различными цветами. Коммуникации лучше добавлять после размещения зданий.

Координаты всех объектов вынесены в отдельные файлы и представляют собой три массива: buildingsCoords – координаты зданий, roadsCoords – координаты дорог, treesPoligonsCoords – координаты полигона с деревьями. Это позволяет использовать координаты в любом проекте, включая соответствующий javascript-файл. Возможно многовариантное проектирование со сменой файлов координат.

Информационные модели кампуса и зданий важны не только для проектирования зданий и их комплексов. Гораздо важнее модели на всех этапах жизненного цикла с учетом ремонта и изменения коммуникаций. После обнаружения деформаций в северном и южном крыльях главного корпуса, студентами под руководством автора был разработан проект мониторинга деформаций второго кампуса СФУ Студгородок на базе оптоволоконных датчиков с решетками Брегга с централизованной обработкой результатов для принятия решений. Проект со стоимостью оборудования был передан проректору по развитию СФУ в 2014 г.

Вычислительные системы должны проектироваться с учетом назначения и условий окружающей среды. При наличии образцов модулей студент в течение семестра может выполнить проект или создать аналогичный модуль.

Исследованы пути развития программно-методического комплекса, позволяющего производить многовариантное проектирование и сопровождение объекта, с автоматической оценкой параметров и критериев эффективности и автоматическим преобразованием результатов в форматы или командные файлы САПР схемного и конструкторского проектирования на различных уровнях иерархии. Предложен общий подход к автоматизации проектирования для объектов на разных уровнях иерархии несмотря на существенные различия в технологиях. Итерационный характер проектирования и накопление знаний об объекте активизирует процесс обучения.

6. ЛОКАЛЬНОЕ ВЫПОЛНЕНИЕ И СЕТЕВЫЕ СЕРВИСЫ МОДЕЛИРОВАНИЯ И ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

6.1. Программно-аппаратные комплексы САПР

САПР предназначена для синтеза описаний множества технических аналого-цифровых вычислительных устройств и систем. Описания возможны на различных уровнях абстракции. Описания одного или нескольких вариантов могут быть переданы в системы конструкторского проектирования.

Область применения САПР – обучение студентов и инженеров проектированию вычислительных устройств и вычислительных систем на базе микроЭВМ с ранних этапов. УИ САПР работает на ЭВМ с операционными системами Windows, Linux, OS2 и ZVM от персональных до мощных с операционной системой виртуальных машин. Предусмотрены режимы клиента и сервера, что позволяет использовать комплекс для локального и дистанционного обучения. Сетевые сервисы доступны мобильным устройствам с ОС Android. На сервере находятся: программный комплекс, методические материалы, учебные пособия на русском и английском языках.

Особенности:

1) возможность описания вычислительных систем на различных уровнях абстракции;

2) возможность описания множества технических решений;

3) унификация описания аппаратуры на языках высокого уровня в различных синтаксических средах;

4) автоматическое сравнение предполагаемых и фактических результатов;

5) автоматическая оценка ресурсов;

6) интерфейс с системами конструкторского проектирования;

7) автоматизация формирования библиотек символов и конструктивов компонентов;

8) альтернативные маршруты выполнения формализованных заданий для возможности сравнения результатов. Различие маршрутов может быть в языке описания, операционной системе или компонентах.

Основные функции: синтез описаний множества технических решений; автоматический многовариантный анализ работы вычислительных устройств с оценкой ресурсов и критериев эффективности; автоматическое сравнение предполагаемых и фактических результатов; импорт и экспорт описаний в форматы конкретных САПР.

Состав системы: библиотеки компонентов, узлов, устройств и систем; программно-методический комплекс, включающий диалоговый монитор и различные функциональные модули, учебные пособия с грифом Министерства образования [28–33].

Требования к аппаратным и программным средствам зависят от версии учебно-исследовательской САПР. Версия первая УИ САПР Код (CODVI) предназначена для диалоговых систем коллективного пользования на IBM 360 в среде OS/360 и совместимых с ними технических (ЕС ЭВМ) и программных средств.

Вторая версия УИ САПР Код (CODV2) работает в системе коллективного пользования на ЭВМ IBM S370–S390 и новых персональных серверах IBM PC ServerS390/Z с локальными или удаленными терминалами или персональными ЭВМ с модулем программной эмуляции терминала. Использование персональной ЭВМ обеспечивает графический вывод результатов. Средой является система виртуальных машин с подсистемой диалоговой обработки. Программное обеспечение доступно всем пользователям для чтения одновременно и осуществляется с двух мини-дисков. Количество пользователей ограничивается только числом терминалов. Результаты отображаются на ПЭВМ и сетевых станциях в графическом виде, а на алфавитно-цифровых терминалах – в символьном виде.

Версия третья УИ САПР Код (CODV3) предназначена для персональных IBM-совместимых машин в среде MS DOS, PC DOS или DOS-сессии OS/2. При использовании трансляторов фирмы Borland C, C++ версии 3.1, PCAD v8 объем оперативной памяти 4 Мб достаточен, но рекомендуется 8 Мб. Использование локальной сети с файл-сервером снижает требования к дисковой памяти. Версия характеризуется автоматическим интерфейсом с промышленными САПР PCAD, GRIF4, ORCAD, CADDY, EAGLE и ПРАМ5.3. Предусмотрено локальное и дистанционное выполнение формализованных заданий в сети ЭВМ, что расширяет возможности рабочего места, на котором производится синтез ФЗ и просмотр результатов. Возможно выполнение ФЗ на серверах, которые на самой рабочей станции не выполнимы. Примером могут служить формализованные задания на языках PL/1, JAVA и сложные задания на C++. Затраты на сервер значительно меньше, чем модернизация большого числа рабочих станций.

Версия четвертая УИ САПР Код (CODV4) разработана для 32-разрядных машин IBM PC для многозадачной среды OS/2 – Windows, Linux.

Оболочка УИ САПР для синтеза, локального и дистанционного выполнения ФЗ создана на базе инструментальных пакетов Visual Age фирмы IBM. Пакеты программ Visual Age созданы для основных универсальных языков программирования PL/1, C++, JAVA и работают на различных платформах. Исходный текст может быть использован для создания программных комплексов на различных платформах. Непосредственно оболочка УИ САПР создана на базе многофункционального текстового редактора LPEX (Live Parsing EXtensible editor) из пакетов Visual Age. Редактор LPEX работает на различных платформах, может запускаться самостоятельно или из браузера, например Mozilla Firefox и Google Chrome. Версия четвертая УИ САПР Код может работать как на машине клиента, так и на сервере. Операционной средой сервера может быть OS/2, Windows, Linux, VM/ESA S390, ZVM. Операционная среда сервера может быть расширена системами, для которых выпускаются инструментальные средства Visual Age или Web Sphere.

Основным направлением является переход на 64-разрядные продукты FSF-Free Software Foundation. Пятая версия (CODV5) предназначена для web-серверов в системе локального и дистанционного образования и может быть использована для решения задач 64-разрядных САПР с поддержкой стандартов информационного моделирования ГОСТ Р 57563–2017/ISO/TS 12911:2012, ГОСТ Р 57310–2016.

Методический материал обеспечивает интенсивную подготовку по синтезу и выполнению формализованных заданий. Методический материал лучше разместить на сервере СФУ (e.sfu-kras.ru) и библиотеке СФУ (bik.sfu-kras.ru), а выполнение заданий производить на сервере приложений. Преимуществом пятой версии является выполнение формализованных заданий на серверах приложений в различных средах Windows и Linux. При этом в Linux используется FCF GCC, а для Windows FSF mingw-64 с поддержкой языков – GCC и GNU ADA. Создание меню в виде расширения браузера позволяет полноценно использовать мобильные устройства.

6.2. Локальное выполнение формализованного задания на персональных ЭВМ в средах Windows, Linux и OS/2

В среде многозадачной операционной системы OS/2 [145] могут работать различные версии исследовательской САПР. Версия 3 может выполняться как DOS-программа с оболочкой для MSDOS или со специально разработанной оболочкой для OS/2 на базе языка REXX. Версия 4 системы САПР COD (CODOS) разработана для среды OS/2 и Windows. Версия 4

CODOS состоит из программных модулей, статических и динамических библиотек, командных файлов и оболочки с режимом регламентированного диалога. Оболочка служит для выполнения формализованных заданий различными проектными процедурами в локальном и дистанционном вариантах с минимальными затратами времени. В результате выполнения проектных процедур получаются проектные решения в средах различных САПР. Одному формализованному заданию соответствует одно или множество проектных решений САПР более низкого уровня. Основные подкаталоги CODOS: \BIN, \DOC, \INCLUDE, \INCPLI, \JAVA, \LIB, \LIBWIN.

Система автоматизированного технического проектирования PCAD4.5–8.5 выполняется в DOS-сессиях, а ORCAD, CADDY и PCAD2001–2006 – в среде Windows. В среде OS/2 можно использовать инструментальные средства Toolkit for OS/2, WorkFrame/2, трансляторы IBM C Set++ и IBM PL/1, интерпретатор REXX, инструментальные средства для организации локальной сети и решения задач в режиме «клиент – сервер». Однако лучше использовать интегрированные инструментальные средства Visual Age фирмы IBM для операционных систем OS/2 и Windows и языки C++, PL/1 и JAVA. В комплекс Visual Age входит редактор LPEX, на базе которого удобно строить оболочку для выполнения и редактирования формализованных заданий на проектирование с различными языками программирования. При этом редактор LPEX позволяет обнаружить синтаксические ошибки. В среде текстового редактора LPEX разработаны синтаксические анализаторы для текстового описания схемы в форматах ALT файла PCAD и файлов САПР ПРАМ 5.3. Оболочка позволяет в режиме регламентированного диалога выполнять формализованное задание на локальных машинах в среде сервера OS/2 или Windows, а также на серверах IBM S390 в операционной среде виртуальных машин (VM) или OS390 (MVS). Общность языковых сред IBM VM, MVS, Linux и OS/2 облегчает обучение и переход из одной системы в другую.

Исходные файлы для четвертой версии соответствуют каталогам COD для размещения САПР и X:\FA\CPP для размещения контрольных примеров на языке C++. Для языка PL/1 контрольные примеры размещаются в подкаталоге X:\FA\PLI, для языка ADA (VHDL) – X:\FA\ADA. Для языка JAVA служит подкаталог X:\FA\JAV с трехсимвольным типом файлов JAV и возможностью размещения общего каталога FA на дисках с файловой системой FAT. Для выполнения формализованных заданий на языке JAVA в среде OS/2 файл копируется в подкаталог Y:\FA\JAV диска Y с файловой системой HPFS, с типом файлов JAVA. Задание выполняется, и результаты пересылаются в основной подкаталог X:\FA\JAV. В среде Windows файл типа JAV копируется в файл типа JAVA в том же подкаталоге с последующим выполнением. После выполнения задания промежу-

точные файлы удаляются. Каталог CODOS содержит те же подкаталоги, что и COD для MSDOS, дополнительным подкаталогом является INCPLI, в котором содержатся файлы типа INC. Все файлы выбора точек входа в процедуры объединены в один общий файл WALL.INC и используются при большом числе различных типов компонентов. Статически подключаемые библиотеки типа LIB, динамически подключаемые библиотеки типа DLL и библиотеки импорта типа LIB находятся в подкаталоге LIB для OS/2 и в подкаталоге LIBWIN для Windows. Имена библиотек и команд выполнения формализованных заданий образуются в соответствии с табл. 4.4.

В среде Windows в переменные окружения или в AUTOEXEC.BAT включают пути:

```
SET PATH=X:\CODOS\BIN;  X:\CODOS\LIBWIN; X:\PCAD\EXE;
SET INCLUDE=X:\CODOS\INCLUDE; X:\CODOS\INCPLI;
SET LIB=X:\CODOS\LIBWIN;
SET LPATH=X:\CODOS\BIN; (первая запись)
SET PLILPATH4=X:\CODOS\BIN; (первая запись)
SET CODLOC=X; (верхний регистр).
```

Анализ поведения объектов и оценка ресурсов, полученных в различных средах с помощью неодинаковых инструментальных средств, повышает достоверность результатов.

Формализованные задания на языке FSF C++ не зависят от среды и типа транслятора (GCC и CSET2 фирмы IBM), поэтому формализованное задание на C++ версии 3 COD и версии 4 CODOS совпадают, и можно использовать общий подкаталог X:\FA\CPP. Тексты формализованных заданий на языке PL/1 (типа PLI) для сред VM и OS/2 тоже совпадают. Формализованные задания типа PLI могут выполняться автономно на рабочей станции, на сервере в локальной сети или на удаленном сервере. Формализованное задание с моделями компонентов будет выполняться на серверах S390 при включении модели в главную процедуру. В операциях логического сложения и сцепления вместо символа «вертикальная черта» (|) используется «восклицательный знак» (!). В средах OS/2 и Windows символ операции можно не задавать.

Работа в среде УИ САПР для OS/2 и Windows в основном аналогична работе в среде MSDOS. Предусмотрено локальное и удаленное выполнение формализованных заданий на рабочей станции, на серверах OS/2, Windows, VM S390 (VMz) и OS390 (zOS). Графическое отображение результатов на рабочей станции выполняется программами UISGO для OS/2 и UISGW для Windows. Программа отображения вызывается в меню результатов (Result) в пункте *График* (Graphics), но для выполнения следующего задания из нее нужно выйти. Можно до начала работы вызвать

программу отображения и редактор, а затем осуществлять переход между процессами синтеза формализованного задания в редакторе и графическим отображением результатов анализа. Таблицы сравнения и оценки сигналов, оценки ресурсов и критериев эффективности для всех вариантов автоматизированного анализа выводятся в файлы сообщений типа MSG.

Таблица 6.1

Таблица результатов выполнения формализованных заданий

Имя одкоманды	Назначение	Вид результата	Представление результата
Command mode	Режим команд		
Modelling	Моделирование	name.lsg, name.msg	Диаграмма работы
Interface PCAD 45	Интерфейс PCAD 45	Table	Двоичный вариант схемы
		Command	Командный файл
Interface PCAD 85	Интерфейс PCAD 85	Table	Двоичный вариант схемы
		Command	Командный файл
Interface ORCAD	Интерфейс ORCAD	Command	Командный файл
Interface PCAD 2001–2006	Интерфейс PCAD 2001–2006	Command PCAD 2001–2006	Командный файл PCAD 2001–2006
Interface CATIA	Интерфейс CATIA	Command CATIA	Проект в CATIA
Interface VRML	Интерфейс VRML	name.wrl	Объемный образ (3D)
Interface X3D	Интерфейс X3D	name.x3d	Объемный образ (3D)
Interface WebGL	Интерфейс WebGL		Объемный образ (3D)
Format FZ COD	Формат ФЗ УИ САПР	FZ: C++, ADA, JAVA, PL/1	ФЗ на языках C++, JAVA, PL/1
Format STEP (EXPRESS)	Формат Р ИСО 10303	name.stp, name.exp	Описание проекта
Format EDIF	Формат EDIF	name.edf	Описание проекта
Format PRAM 53	Формат ПРАМ 53	On components	По компонентам
		On Nets	По цепям
Format HTML	Формат HTML, XML	name.htm, ame.xml	Принципиальная схема
Format CADDY	Формат CADDY		Принципиальная схема
Format EAGLE	Формат EAGLE	name.xml	Принципиальная схема
Format TSCH	Формат варианта схемы	name.tsh	Таблица варианта схемы
Format ALT	Конструктив (ALT)	PCAD 45	PCAD 45
		PCAD 85	PCAD 85
		PCAD 2001–2006	PCAD 2001–2006

Новым является возможность вывода описания на языке электронных схем (SML) с отображением в любой программе просмотра, поддер-

живающей язык XML [28]. Одновременно с просмотром схемы или конструкции модуля можно наблюдать результаты моделирования.

Режим регламентированного диалога можно строить на базе различных инструментальных средств: многофункциональных редакторов EPM, LPEX или web-браузеров, например дополнений Firefox. Дополнения web-браузеров предпочтительны для мобильных устройств. Многофункциональные редакторы могут вызываться из web-браузеров после установки соответствия прикладных программ по типу файла. Синтез формализованных заданий на языках высокого уровня лучше выполнять в среде многофункциональных редакторов с синтаксическим анализом заданий на различных языках. Формализованные задания просматриваются в различных окнах редактора, и с помощью ниспадающего меню организуются автоматизированный анализ и оценка ресурсов и критериев эффективности. В режиме «моделирование» открывается окно с режимом командной строки, куда выводятся результаты трансляции, редактирования и выполнения формализованного задания. При нулевом или отрицательном уровне результата, задаваемом параметром LRES, символьная информация выводится в то же окно, а временная диаграмма отображается графически в отдельном окне. В зависимости от заданного уровня результата выводится различная информация. При значении уровня 2 фактические результаты сравниваются с предполагаемыми и выводятся такты с отличиями. Возможен просмотр схемы в среде PCAD, конкретный модуль которого выбирается в меню. Автоматический интерфейс УИ САПР PCAD также поддерживается меню. Часто используемые функции дублируются пиктограммами.

В табл. 6.2 приведены основные команды редактора EPM. Возможно открытие множества файлов для редактирования путем задания символа «звездочка» (*) в имени или типе файла. Открыть файл для редактирования можно и после команды DIR, подведя указатель к требуемому файлу и нажав клавиши Alt+1.

Редактор EPM настраивается с помощью специального файла (PROFFILE.ERX) по команде PROFILE ON. Режим следует сохранить командой SAVE в меню OPTIONS. В среде редактора возможен режим командной строки для OS/2 и MSDOS. Фактически действия выполняются с помощью командных файлов на языке REXX. Например, автоматизированный анализ поведения объекта, структура которого описана формализованным заданием на языке C++, выполняется с помощью командного файла FCOF.CMD, соответствующего программному комплексу Visual Age C++. Автоматизированный анализ поведения объекта, структура которого описана в формализованном задании на языке PL/1, выполняется с помощью командного файла FPOF.CMD, а задания на языке JAVA – с помощью файла FJOF.CMD. Все командные файлы размещаются в каталоге

X:\CODOS\BIN. В комплексе Visual Age утилита работы с библиотеками ILIB.EXE, с библиотеками импорта – IMPLIB.EXE. УИ САПР включает статические и динамические библиотеки. Статические и динамические библиотеки для различных приложений объединены в подкаталог X:\CODOS\LIB для среды OS/2 и X:\CODOS\BIN для среды Windows. Несмотря на все достоинства редактора EPM, на базе которого была разработана оболочка комплекса, предпочтение было отдано многофункциональному редактору LPEX, входящему в комплексы IBM Visual Age, Web Sphere и браузерам Firefox и Chrome. Инструментальные средства Visual Age выпускаются фирмой IBM для различных платформ и языков. Используя средства Visual Age, можно иметь единые исходные тексты программ для различных платформ.

Браузерные версии меню для серверов приложений предпочтительны при переходе к сетевым образовательным технологиям и онлайн-курсам. Но это стало возможным после унификации дополнений меню для последних версий браузеров Firefox и Chrome. Редактор LPEX поддерживает основные языки программирования, имеет несколько профилирующих файлов и обеспечивает интерфейс редакторов EPM (табл. 6.2), Firefox (табл. 6.1). Запуск заданий производится из меню при наличии командного окна. Символьные результаты выводятся в окно редактора, а графические – в специальное окно.

В соответствии с методическим обеспечением УИ САПР меню редактора LPEX или браузера Firefox дополнено следующими пунктами: *Установка параметров* (Set parameters), *Выполнение* (Execute), *Регистрация* (Registration), *Результат* (Result), *Удаленное выполнение* (Remote Execute), *Эффективность* (Effection), *Информация* (Information).

Пункт *Установка параметров* позволяет: выбрать удаленный сервер (Choice HOST), язык программирования (Language), вид описания для импорта (CADIN), вид инерфейса с САПР, формат чертежа (Format of frame), вид библиотек (Library).

Пункт *Регистрация* позволяет пройти полную регистрацию (Full registration) или ввести имя формализованного задания (Name of task). Имя формализованного задания вводится без типа файла (расширения). Полная регистрация позволяет задать данные, необходимые для заполнения штампа чертежа.

Пункт *Выполнение* (Execute) позволяет на рабочей станции выполнить моделирование (Modelling) или интерфейс с одной из промышленных САПР. Интерфейс может быть командным (Command) или табличным (Table). В командном интерфейсе создается файл, содержащий последовательность команд, необходимую для создания схемы. Табличный интерфейс позволяет преобразовать формализованное задание во внутреннюю

структуру схемы в конкретной САПР. Табличный и командный интерфейсы реализованы для САПР PCAD. Для САПР ORCAD, PCAD200x, EAGLE, Cadstar реализован командный и для САПР CADDY – программный интерфейсы. Командные интерфейсы объединены в общие библиотеки LGOI и LGWI.

Таблица 6.2

Команды редакторов EPM и LPEX (режим EPM)

Функциональная клавиша	Назначение
F1	Помощь
F2	Сохранить текущий файл
F3	Закрыть текущий файл. Выход из редактора
F4	Сохранить и закрыть текущий файл
F7	Удалить текущий файл
F8	Открыть новый файл для редактирования
F9	Возврат к текущей строке
F11 или Ctrl+N	Переход к предыдущему открытому файлу
F12 или Ctrl+P	Переход к последующему открытому файлу
Home	Переход к началу строки
End	Переход в конец строки
Ctrl+Home	Переход к началу файла
Ctrl+End	Переход в конец файла
Ctrl+Backspace	Удаление текущей строки
Ctrl+E	Удаление части строки, от курсора до конца
Alt+B	Выделить фрагмент текста
Alt+C	Скопировать выделенный фрагмент
Alt+D	Удалить выделенный фрагмент
Alt+=	Выполнение текущей строки как команды

Пункт *Результат* (Result) позволяет просмотреть результат анализа в графической форме (Graphics) или файл сообщений в текстовой форме (View MSG file), ознакомиться со схемой (Table) или наблюдать выполнение командного файла (Command) в САПР. Для просмотра результатов нужно в конкретной САПР открыть файл формата схемы и выполнить файл макрокоманд с именем формализованного задания.

Специализация редактора производится с помощью файла PROFILE.LX, который находится в подкаталоге X:\CODOS\BIN, и файлов типа LX. Путь к ним нужно указать первым в операторе SET LPATH для редактора LPEX пакета Visual Age или IBM Web Sphere. Используется несколько профилирующих файлов, из которых первым загружается PROFINIT.LX, затем редактируемый файл, профилирующий файл PROFSYS.LX, а после этого PROFILE.LX. Особенности языка конкретизируются с помощью файлов

типа LXL, имена которых соответствуют типу файлов CPP, JAVA (JAV), PLI, CMD языков C++, REXX.

Особенностью транслятора PL/1 является возможность задания соответствия между файлом и набором данных с помощью команды SET. Например, выполнение формализованного задания использует файл SYSIN для набора входных данных типа DAT, файл DBT – для набора данных UIPCAD.DBT, файл спецификации SYSOUT – для набора данных типа LST, файл результатов анализа GOUT – для набора данных типа LSG или OUTPUT.000. Примеры команд SET для командного файла на языке REXX, в котором имя формализованного задания присвоено переменной NAME, приведены ниже:

```
’SET DD:SYSIN=’name’.dat’
’SET DD:DBT=’uipcad.dbt’
’SET DD:SYSOUT=’name’.lsp’
’SET DD:GOUT=’name.lsg’.
```

Аналогично можно связывать имена файлов и наборов данных для языка C++ в рамках инструментальных средств Visual Age и трансляторов для операционных систем VM/ESA и OS/390. Для совместимости с другими трансляторами для языка C++ используется функция чтения переменных окружения getenv(). Динамическое связывание имен файлов и наборов данных перед шагом выполнения задания важно для совместимости клиентских и серверных версий программного комплекса CODOS. На рабочей станции после регистрации и установок информация об имени формализованного задания и рамки чертежа находятся в файле COD.INI. На серверах задания на выполнение поступают в произвольные моменты времени, и передача параметров производится в командной строке или с помощью установки переменных окружения. Поэтому на языке C++ производится проверка наличия переменных окружения CODLOC для символа диска с установленной системой CODOS, CODE для имени формализованного задания и FRAME для имени рамки (A0–A4) типа SCH. Примеры выполнения команд для автоматизированного анализа формализованного задания приведены ниже. Выбрав в основном меню пункт *Выполнить* (Execute) и подпункт *Моделирование* (Modeling), выполняют команду редактора fexec.lx, которая в зависимости от операционной системы и выбранного языка выполняет командный файл OS/2 типа CMD или командный файл Windows типа BAT в соответствии с табл. 4.4. Например, для языка C++ в среде Windows моделирование выполняется командой fcwf.bat, а для среды OS/2 – командой fcof.cmd. Для языка JAVA моделирование в среде OS/2 выполняется командой fjof.cmd, а в среде Windows – командой fjwf.bat. Все командные файлы типов LX, BAT и CMD находятся в подкаталоге

X:\CODOS\BIN. Примеры созданных команд fcwf.bat,fjwf.bat и fpwf.bat для Windows приведены ниже:

```
REM C++: fcwf.bat, %1-имя ФЗ, %2-тип библиотеки (LIB или DLL)
icc  /qautoimported /Ss /G4 /Q /W2 /C  %1.cpp
IF "%2"=="DLL" goto dll
goto lib
:DLL
ilink %1.obj /pm:vio /noe  icwf.lib icwbits.lib icwintv.lib
goto end
:LIB
ilink %1.obj /pm:vio /noe  lcwf.lib lcwbits.lib lcwintv.lib
:end
SET DD:CODE=%1
SET DD:Lang=%3
%1.exe
REM   JAVA: fjwf.bat, %1-имя ФЗ, %2-имя диска
SET CLASSPATH=c:\java\lib;%2:\codos\java\ljof.zip;%2:\fa\jav
IF EXIST %1.lsg del %1.lsg
IF EXIST %1.msg del %1.msg
if exist %1.java goto begin
if exist %1.jav copy %1.jav %1.java
:begin
javac %1.java >%1.err
echo java begin....
SET user=%1,%2,,,%3,
if exist %1.class java -Duser.name=%user% %1 >%1.msg
goto end
:delete
if exist %1.class del %1.class
if exist %1.java del %1.java
:end@echo off

REM PL/1: fpwf.bat, %1-имя ФЗ, %2-имя диска ,%3-тип библиотеки
pli %1.pli   ( m s  default(linkage(system)) langlvl(saa2))
IF "%2"=="DLL" goto dll
goto lib
:DLL
ilink %1.obj  /nologo ipwf.lib
goto end
:LIB
```

```
ilink %1.obj  /nologo lpwf.lib
:end
SET DD:SYSIN=%1.dat
SET DD:DBT=%3codos\bin\uipcad.dbt
SET DD:GOUT=%1.lsg
SET DD:SYSOUT=%1.lsp
SET DD:SYSPRINT=%1.msg
SET DD:Lang=%4
%1.exe
```

Пакет разработки JDK JAVA входит в состав Windows, Linux и OS/2, устанавливается при инсталляции операционной системы. В конфигурационном файле CONFIG.SYS OS/2 формируются следующие операторы:

```
LIBPATH=X:\JAVA\DLL
SET PATH=X:\JAVA\BIN;
SET INCLUDE=X:\JAVA\INCLUDE; X:\JAVA\INCLUDE\OS2;
SET LIB=X:\JAVA\LIB;
    SET CLASSPATH=X:\JAVA\LIB\CLASSES.ZIP,
```

где X – имя диска.

В среде Windows пакет разработки JDK JAVA устанавливается дополнительно в подкаталог JAVA и в файле AUTOEXEC.BAT, или в переменные окружения нужно включить операторы:

```
SET PATH=C:\JAVA\BIN;
SET INCLUDE=C:\JAVA\INCLUDE; C:\JAVA\INCLUDE\WIN32;
SET LIB=C:\JAVA\LIB;
SET CLASSPATH=C:\JAVA\LIB\CLASSES.ZIP.
```

Пример командного файла fcof.cmd на языке REXX для выполнения ФЗ на языке C++ в среде OS/2 приведен ниже, где переменная name – имя ФЗ:

```
/* fcof.cmd */
parse arg names
name = word(names,1);  libsys = word(names,2);

DN=VALUE('CODLOC',,'OS2ENVIRONMENT')
'IF EXIST 'name'.lsg del 'name'.lsg'
'icc /Ss /G4 /Q /W2 /C ' name'.cpp'
if libsys='DLL' then    'ilink' name'.obj icof.lib icobits.lib icointv.lib'
```

```
 else    'ilink' name'.obj lcof.lib lcobits.lib lcointv.lib'
'SET DD:CODE='name
name
```

В примерах OS/2 установлена на диск Q, а Windows или MSDOS – на диск C (H). В случае установки на других дисках необходимо заменить имя диска.

При установке УИ САПР с использованием инструментальных средств Visual Age в операционных системах Windows и OS/2 могут возникнуть вопросы. На некоторые из них в табл. 6.3 приведены ответы.

Таблица 6.3

Решение вопросов при установке УИ САПР

Вопрос	Ответы для операционных систем	
	WINDOWS	OS/2
Не загружается дополнительное меню CODOS при загрузке редактора	Проверить установки переменных окружения	
	в файле autoexec.bat: set LPATH, set PLILPATH4	в файле config.sys: set LPATH, set LPATH2
Не включаются в ФЗ файлы из подкаталогов INCLUDE и INCPLI	Проверить установки в конфигурационных файлах	
	в файле autoexec.bat: set Include	в файле config.sys: set Include
Не загружается PCAD в DOS-сессии	Наличие файла PCADDRV.SYS в корневом каталоге диска, на котором установлен PCAD	
	PCAD работает только в режиме полного экрана	
	Наличие путей для операционной системы OS/2 в файле config.sys и autoexec.bat для DOS-сессии, а для Windows – только в autoexec.bat	
Обращение к подкаталогу X:\FA\CPP на диске, отличном от заданного X:	Проверить установки в конфигурационных файлах	
	в файле autoexec.bat: set codloc	в файле config.sys: set codloc
Пакет JDK установлен в каталог X:\JAVA, а формализованные задания на языке Java не выполняются	Не установлены пути к основным компонентам JDK	

Файл конфигурации OS/2 CONFIG.SYS дополняется указанием путей к командным файлам и модулям в строке SET PATH=D:\CODOS\BIN, к включаемым файлам в строке SET INCLUDE=D:\CODOS\INCLUDE и D:\CODOS\INCPLI, к динамическим библиотекам в строке LIBPATH= D:\CODOS\LIB. Для использования системы САПР COD с различными операционными системами рекомендуется устанавливать ее на диск, доступный всем системам, например D. Там же располагается каталог FA с контрольными и рабочими примерами, с библиотеками компонентов и рамками чертежей для промышленных САПР.

6.3. Сетевые сервисы и выполнение формализованных заданий в среде Internet

Пятая версия программно-методического комплекса УИ САПР предназначена для обучения методам проектирования и основам САПР ЭВМ. Реализована пятая версия с помощью стандартных средств Internet: web-сервера и программ просмотра (Mozilla, Netscape Communicator или Microsoft Internet Explorer) для IBM PC или программ просмотра web-страниц для сетевых мобильных компьютеров [153].

Программно-методический комплекс УИ САПР COD (ПМК) состоит из свободно распространяемого web-сервера Apache [31, 153] (www.apache.com) для различных платформ, программного обеспечения CGI (Common Gateway Interface) интерфейса, методического и программного обеспечения для обучения и выполнения формализованных заданий. Программное обеспечение CGI-интерфейса реализовано на базе языка REXX и PHP, имеющегося для различных операционных систем [27, 31]: OS/2, Windows, LINUX, VM/ESA, zVM. Все перечисленные операционные системы обеспечивают использование многопроцессорных вычислительных комплексов [4, 27, 31, 51, 53, 119, 137].

ПМК доступен в сети Интернет по адресам http://e.sfu-kras.ru, http://study.sfu-kras.ru и на оптическом диске (CD-ROM) в зависимости от возможностей пользователя и пропускной способности сети. Использование ПМК на сетевом сервере позволяет пользоваться методическим и программным обеспечением без его установки у пользователя. Для возможности просмотра результатов моделирования нужно установить программу просмотра цифровых и аналоговых сигналов UISGO.EXE для OS/2 и USGW.EXE для Windows, которая вызывается по типу файла lsg.

Методические материалы представлены на русском и английском языках программой курса «Методы проектирования и САПР ЭВМ», рассчитанной на два семестра, и контрольными вопросами по каждому из семестров.

Основное внимание в осеннем семестре уделяется моделированию вычислительных устройств и систем, способам построения микромоделей и макромоделей компонентов, сравнению предполагаемых и фактических результатов. Сложность задач проектирования, решаемых на ПМК, относится ко второму уровню [27, 103], соответствующему множеству проектных решений.

В процессе автоматизированного анализа производится автоматическая оценка ресурсов и вычисление критериев эффективности, что облегчает структурную оптимизацию вычислительных устройств и систем.

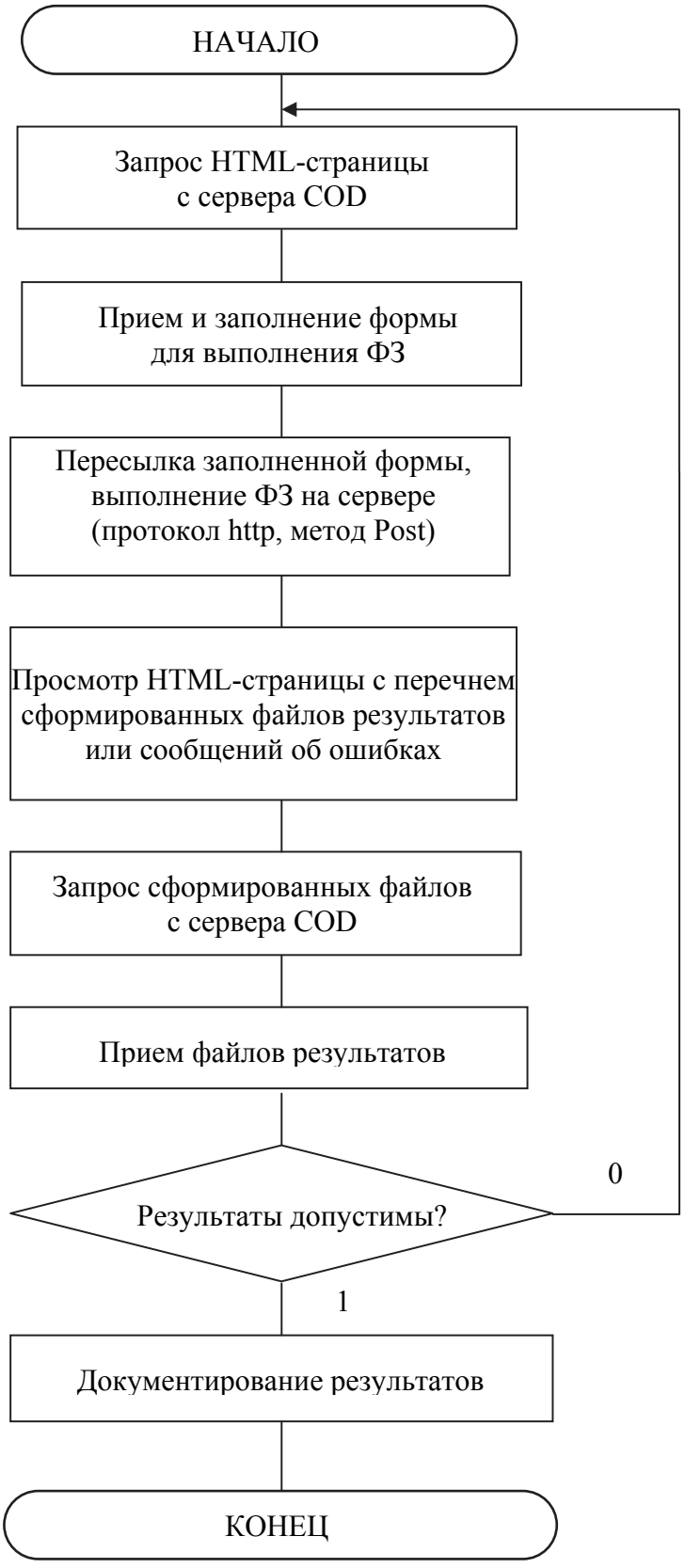


Рис. 6.1. Блок-схема маршрута выполнения формализованных заданий в различных операционных системах в среде Internet

В следующем семестре основное внимание уделяется конкретным промышленным САПР и автоматическому преобразованию высокоуровневого описания множества вариантов в одном формализованном задании во множество схем для конкретной САПР. Для использования зарубежных САПР EAGLE, PCAD 2001–2006, ORCAD, CADDY необходимы библиотеки отечественных компонентов. Библиотеки компонентов можно сформировать на сервере автоматически при наличии информационного обеспечения в форме таблиц.

Содержание оптического диска и www-сервера дополнено учебными версиями САПР электронной аппаратуры (PCAD, ORCAD, CADDY) и временными версиями инструментальных средств Visual Age для выполнения заданий, разработки и развития комплекса для различных операционных сред (Windows, OS/2, LINUX). Программно-методический комплекс (ПМК) состоит из программного обеспечения в каталогах COD и CODOS, контрольных примеров и библиотек компонентов в каталоге FA. Содержание CD-ROM и web-сервера обеспечивает выполнение лабораторных, курсовых и дипломных работ в локальном и дистанционном вариантах в сети ЭВМ и способствует индивидуализации обучения.

Маршрут выполнения формализованных заданий в различных операционных системах в среде программы просмотра web-страниц приведен на рис. 6.1. Формализованное задание (ФЗ) должно быть сформировано пользователем и послано на сервер, или можно выполнить ФЗ из множества примеров, имеющихся на сервере. При первом обращении к серверу приложений рекомендуется выполнить приведенный пример проекта для проверки работоспособности всей системы.

Подключившись к серверу, на котором находится ПМК COD V.5, пользователь знакомится с пособиями и находит HTML-страницу для внесения данных в форму выполнения формализованного задания.

Заполняется имя ФЗ и вносятся требуемые результаты. В качестве результатов может быть файл с данными анализа и оценки ресурсов, схема в формате конкретной САПР, например PCAD, или командный файл для выполнения в одной из САПР электронной аппаратуры (PCAD, ORCAD, ALTIUM, EAGLE, CADDY). После заполнения требуемых результатов и подтверждения готовности HTML-страница пересылается на сервер, где и выполняется ФЗ.

В результате пользователь получает перечень сформированных файлов или сообщения об ошибках. После приема сформированных файлов проверяется допустимость результатов. В случае допустимости результаты сохраняются, в противном случае необходимо повторить синтез или исправить ФЗ.

Пособия и методические материалы лучше разместить на сетевом сервере, а выполнение формализованных заданий производить на множестве серверов приложений в различных операционных системах. Пользователь может работать в одной операционной системе с сетевой программой просмотра, а выполнять ее в различных средах [22–26, 47, 50, 53]. К серверу приложений могут подключаться различные сетевые устройства для сравнения результатов моделирования и эксперимента (рис. 6.2). Использовано аналого-цифровое устройство фирмы National Instruments USB DAQ и программное обеспечение Lab View 8.2–8.5 [29].

Рассмотрим формализованное задание для моделирования и работы с реальным usb-модулем NI6009 на БИС в файле fp04ninp.cpp:

```
#include <uicpp.h>
void main() // fp04ninp
  {
 int noutdac1=2; // номер выхода dac1
 int noutdac2=3; // номер выхода dac2
 int mninadc[8]={4,0,0,0,0,0,0,0}; // массив номеров цепей данных АЦП
 int mndio1[8],
 mndio2[4]={5,6,7,8}; /* соответствие номеров цепей и выводов
          mndio2[0],
          mndio2[1] - управляющие входы мультиплексора
          mndio2[2] - строб
          mndio2[3] - управляющий сигнал генератора тока
               */
 int mode;  // режим работы АЦП: 10126 – дифф., 10083 – недифф.
 #include <txtpar>
INIT:  /* Установка начальных значений */
  nvarmax=1; ns=16; ntmax=20; nas=4; lres=-1; deltp=500; deltn=500; // нс
  mode=10083;
  #include <txtdcl>
INPUT: /* Генерация внешних воздействий */
  SignalA(2,2*cos(xt/3+1.0)+2,ntmin,ntmax);  // аналоговый сигнал 1
  SignalA(3,2*sin(xt/3+1.0)+2,ntmin,ntmax);  // аналоговый сигнал 2
UNIT:  /* Описание устройства */
   AD-
CUSB("NI6009",1,noutdac1,noutdac2,mninadc,mndio1,mndio2,mode);
    MKM("M686",1);
    APrint (5,12,ntmin,ntmax,2,4);
     #include <w>
}
```

Подключение аналого-цифровых устройств с цифровыми и аналоговыми входами-выходами по USB или сетевому интерфейсу обеспечивает высокую скорость обмена при сохранении гарантии на технические средства и вычислительные системы. Аналогично к программируемому технологическому оборудованию подключаются сетевые устройства на базе микроЭВМ с доступом через сетевые программы просмотра информации Mozilla Firefox, Google Chrome. Поэтому производственный персонал может управлять технологическим процессом через высокоскоростную сеть независимо от места расположения объектов. Это особенно важно для распределенных объектов, например в энергетике.

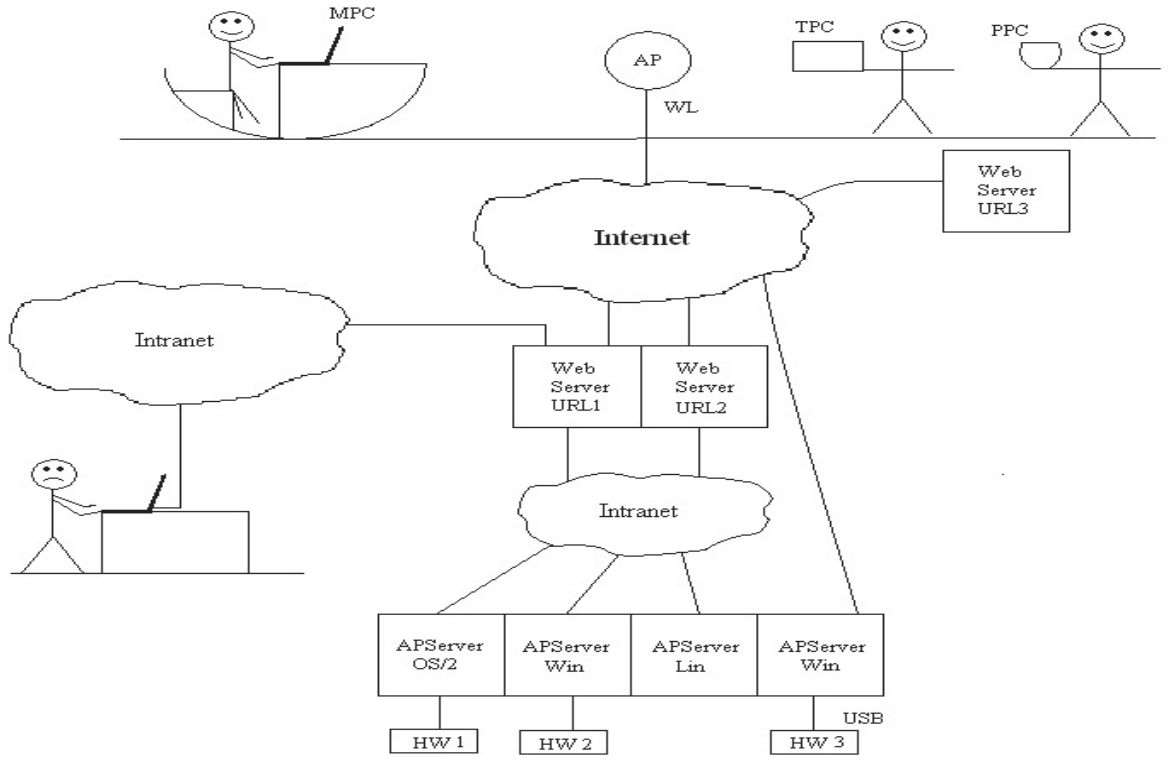


Рис. 6.2. Фрагмент сети с web-серверами, серверами приложений APServer, стационарными и мобильными клиентами. К серверам приложений подключены различные аппаратные комплексы HWi

Меню работающего многоплатформенного сервера приложений приведено на рис. 6.3. Основные пункты меню выполнения формализованных заданий соответствуют локальному и удаленному варианту меню в редакторе Lpex или браузерах Firefox, Chrome.

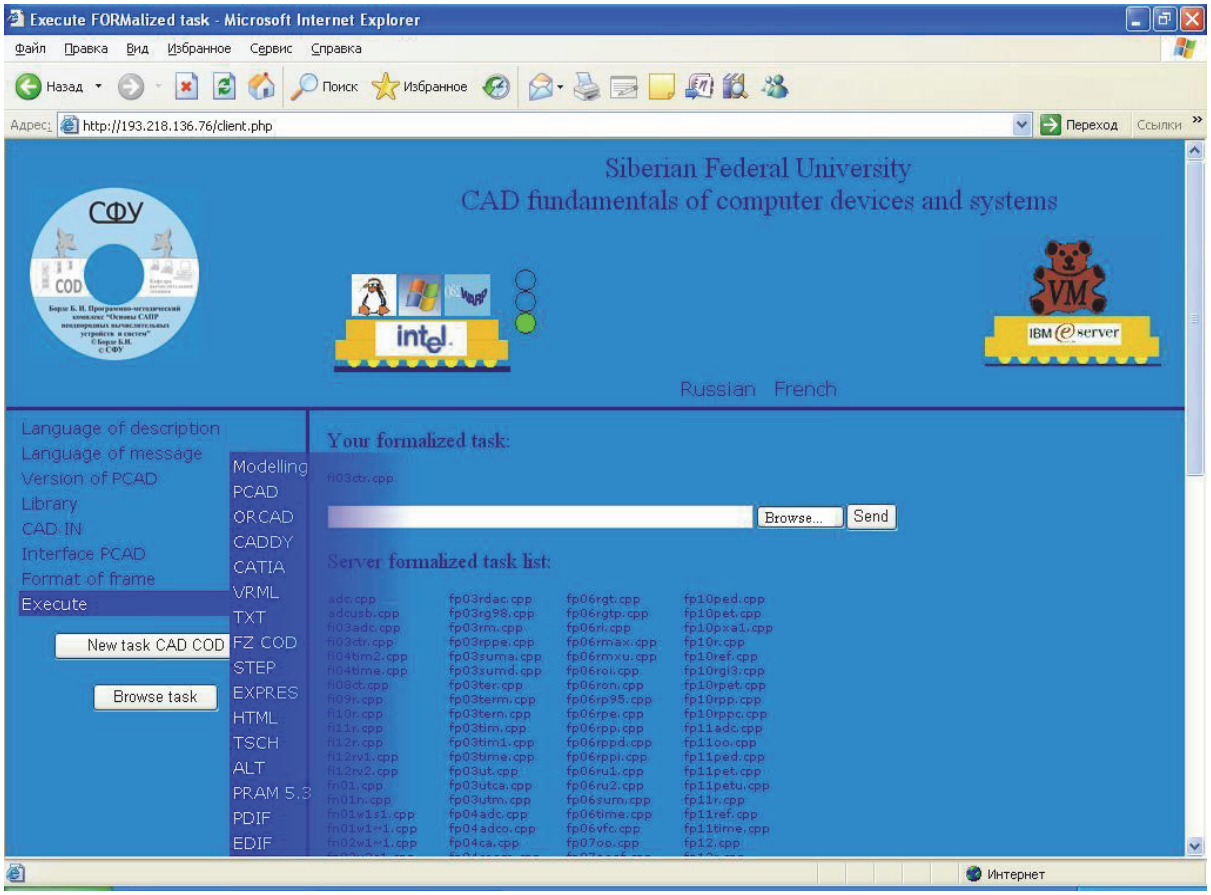


Рис. 6.3. Меню работающего многоплатформенного сервера приложений

Сетевые устройства на базе микроЭВМ являются специализированными серверами приложений с сетевым адресом, управлением и доступом с помощью сетевой программы просмотра. Разработанные для разных платформ программы просмотра временных диаграмм, схем и трехмерных моделей с отображением сигналов и температуры найдут применение в таких системах.

ЗАКЛЮЧЕНИЕ

Выполненная автором работа по методам автоматизации моделирования и проектирования неоднородных вычислительных систем позволила получить следующие результаты: создана методология и решена проблема автоматизации многовариантного объектно-компонентного моделирования и проектирования на разных уровнях абстракции создания объектов и разработан открытый модульный развиваемый многоуровневый программно-методический комплекс. В процессе проектирования формируется таблица решений, и качество изделия зависит от числа эффективных итераций создания объекта.

Основные научные и практические результаты приведены ниже:

1. Решена проблема создания открытого модульного развиваемого многоуровневого программно-методического комплекса, позволяющего из одного формализованного описания множества вариантов объекта получить необходимые представления проектируемого объекта, ресурсы для создания вариантов, критерии эффективности для принятия решения и получить описания для следующего этапа проектирования.

2. Созданы представления моделей множества технических решений вычислительных систем на различных уровнях абстракции на языках высокого уровня в различных синтаксических средах и алгоритмы их построения для различных приложений. Новое совмещение в одном описании вычислительных систем на различных уровнях абстракции и алгоритмы их преобразования в различных синтаксических средах снижают объем описаний, позволяют обеспечить концептуальное единство и снизить возможность ошибки среды.

3. Созданы методология и алгоритмы работы обобщенных многофункциональных моделей компонент с общим интерфейсом. Новые многофункциональные модели компонент с общим интерфейсом связаны с общими данными для различных моделей и приложений, и позволяют из одного описания объекта получить множество приложений. Многофункциональные модели компонент с общим интерфейсом значительно снижают трудоемкость проектирования, в особенности на начальных этапах.

4. Созданы алгоритмы инструментальных средств автоматической оценки ресурсов, сравнения предполагаемых и фактических результатов аналого-цифровых подсистем для структурной оптимизации. Переход на второй уровень сложности задач проектирования невозможен без программной поддержки принятия решений. Программная поддержка

принятия решений включает информационное обеспечение, программы оценки ресурсов и основных критериев эффективности. Наряду с традиционными параметрами информационное обеспечение дополнено энергетическими и тепловыми характеристиками компонент, позволяющими оценить температуру компонент и энергию решения задачи или сервиса.

5. Созданы модели и алгоритмы реализации модульных интерфейсов с системами схемотехнического и конструкторского проектирования. Единое представление варианта объекта преобразуется модульным комплексом в представления, воспринимаемые множеством разнообразных специализированных и комплексных систем схемотехнического и конструкторского проектирования. Построены табличный, командный и текстовый варианты моделей интерфейсов. Наибольшим быстродействием отличается табличный интерфейс, наглядностью и эффектом обучения – командный. Особо следует выделить автоматическое создание формализованных заданий в различных синтаксических средах.

6. Создана методология проектирования и использования локального и сетевого сервисного вариантов программно-методического комплекса для различных платформ. Программно-методический комплекс создавался на различных этапах развития аппаратных и программных средств. Открытый модульный принцип с единым представлением варианта объекта в различных синтаксических средах позволил развить сервисы системы виртуальных машин с возможностями серверов, рабочих станций и мобильных клиентов. Основная передача результатов производится в текстовой форме с интерпретацией на стороне клиента в сетевой программе просмотра результатов, что позволяет минимизировать трафик.

Основные научные работы (см. табл. П3.1) выполнены автором по важной для Российской Федерации и Красноярского края тематике в соответствии с отраслевыми планами, актами испытаний, внедрения и использования результатов. Результаты использованы в учебном процессе на различных этапах развития информационных технологий с рекомендацией учебных пособий и программно-методического комплекса учебно-методическим управлением и Министерством образования России.

Автоматический многовариантный анализ работы вычислительных систем с оценкой ресурсов и критериев эффективности позволяет пользователю решать задачи проектирования второго уровня сложности и сосредоточиться на творческих процедурах синтеза и принятия решения, сформировать необходимые представления проектируемого объекта, ресурсы для создания вариантов, критерии эффективности для принятия решения и получить описания для следующего этапа проектирования.

СПИСОК ЛИТЕРАТУРЫ

1. Актуальные проблемы моделирования в системах автоматизации схемотехнического проектирования / под ред. А. Л. Стемпковского. – М. : Наука, 2003. – 430 с.

2. Артамонов, Е. И. Структурное проектирование систем / Е. И. Артамонов // Информационные технологии в проектировании и производстве. 2008. – № 2. – С. 3–10.

3. Артамонов, Е. И. Интерактивные системы. Синтез структур / Е. И. Артамонов. – М. : Инсвязьиздат, 2010. – 185 с.

4. Александров, А. Спирали аппаратной виртуализации. SOA для сетей / А. Александров // Открытые системы. – 2007. – № 3. – С. 13–37.

5. Алексеев, А. В. Программирование в подсистеме диалоговой обработки СВМ ЕС / А. В. Алексеев. – М. : Радио и связь, 1990. – 256 с.

6. Аналоговые и цифровые интегральные схемы / под ред. С. В. Якубовского. – М. : Сов. радио, 1979. – 336 с.

7. Арайс, Е. А. Моделирование неоднородных цепей и систем на ЭВМ / Е. А. Арайс, В. М. Дмитриев. – М. : Радио и связь, 1982. – 160 с.

8. Армстронг, Дж. Р. Моделирование цифровых систем на языке VHDL : пер. с англ. / Дж. Р. Армстронг. – М. : Мир, 1992. – 175 с.

9. Общие принципы организации проектирования в КАСПИ / Б. А. Бабаян, О. К. Гущин, Г. Г. Рябов, А. М. Степанов. – М. : ИТМиВТ, 1975. – 32 с.

10. Башмаков, А. И. Интеллектуальные информационные технологии : учеб. пособие / А. И. Башмаков, И. А. Башмаков. – М. : Изд-во МГТУ им. Н. Э. Баумана, 2005. – 304 с.

11. Балашов, Е. П. Проектирование информационно-управляющих систем / Е. П. Балашов, Д. В. Пузанков. – М. : Радио и связь, 1987. – 256 с.

12. Борде, Б. И. Многоуровневая структурная оптимизация неоднородных вычислительных систем / Б. И. Борде // Вестн. Краснояр. гос. ун-та. Сер. Физико-математические науки. – 2006. – Вып. 7. – С. 155–161.

13. Борде, Б. И. Моделирование неоднородных вычислительных систем / Б. И. Борде // Вестн. Краснояр. гос. ун-та. Сер. Физико-математические науки. – 2005. – Вып. 4. – С. 197–202.

14. Борде, Б. И. Многофункциональные модели компонентов неоднородных вычислительных систем / Б. И. Борде // Вычислительные технологии. – 2005. – Т. 10. Спец. вып. – С. 69–77.

15. Борде, Б. И. Цифроаналоговые преобразователи с выравненными весовыми разрядными коэффициентами / Б. И. Борде // Вопросы радиоэлектроники. Сер. Электронная вычислительная техника. – 1971. – Вып. 4. – С. 29–36.

16. Борде, Б. И. Пассивный сумматор сплошной проводящей структуры для цифроаналогового преобразователя / Б. И. Борде // Вопросы радиоэлектроники. Сер. Электронная вычислительная техника. – 1971. – Вып. 11. – С. 125–128.

17. Борде, Б. И. Структуры программируемых многофункциональных аналого-цифровых преобразователей / Б. И. Борде // Вопросы радиоэлектроники. Сер. Электронная вычислительная техника. – 1978. – Вып. 2. – С. 83–88.

18. Борде, Б. И. Цифроаналоговые преобразователи с активными генераторами разрядных токов / Б. И. Борде // Вопросы радиоэлектроники. Сер. Электронная вычислительная техника. – 1978. – Вып. 4. – С. 87–97.

19. Борде, Б. И. Транзисторные коммутационные элементы с цепями управления на туннельных диодах / Б. И. Борде // Автометрия. – 1966. – № 4. – С. 69–77.

20. Борде, Б. И. Динамическая индикация данных на сегментных матрицах / Б. И. Борде, Е. Г. Петрикеев // Приборы и системы управления. – 1976. – № 1. – С. 50–51.

21. Борде, Б. И. Оптоэлектронное управление МНОП-матрицами памяти / Б. И. Борде, В. Л. Кузнецов // Приборы и системы управления. – 1979. – № 9. – С. 32–33.

22. Борде, Б. И. Синтаксически управляемая трансляция задания на испытание в управляющую программу для микроЭВМ / Б. И. Борде, В. Л. Кузнецов // Программирование. – 1980. – № 4. – С. 61–63.

23. Автоматическая обработка результатов исследования распределения термоЭДС в полупроводниковых минералах / Б. И. Борде, А. С. Гурьевич, В. И. Красников, В. Г. Романов, В. Г. Черепанов // Автометрия. – 1977. – № 4. – С. 25–30.

24. Борде, Б. И. Система обработки данных физических исследований природных минералов на базе микроЭВМ / Б. И. Борде, В. Г. Черепанов // Управляющие системы и машины. – 1979. – № 3. – С. 93–98.

25. Борде, Б. И. Автоматизация сбора и обработки геолого-минералогических данных при разведке рудных месторождений / Б. И. Борде [и др.] // Геофизическая аппаратура 1989. – Вып. 90. – С. 98–104.

26. Борде, Б. И. Автоматизация обработки данных при испытании на растяжение / Б. И. Борде, Т. А. Куклина // Заводская лаборатория. – 1983. – № 8. – С. 88–89.

27. Борде, Б. И. Программно-методический комплекс «Основы САПР неоднородных вычислительных устройств и систем» / Б. И. Борде. – Красноярск: КГТУ, 1999. – CD-ROM (языки рус., англ.). Номер гос. регистрации НТЦ ИНФОРМРЕГИСТР 0329900090.

28. Борде, Б. И. Программно-методический комплекс «Основы САПР неоднородных вычислительных устройств и систем» / Б. И. Борде. – Красноярск: КГТУ, 2001. – CD-ROM (языки рус., англ.). Номер гос. регистрации НТЦ ИНФОРМРЕГИСТР 0320000143.

29. Борде, Б. И. Программно-методический комплекс «Основы САПР неоднородных вычислительных устройств и систем» / Б. И. Борде. – Красноярск: КГТУ, 2006. – CD-ROM (языки рус., англ.). Номер гос. регистрации НТЦ ИНФОРМРЕГИСТР 0320702238.

30. Борде, Б. И. Основы САПР вычислительных устройств / Б. И. Борде. – Красноярск: КрПИ, 1987. – 63 с.

31. Борде, Б. И. Основы САПР вычислительных устройств и систем / Б. И. Борде. – Красноярск : КГУ, 1989. – 176 с.

32. Борде, Б. И. Основы САПР неоднородных вычислительных устройств и систем / Б. И. Борде. Красноярск : КГТУ, 1996. – 248 с.

33. Борде, Б. И. Основы САПР неоднородных вычислительных устройств и систем : учеб. пособие / Б. И. Борде. – 2-е изд., перераб. и доп. – Красноярск : ИПЦ КГТУ, 2001. – 350 с.

34. Борде, Б. И. Расчет оптимальных режимов и точности работы транзисторных переключателей для цифроаналоговых преобразователей / Б. И. Борде // Автоматический контроль и методы электрических измерений : тр. IV Всесоюз. конф. / РИО СО АН СССР. – Новосибирск, 1964. – Т. 1. – С. 119–127.

35. Борде, Б. И. Информационная производительность цифроаналоговых преобразователей с транзисторными переключателями / Б. И. Борде // Автоматический контроль и методы электрических измерений. – Новосибирск : Наука, 1965. – С. 130–135.

36. Борде, Б. И. Статистический анализ путей снижения погрешности работы цифроаналоговых преобразователей с транзисторными переключателями / Б. И. Борде // Системы автоматизации научных экспериментов : тр. Всесоюз. конф. ИАЭ СО АН СССР. – Новосибирск : Наука, 1970. – С. 358–361.

37. Борде, Б. И. Статистические характеристики погрешности работы цифроаналоговых преобразователей / Б. И. Борде // Автоматический контроль и методы электрических измерений : тр. VIII Всесоюз. конф. – Новосибирск : Наука, 1971. – С. 183–191.

38. Борде, Б. И. Анализ способов снижения погрешности работы транзисторных коммутаторов / Б. И. Борде // Системы автоматизации научных экспериментов : тр. Всесоюз. конф. – Новосибирск : Наука, 1971. – С. 154–159.

39. Борде, Б. И. Проектирование многоуровневой системы моделирования неоднородных вычислительных систем по Р-технологии / Б. И. Борде // Р-технология программирования и средства ее инструментальной поддержки. – Киев : ИК АН УССР, 1982. – С. 61–63.

40. Борде, Б. И. Проектирование многоуровневой системы выбора структуры и моделирования неоднородных вычислительных систем / Б. И. Борде // Р-технология программирования. – Киев : ИК АН УССР, 1983. – Ч. 2. – С. 24–26.

41. Борде, Б. И. Комплексные и специализированные САПР в инженерном образовании / Б. И. Борде // Высшая школа на пути реформ : сб. Всерос. конф. – Красноярск : Изд-во «Кларетианум», 1998. – С. 171–172.

42. Борде, Б. И. Развитие структур аналого-цифровых подсистем автоматизации испытаний / Б. И. Борде // Автоматизация инженерных исследований и эксперимента / под ред. И. М. Витенберга. – М. : МД НТП, 1978. – С. 56–61.

43. Борде, Б. И. Оценка эффективности распределенных аналого-цифровых подсистем / Б. И. Борде // Повышение эффективности и качества продукции на базе развития распределенных систем управления и применения микропроцессорных устройств. – М., 1978. – С. 80–81.

44. Борде, Б. И. Оптимизация распределения подсистем преобразования и обработки сигналов / Б. И. Борде // Автоматизация научных исследований. – Красноярск : СО АН СССР ИФ, 1980. – С. 56–63.

45. Борде, Б. И. Развитие вычислительно-коммуникационных систем для инженерного образования / Б. И. Борде // Повышение качества непрерывного профессионального образования : материалы Всерос. науч.-метод. конф. – Красноярск : КГТУ, 2005. – С. 243–245.

46. Борде, Б. И. Развитие сетевых образовательных услуг для многоуровневого инженерного образования / Б. И. Борде // Совершенствование системы управления качеством подготовки специалистов : материалы Всеросс. науч.-метод. конф. с междунар. участием. – Красноярск : КГТУ, 2004. – С. 166–167.

47. Борде, Б. И. Автоматизация сбора и обработки геолого-минералогических данных при разведке рудных месторождений / Б. И. Борде [и др.] // Международный геологический конгресс : тез. докл. – М. : Наука, 1984. – С. 361–363.

48. Реализация веб-сервисов САПР электронной аппаратуры / Б. И. Борде, Е. В. Хабаров, С. И. Осит, Д. А. Подкаменный // Проблемы информатизации региона (ПИР-2003). – Красноярск : КГТУ, 2003. – С. 118–119.

49. Borde, B. I. Education– Research CAD of analog-digital computer units and system / B. I. Borde // Proceedings of First International Conference

on Distance learning and new tehnologies in education (ICDED94). – 1994. – P. 277–278.

50. Borde, B. I. Modelling and Presentation local Network with mobile users / B. I. Borde // 14 IST SUMMIT. – Dresden – 2005.

51. А.с. № 902242. Цифроаналоговый преобразователь / Б. И. Борде // ИПОТЗ № 4. – 1982.

52. Борде, Б. И. Модель релейной системы регулирования инерционным объектом / Б. И. Борде. – Красноярск : Красноярский политехнический институт, 1969. – 15 с.

53. А.с. 367541. Устройство гальванического разделения аналоговых сигналов / Б. И. Борде // ИПОТЗ № 8. – 1973.

54. Borde, B. Research and Industrial CAD in engineering education / B. Borde // International conference of Engineering Education ICEE95. – M., 1995. – P. 240.

55. Борде, Б. И. Автоматизация анализа и представления множества результатов неоднородных вычислительных систем / Б. И. Борде // Сб. докл. конф. CAD/CAM/PDM-2007. – М. : ИПУ РАН, 2007. – С. 42–45.

56. Борде, Б. И. Автоматизация формирования образов объектов вычислительных систем. / Б. И. Борде // Модели и методы обработки изображений. ММОИ-2007 : Материалы Всерос. науч. конф. – Красноярск, 2007. – С.112–117.

57. Борде, Б. И. Программно методический комплекс многовариантного проектирования неоднородных вычислительных систем / Б. И. Борде // Сб. докл. конф. CAD/CAM/PDM-2008. – М. : ИПУ РАН, 2008. – С. 23–26.

58. Бухтеев, А. Методы и средства проектирования систем на кристалле / А. Бухтеев // CHIP NEWS. – 2003. – № 4. – С. 4–14.

59. Венгер, О. В. Изменение маршрута проектирования БИС при переходе к нанотехнологиям / О. В. Венгер, А. В. Жмурин, Д. А. Рыбин // Информационные технологии. – 2005. – № 5. – Прил. – С. 2–4.

60. Вермишев, Ю. Х. Методы автоматического поиска решений при проектировании сложных технических систем / Ю. Х. Вермишев. – М. : Радио и связь, 1982. – 152 с.

61. Вермишев, Ю. Х. Основы автоматизации проектирования / Ю. Х. Вермишев. – М. : Радио и связь, 1988. – 280 с.

62. Вермишев, Ю. Х. Управление разработкой сложного объекта / Ю. Х. Вермишев // Информационные технологии в проектировании и производстве. – 2004. – № 2. – С. 4–12.

63. Вермишев, Ю. Х. Концепция синтеза технических решений / Ю. Х. Вермишев // Информационные технологии в проектировании и производстве. – 2007. – № 1. – С. 49–55.

64. Гагарин, А. П. О необходимости формирования национальной базы системного проектирования инфокоммуникационных систем и технологий / А. П. Гагарин, Е. И. Шульгин // Информационные технологии в проектировании и производстве. – 2007. – № 1. – С. 3–9.

65. Гитис, Э. И. Аналого-цифровые преобразователи / Э. И. Гитис, Е. А. Пискулов. – М. : Энергоиздат, 1981. – 360 с.

66. Глушков, В. М. Автоматизация проектирования вычислительных машин / В. М. Глушков, Ю. В. Капитонова, А. А. Летичевский. – Киев : Наук. думка, 1975. – 231 с.

67. Горбань, А. Н. Нейроинформатика / А. Н. Горбань [и др.]. – Новосибирск : Наука, 1998. – 296 с.

68. Грибов, В. В. Развитие онтологического подхода для автоматизации разработки пользовательских интерфейсов с динамаческими данными / В. В. Грибов // Информационные технологии. – 2010. – № 10. – С. 54–58.

69. Гуревич, И. М. Законы информатики – основа исследований и проектирования сложных систем / И. М. Гуревич // Информационные технологии. – 2003. – Прил. № 11. – 24 с.

70. Гусаков, А. А. Системотехника строительства / А. А. Гусаков. – М. : Стройиздат, 1993. – 368 с.

71. Добронец, Б. С. Интервальная математика : учеб. пособие / Б. С. Добронец. – Красноярск : КГУ, 2004. – 216 с.

72. Дружинин, В. В. Проблемы системологии / В. В. Дружинин, Д. С. Конторов. – М. : Радио и связь, 1976. – 276 с.

73. Дружинин, В. В. Системотехника / В. В. Дружинин, Д. С. Конторов. – М. : Радио и связь, 1985. – 200 с.

74. Евгенев, Г. Б. Системология инженерных знаний / Г. Б. Евгенев. – М. : Изд-во МГТУ им. Н. Э. Баумана, 2001. – 376 с.

75. Евреинов, Э. В. Однородные вычислительные системы, структуры и среды / Э. В. Евреинов. – М. : Радио и связь, 1981. – 208 с.

76. Евреинов, Э. В. Однородные вычислительные системы / Э. В. Евреинов, В. Г. Хорошевский. – Новосибирск : Наука, 1978. – 319 с.

77. Елшин, Ю. М. ГРИФ-4, Информационно-программный комплекс расширения функционала САПР P-CAD 200X. – М. : ПАО НПО «АЛМАЗ», 2017. – 496 с.

78. Елшин, Ю. М. Автоматизированные рабочие места при проектировании РЭА / Ю. М. Елшин. – М. : Радио и связь, 1983. – 128 с.

79. Елшин, Ю. М. Электронная промышленность: шаг к CALS-технологиям – стандарт ODB++ / Ю. М. Елшин. – М. : Родник Софт, EDA Express № 7. – 2003. – С. 16–24.

80. Зарипов, Р. Х. Машинный поиск вариантов при моделировании творческого процесса / Р. Х. Зарипов. – М. : Наука, 1983. – 232 с.

81. Инженерная поддержка жизненного цикла электронных средств : монография / В. В. Гольдин, В. Г. Журавский, А. В. Сарафанов, Ю. Н. Кофанов. – М. : Радио и связь, 2002. – 379 с.

82. Карман, Т. Математические методы в инженерном деле / Т. Карман, М. Био, пер. с англ. М. Г. Шестопал. – М. : ОГИЗ, 1948. – 424 с.

83. Каляев, А. В. Многопроцессорные системы с программируемой архитектурой / А. В. Каляев. – М. : Радио и связь, 1984. – 240 с.

84. Коломбет, Е. А. Микроэлектронные средства обработки аналоговых сигналов / Е. А. Коломбет. – М. : Радио и связь, 1991. – 376 с.

85. Коробков, Б. П. Модели структурной адаптации в процессах управления сложными объектами / Б. П. Коробков, Л. А. Растригин. – Рига : Зинатне, 1977. – С. 3–21.

86. Королев, Л. Н. Структуры ЭВМ и их математическое обеспечение / Л. Н. Королев. – М. : Наука, 1974. – 256 с.

87. Корячко, В. П. Теоретические основы САПР / В. П. Корячко, В. М. Курейчик, И. П. Норенков. – М. : Энергоиздат, 1987. – 400 с.

88. Кузин, Е. С. Представление знаний и решение информационно-сложных задач в компьютерных системах / Е. С. Кузин // Информационные технологии. Прил. – 2004. – № 4. – 32 с.

89. Лазарев, И. А. Композиционное проектирование сложных агрегативных систем / И. А. Лазарев. – М. : Радио и связь, 1986. – 312 с.

90. Лапко, А. В. Непараметрические системы обработки информации / А. В. Лапко, С. В. Ченцов. – М. : Наука, 2000. – 350 с.

91. Лебедев, С. А. Быстродействующие универсальные вычислительные машины / С. А. Лебедев // Тр. конф. «Пути развития советского математического машиностроения и приборостроения». – М. : ВИНИТИ, 1956. – С. 31–43.

92. Майоров, С. А. Структура электронных вычислительных машин / С. А. Майоров, Г. Н. Новиков. – Л. : Машиностроение, 1979. – 384 с.

93. Месарович, М. Общая теория систем: математические основы / М. Месарович, Я. Такахара. – М. : Мир, 1978. – 311 с.

94. Методические рекомендации по использованию электрических свойств рудных минералов для изучения и оценки эндогенных месторождений. – Л., 1983. – 90 с.

95. Моисеев, Н. Н. Математические задачи системного анализа / Н. Н. Моисеев. – М. : Наука, 1981. – 488 с.

96. Методы оптимизации / Н. Н. Моисеев [и др.]. – М. : Наука, 1978. – 352 с.

97. Нариньяни, А. С. Введение в неопределенность / А. С. Нариньяни // Информационные технологии. Прил. – 2007. – № 4. –32 с.

98. Нетребенко, К. А. Цифровые делители напряжения / К. А. Нетребенко. – М. : Энергия, 1970. – 224 с.

99. Нефедов, А. В. Интегральные микросхемы и их зарубежные аналоги : справ. в 11 т. Т. 5 / А. В. Нефедов. – М. : ИП Радио Софт, 2000. – 608 с.

100. Нейман, Дж. Теория самовоспроизводящихся автоматов / Дж. Нейман. – М. : МИР, 1971. – 382 с.

101. Норенков, И. П. Системы автоматизированного проектирования электронной вычислительной аппаратуры / И. П. Норенков, В. Б. Маничев. – М. : Высшая школа, 1983. – 272 с.

102. Норенков, И. П. Разработка систем автоматизированного проектирования : учеб. для вузов / И. П. Норенков. – М. : МГТУ им. Н. Э. Баумана, 1994. – 207 с.

103. Норенков, И. П. Основы автоматизированного проектирования : учеб. для вузов / И. П. Норенков. – М. : МГТУ им. Н. Э. Баумана, 2000. – 360 с.

104. Норенков, И. П. Информационная поддержка наукоемких изделий / И. П. Норенков, П. К. Кузьмик. – М. : МГТУ им. Н. Э. Баумана, 2002. – 320 с.

105. Норенков, И. П. Информатика в техническом университете / И. П. Норенков, А. М. Зимин. – М. : МГТУ им. Н. Э. Баумана, 2004. – 352 с.

106. Норенков, И. П. Генетические алгоритмы поиска решений в онтологических базах знаний / И. П. Норенков // Информационные технологии. – 2010. – № 9. – С. 20–24.

107. Норенков, И. П. Телекоммуникационные технологии и сети / И. П. Норенков, В. А. Трудоношин. – М. : МГТУ им. Н. Э. Баумана, 2000. – 248 с.

108. Овчинников, В. А. Алгоритмизация комбинаторно-оптимизационных задач при проектировании ЭВМ и систем / В. А. Овчинников. – М. : МГТУ, 2001. – 288 с.

109. Основы теории вычислительных систем / под ред. С. А. Майорова. – М. : Высшая школа, 1978. – 408 с.

110. Основы автоматизированного проектирования : учебник / под ред. А. П. Карпенко. – М. : ИНФРА М, 2015. – 329 с.

111. Петров, Г. М. Тенденции развития структур аналого-цифровых вычислительных систем / Г. М. Петров, Ю. А. Шубин // Вопросы радиоэлектроники. Сер. Электронная вычислительная техника. – 1974. – Вып. 2. – С. 111–128.

112. Половинкин, А. И. Основы инженерного творчества / А. И. Половинкин. – М. : Машиностроение, 1988. – 368 с.

113. Поляков, А. К. Моделирование цифровой аппаратуры на базе языка ПЛ/1 / А. К. Поляков. – М. : МЭИ, 1978. – 68 с.

114. Поляков, А. К. Моделирование ЭВМ на ЭВМ / А. К. Поляков. – М. : МЭИ, 1981. – 105 с.

115. Поляков, А. К. Языки VHDL и VERILOG в проектировании цифровой аппаратуры / А. К. Поляков. – М. : СОЛОН-Пресс, 2003. – 230 с.

116. Потапов Ю. Обзор САПР печатных плат / Ю. Потапов // CHIP NEWS. – 2003. – № 4. – С. 36–39.

117. Потемкин, И. С. Автоматизация синтеза функциональных схем / И. С. Потемкин. – М. : Энергоиздат, 1981. – 88 с.

118. Преобразователи информации в аналого-цифровых вычислительных устройствах и системах / под ред. Г. М. Петрова. – М. : Машиностроение, 1973. – 369 с.

119. Программирование, отладка и решение задач на ЭВМ единой серии. Язык ПЛ/1 / под ред. И. А. Кудряшова. – Л. : Энергоатомиздат, 1989. – 280 с.

120. Разевиг, В. Д. Система P-CAD 8.5–8.7: Руководство пользователя / В. Д. Разевиг. – М. : Солон-Р, 1999. – 720 с.

121. Разевиг, В. Д. Система проектирования печатных плат ACCEL EDA 15 (P-CAD 2000) / В. Д. Разевиг. – М. : Солон-Р, 2000. – 416 с.

122. Разевиг, В. Д. Система проектирования цифровых устройств ORCAD / В. Д. Разевиг. – М. : Солон-Р, 2000. – 160 с.

123. Мелехин, В. Ф. Вычислительные машины, системы и сети / В. Ф. Мелехин, Е. Г. Павловский. – М. : Академия, 2006.

124. Рубан, А. И. Методы оптимизации : учеб. пособие / А. И. Рубан. – Красноярск : НИИ ИПУ, 2001. – 528 с.

125. Рябов, Г. Г. Выбор функциональных схем узлов ЭВМ для интегрального исполнения / Г. Г. Рябов, В. Г. Соколов, В. С. Чунаев. – М. : ИТМиВТ, 1969. – 83 с.

126. Рябов, Г. Г. Интеллектуальные объекты – концепция от компьютерных технологий / Г. Г. Рябов, В. В. Суворов // Информационные технологии. – 2004. – № 6. – С. 9–16.

127. Рябов, Г. Г. Комплексное фундаментальное исследование интеллекта: путь к созданию компьютерных технологий новых поколений / Г. Г. Рябов, В. В. Суворов // Вычислительные методы и программирование. – 2004. – № 2. – С. 206–209.

128. Системы автоматизированного проектирования в радиоэлектронике : справ. / под ред. И. П. Норенкова. – М. : Радио и связь, 1986. – 368 с.

129. Сабунин, А. Г. Altium Designer / А. Г. Сабунин // Новые решения в проектировании электронных устройств. – М. : Солон-Пресс, 2010. – 432 с.

130. Советов, Б. Я. Информационная технология / Б. Я. Советов. – М. : Высш. шк., 1994. – 368 с.

131. Солодовников, В. В. Принцип сложности в теории управления / В. В. Солодовников, В. Ф. Бирюков, В. И. Тумаркин. – М. : Наука, 1977. – 344 с.

132. Стемпковский, А. Л. Система разработки макромоделей аналоговых и цифроаналоговых узлов для САПР БИС / А. Л. Стемпковский, Ю. Б. Егоров, А. А. Лялинский // Информационные технологии. – 2000. – № 2. – С. 31–33.

133. Судов, Е. В. Интегрированная информационная поддержка жизненного цикла машиностроительной продукции / Е. В. Судов. – М. : МВМ, 2003. – 264 с.

134. Сучков, Д. И. Адаптация САПР PCAD к отечественному технологическому оборудованию / Д. И. Сучков. – Обнинск : Призма, 1993. – 460 с.

135. Трудоношин, В. А. Моделирование электромеханических систем с помощью программно-методического комплекса ПА9 / В. А. Трудоношин, И. В. Трудоношин // Информационные технологии. – 2006. – № 4. – С. 10–13.

136. Создание VRML-миров : пер. с англ. / Э. Титтел, К. Сандерс, С. Чарли, П. Вольф. – М. : Изд. группа BHV, 1997. – 320 с.

137. Топорков, В. В. Выбор целевой архитектуры и стратегий распределения ресурсов вычислительных систем / В. В. Топорков // Информационные технологии. Прил. 9. – 2004. – 32 с.

138. Топорков, В. В. Автоматизация совместного проектирования программно-аппаратных систем: поведенческий подход / В. В. Топорков // Материалы II Междунар. конф. CAD/CAM/PDM-2002. – Т. 1. – М. : ИПУ РАН, 2002. – С. 26–37.

139. Учебно-исследовательская система автоматизированного проектирования радиоэлектронных схем : учеб. пособие / под ред. В. И. Анисимова. – Л. : ЛГУ, 1989. – 256 с.

140. Ушаков, В. Б. Моделирующие установки и пути их развития / В. Б. Ушаков // Пути развития советского математического машиностроения и приборостроения : тр. конф. – М. : ВИНИТИ, 1956. – С. 82–132.

141. Ушаков, В. Б. Структурные и функциональные возможности проблемно-ориентированных аналого-цифровых вычислительных систем третьего поколения / В. Б. Ушаков, И. М. Витенберг, Г. М. Петров // ПИСУ. – 1979. – № 5. – С. 5–8.

142. Файзулаев, Б. Н. Предельное быстродействие и основные закономерности развития логических БИС ЭВМ / Б. Н. Файзулаев // Микроэлектроника и полупроводниковые приборы. – 1984. – Вып. 8. – С. 5–15.

143. Федоров, И. Б. Организация подготовки специалистов в области компьютерных науки, техники и технологий, отраженная в российских

и зарубежных образовательных документах / И. Б. Федоров, И. П. Норенков, С. В. Коршунов // Информационные технологии. – 2006. – № 9. – С. 73–77.

144. Флейшман, Б. С. Основы системологии / Б. С. Флейшман. – М. : Радио и связь, 1982. – 368 с.

145. Фролов, А. В. Операционная система IBM OS/2 Warp / А. В. Фролов, Г. В. Фролов. – М. : Диалог-Мифи, 1996. – 272 с. (Библиотека системного программиста. Т. 20).

146. Фу, К. Структурные методы в распознавании образов / К. Фу. – М. : Мир, 1977. – 318 с.

147. Холстед, М. Х. Начала науки о программах / М. Х. Холстед. – М. : Финансы и статистика, 1981. – 128 с.

148. Цапенко, М. П. Измерительные информационные системы / М. П. Цапенко. – М. : Энергия, 1974. – 319 с.

149. Цирлин, А. М. Методы оптимизации в необратимой термодинамике и микроэкономике / А. М. Цирлин. – М. : Физматлит, 2003. – 416 с.

150. Цибульский, Г. М. Мультиагентный подход к анализу изображений / Г. М. Цибульский. – Новосибирск : Наука, 2005. – 188 с.

151. Цыпкин, Я. З. Основы теории обучающихся систем / Я. З. Цыпкин. – М. : Наука, 1970. – 252 с.

152. Чекмарев, А. Средства визуального проектирования на JAVA / А. Чекмарев. – СПб. : BHV, 1998. – 400 с.

153. Шапошников, И. В. Справочник Web-мастера. XML / И. В. Шапошников. – СПб. : BHV, 2001. – 304 с.

154. Шереметов, Л. Б. Виртуальные образовательные среды / Л. Б. Шереметов, В. Л. Усков // Информационные технологии. Прил. – 2002. – № 5. – 24 с.

155. Шокин, Ю. И. Интервальные вычисления / Ю. И. Шокин. – Новосибирск : Наука, 1981. – 112 с.

156. Шрейдер, Ю. А. Системы и модели / Ю. А. Шрейдер, А. А. Шаров. – М. : Радио и связь, 1982. – 152 с.

157. Штагер, В. В. Цифровые системы связи. Теория, расчет и оптимизация / В. В. Штагер. – М. : Радио и связь, 1993. – 312 с.

158. Эшби, У. Р. Введение в кибернетику / У. Р. Эшби. – М. : ИИЛ, 1959.

159. Янбых, Г. Ф. Методы анализа и синтеза сетей ЭВМ / Г. Ф. Янбых, Б. Я. Эттингер. – М. : Энергия, 1980. – 96 с.

160. CADdy: Опыт использования технологий. САПР и графика. Спец. вып. – М. : Компьютер Пресс, 2000. – 112 с.

Использованные интернет-ресурсы

161. www.autodesk.com
162. www.fsf.org
163. www.cypress.com
164. www.docinfo.ru
165. www.eclipse.org
166. www.ibm.com
167. www.elibrary.ru
168. www.intel.com
169. www.intergraph.com
170. www.inventionmachine.com
171. www.iso.org
172. www.java-source.net
173. www.mentor.com
174. www.microsoft.com
175. cad.onshape.com
176. www.mozilla.com
177. www.nanocad.ru
178. www.ni.com
179. www.web3d.org
180. bim-association.ru
181. www.oracle.com
182. www.parallelgraphics.com
183. www.protege.stanford.edu
184.www.tadviser.ru/index.php
185. www.vmware.com
186. www.w3c.org
187. https://www.lektorium.tv/mooc2/26296
188. rustandards.ru/docs/themes/13СП.2016. Системы автоматизации.
189. docs.cntd.ru/document/1200146763. Разработка СП «Информационное моделирование в строительстве. Правила применения в проектах повторного использования и при их привязке».
190. allgosts.ru/35/240/gost_r_57310-2016. Разработка СП «Информационное моделирование в строительстве. Правила разработки планов проектов, реализуемых с применением технологии информационного моделирования».
191. Профессиональный стандарт для подготовки специалистов по созданию и модификации интеграционных решений, код 06.041 Минтруда РФ в 2017 году.

ПРИЛОЖЕНИЯ

Приложение 1

Краткий словарь терминов

Анализ – проектная процедура получения информации о свойствах или поведении проектируемых объектов.

Данные (Data) – информация в виде, пригодном для автоматической обработки.

Данные базовые – сущности, свойства и отношения.

Интегрированные ресурсы – множество компонентов для прикладных протоколов.

Интерфейс (Interface) – совокупность средств и правил, обеспечивающих взаимодействие частей вычислительной системы или программного комплекса.

Интерфейс командный (Interface Command) – автоматическое преобразование высокоуровневого описания объекта проектирования в последовательность команд конкретной САПР низкого уровня.

Интерфейс табличный (Interface Table) – автоматическое преобразование высокоуровневого описания объекта во внутреннее представление объекта конкретной САПР низкого уровня.

Критерий эффективности – мера эффективности или правила предпочтения эффективности варианта.

Маршрут проектирования – последовательность проектных операций и процедур, реализующих этап проектирования множества объектов.

Модель функциональная – математическая модель, отражающая физическое или информационное состояние проектируемого объекта и процесс изменения состояний. При отражении в модели внутренних состояний отдельных элементов объекта ее называют **микромоделью,** в противном случае – **макромоделью**.

Противоречие физическое – несоответствие носителей информации или каналов.

Противоречие техническое – улучшение одного требования к объекту приводит к ухудшению другого.

Проектная операция – часть проектной процедуры.

Проектная процедура – часть процесса проектирования, завершающаяся получением проектного решения и состоящая из множества проектных операций.

САПР (CAD – Computer Aided Design) – система автоматизированного проектирования.

САПР высокого уровня – служит для автоматизации в основном алгоритмических и функционально-логических этапов проектирования, с использованием языков высокого уровня.

САПР низкого уровня – служит для автоматизации конечных этапов процесса проектирования конкретных объектов (конструкторских и частично функционально-логических).

Синтез – проектная процедура получения описания проектируемого объекта на каком-либо языке.

Структура объекта – компонентный состав объекта и отношения между компонентами.

Этап проектирования – часть процесса проектирования, завершающаяся получением проектного решения.

AAM (Application Activity Model) – функциональная модель.

AIM (Application Interpreted Model) – модель, представленная в стандарте STEP (ISO 10303) на языке Express.

APIO (Application Programming Interface Overview) – интерфейс прикладных программ.

AP (Application Protocol) – прикладной протокол в STEP, информационная модель объекта, состоящая из сущностей и атрибутов, на языке Express.

ARM (Application Requirements Model) – модель данных, использующая представление и терминологию приложения.

BIM (Building Information Model) – информационная модель здания или объекта. ISO 29481-1:2016. ISO 29481-1:2010. ГОСТ Р 57310–2016 (ИСО 29481-1:2010) Моделирование информационное в строительстве. Руководство по доставке информации. Методология и формат.

CALS (Continued Acquisition of Lifecycle Support) – формы и способы представления данных об объектах на всех этапах жизненного цикла.

EDIF (Electronic Design Interchange Format) – формат обмена данными при проектировании электронной аппаратуры.

EXPRESS – язык моделирования, имеющий две формы представления – текстовую, собственно EXPRESS и графическую EXPRESS-G.

LPEX (Live Parsing EXtensible editor) – многофункциональный расширенный редактор, поддерживающий множество языков программирования и описания технических решений. Редактор входит в состав инструментальных средств Visual Age (VA) фирмы IBM.

ONTOLOGY – формализованная спецификация концептуализации.

Операция абстракции (L2) – сохранение подмножества свойств объектов.

Операция конкретизации (L2) – расширение множества свойств обьектов.

Концептуализация – описание множества понятий (концептов) предметной области, знаний о них и отношений между ними.

RDF – формат описания ресурсов на базе языка XML. RDFS это RDF схема. Стандарты W3C.

OWL – язык представления онтологий в сети. Стандарты W3C.

OWL-S – язык представления сервисов в сети. Позволяет обнаружить сервис, вызвать сервис, создать составной сервис, контролировать выполнение сервиса. Стандарты W3C.

Промышленные автоматизированные системы и интеграция. Интеграция данных жизненного цикла для перерабатывающих предприятий. Ч. 1. ГОСТ Р ИСО 15926-1–2008 (ISO 15926 1–11). Практическая реализация сетевого языка онтологий (OWL).

ST-EXPRESS – программные средства для создания и отображения моделей на языке EXPRESS; позволяют по EXPRESS-схеме создавать EXPRESS-G-диаграммы.

STEP (Standard for the Exchange of Product model data) – международный стандарт ГОСТ Р ИСО 10303 (ISO 10303) для описания данных моделей промышленных изделий и объектов проектирования на всех этапах жизненного цикла.

SOC (System on Chip) – вычислительная система на кристалле.

SOAP (Single Object Accesses Protocol) – протокол доступа к объектам.

SCA (Service Component Architecture) – сервисно-компонентная архитектура.

UML (Unified Modeling Language) – унифицированный язык моделирования.

Visual Age (VA) фирмы IBM – инструментальный комплекс для создания программного обеспечения на различных языках программирования (VA C++, VA PLI, VA JAVA).

VRML (Virtual Reality Modeling Language) – язык описания модели среды виртуальной реальности ISO/IEC 14772-1.

XML (Extensible Markup Language) – расширенный язык разметки.

X3D (Extensible 3D) – расширяемый язык трехмерной графики **ISO/IEC 19775**.

WebGL – формат 3D-графики, поддерживаемый сетевыми программами просмотра без дополнительных программных модулей.

Приложение 2

Студенты группы 130-2 Подлесный Сергей и 130-1 Шибаев Владимир исследуют первый макет ЭВМ на транзисторах в 1962–1963 учебном году. В 1965 году под руководством Б. И. Борде защитили дипломные проекты в области вычислительной техники первые выпускники радиотехнического факультета – С. А. Подлесный, А. В. Сысоев.

Сибирский федеральный университет
Политехнический институт — год 1956

КПИ

Из постановления Совета министров СССР от 4 августа 1956 г. №1043:

Совет министров постановляет принять предложения министерства высшего образования СССР об организации в 1956 г. в г. Красноярске политехнического института (КПИ).

Из приказа Государственного Комитета Российской Федерации по высшему образованию от 21 июня 1993 г. №41:

Переименовать Красноярский политехнический институт в Красноярский государственный технический университет.

Приказ Министерства образования и науки Российской Федерации № 1417 от 28.11.2006

На основании Положения о Федеральном агентстве по образованию, утвержденного постановлением Правительства Российской Федерации от 17 июня 2004 г. № 288, и во исполнение распоряжения Правительства Российской Федерации от 4 ноября 2006 г. № 1518-р приказываю:

Считать федеральное государственное образовательное учреждение высшего профессионального образования "Сибирский федеральный университет" правопреемником реорганизуемых государственных образовательных учреждений высшего профессионального образования "Красноярский государственный университет", "Красноярский государственный технический университет", "Красноярской государственной архитектурно-строительной академии", "Государственный университет цветных металлов и золота".

Корпус А по улице Ленина, 70 – первое здание Красноярского политехнического института (КПИ). В корпусе А размещаются лаборатории электро-технического факультета

Афонский Иван Филиппович, профессор, д.т.н. проректор по научной работе. В 1959 году он первым в КПИ защитил диссертацию на соискание ученой степени доктора технических наук и был первым проректором института по научной работе с 1957 по 1963 год.

Втюрин Николай Иванович, к.ф.-м.н.(1952), доцент (1954) В 1945 году окончил Красноярский государственный педагогический институт. С 1957 года заведовал кафедрой физики. С 1965 по 1973 годы занимал должность проректора по научной работе.

Груш Давид Бенционович, доцент, кандидат исторических наук, заведующий кафедрой Марксизма-Ленинизма. Окончил Киевский педагогический институт в 1941 г.

Жилин Анатолий Николаевич, доцент, к.т.н. Заведующий кафедрой общей и теоретической электротехники КПИ. С 1951 по 1973 заведующий кафедрой,1973–1988– доцент кафедры электротехники Сибирского технологического института.

Шестаков Геннадий Георгиевич, зав. кафедрой Физического воспитания и спорта. Окончил Ленинградский институт физической культуры в 1995 г. и Красноярский медицинский институт в 1970 г. Проделал огромную работу по развитию спорта в КПИ.

Профессор Борисов Василий Николаевич, участник Великой Отечественной Войны, закончил Ленинградскую Лесотехническую Академию и защитил кандидатскую диссертацию. С 1950 года по 1956 год работал заведующим кафедрой, деканом и проректором по учебной работе в Сибирском технологическом институте. В 1956 году стал первым ректором Красноярского политехнического института и бессменно руководил им до 1983 года. Возглавляемый им Красноярский политехнический явился базой для становления в Красноярске двух академий: Архитектурно-строительной и Сибирской аэрокосмической, Хакасского технического института. Развитие материальной базы КПИ, современный уровень лабораторного оборудования, инновационные технологии обучения студентов способствовали переходу ВУЗа на качественно новую ступень – в 1994 году КПИ стал Красноярским государственным техническим университетом. Деятельность Василия Николаевича по подготовке научных и инженерных кадров высоко оценена правительством и он удостоен ряда правительственных наград.

Борисов В.Н. вступил в должность и.о. директора КПИ в соответствии с приказом МВО СССР №524-к от 13 августа 1956 и приказом №1по КПИ от 3 сентября 1956 года.

Корпус Б по проспекту Мира, 49 – второе здание КПИ в котором размещались руководство института, машиностроительный факультет, кафедра высшей математики и физики

Артамонов Валентин Васильевич, доцент, к.т.н., декан вечернего факультета, старший преподаватель кафедры высшей математики. Закончил Московский энергетический институт в 1950 г.

Воробьева Татьяна Ивановна, доцент. В связи с открытием КПИ была рекомендована на заведование кафедрой высшей математики и до 1965 года занимала эту должность. С 1965 по 1967 гг. – проректор по вечернему и заочному обучению, 1967–1985 гг. продолжала работу на кафедре высшей математики.

Гнетулло Сергей Михайлович. В период с 1942 по 1945 год участвовал в Великой Отечественной войне. В 1950 году окончил Красноярский государственный педагогический институт. Прошел путь от ассистента до доцента кафедры высшей математики.

Борде Бернгард Исаакович, ассистент (1956), к.т.н. (1967), доцент (1968), профессор (1990), член-корр. МАН ВШ (1997). Руководитель лаборатории «Неоднородных вычислительных систем», руководитель подразделения «Вычислительная техника» (1963 – 1968), зав. кафедрой «Вычислительная техника» (ВТ) (1969 – 1980).

Волков Валентин Ефремович, к.ф-м.н. (1966), доцент (1967) В 1953 году окончил Ленинградский государственный университет. Прошел путь от старшего преподавателя до заведующего кафедры химии. Опубликовал 42 научные работы по молекулярной спектроскопии.

Первые студенты механического факультета 1956 – 1961

1. Башмаков Лев Полиевктович
2. Павлов Анатолий Георгиевич
3. Логинов Лев Николаевич
4. Зубков Анатолий Иванович
5. Кузяев Борис Анисимович
6. Дурнев Николай Трофимович
7. Журавлев Владимир Николаевич
8. Ткаченко Владимир Феодосьевич
9. Медведев Борис Андреевич
10. Колосницына Нелли Михайловна
11. Хвастунов Александр Павлович
12. Августонович Олег Николаевич
13. Савельев Эдуард Иванович
14. Федорова Альбина Михайловна
15. Федоров Борис Иванович
16. Коган Ефим Иосифович
17. Чебодаев Виталий Михайлович
18. Маслов Владимир Вячеславович
19. Беспрозванных Владимир Емельянович
20. Долженко Любовь Павловна
21. Костомошвили Амиран Георгиевич
22. Яковлева Наталья Изотовна
23. Водопьянова Нина Алексеевна
24. Зотин Анатолий Степанович
25. Башмакова Альвина Иннокентьевна
26. Власова Маргарита Ивановна
27. Мартынов Эдуард Захарович
28. Марчук Елена Иосифовна
29. Татаренко Владимир Николаевич
30. Соловьев Александр Борисович
31. Непомящий Виктор Михайлович
32. Филимонова Галина Александровна
33. Воронова Эмма Николаевна

34. Негадора Владислав Владимирович
35. Кузнецов Николай Анатольевич
36. Меркулова Галина Викторовна
37. Шишов Сергей Александрович
38. Сербатинский Вячеслав Харитонович
39. Машков Владимир Игнатьевич
40. Шамсутдинова Альфия Ганиятулловна
41. Корьюс Ванда Юрьевна
42. Анохин Роберт Петрович
43. Андреев Анатолий Борисович
44. Горбачев Сергей Михайлович
45. Токарев Игорь Борисович
46. Дарданова Людмила Николаевна
47. Матушанский Геннадий Лазаревич
48. Розадаев Евгений Петрович
49. Климов Юрий Иванович
50. Кузякина Валентина Ивановна
51. Скобочкин Алевтин Константинович
52. Хейн Василий Сергеевич
53. Сидорова Светлана Яковлевна
54. Беззубенко Артур Трофимович
55. Григорьев Юрий Николаевич
56. Емельянчик Валерий Максимович

Василием Николаевичем Борисовым был организован профессиональный коллектив, в который, наряду с преподавателями высокой квалификации, посвятившими научным исследованиям многие годы своей жизни, вошли и молодые, полные творческой энергии, сотрудники. Здесь всегда царила творческая атмосфера, и высоко ценились результаты работы. Преподаватели и студенты учились, разрабатывали новые лабораторные работы, совмещая интеллектуальный труд с трудом физическим – закладывая фундамент главного корпуса.

Осенью 1956 были сформированы актуальные для региона научные направления и открыты современные на тот период лаборатории. Основные из них – материаловедение и технология для строительства и машиностроения; информационные технологии и телекоммуникации; энергетика.

Преподаватели и студенты мечтали, чтобы план студенческого городка стал реальностью. Копали котлован под фундаменты, строили здания, изготавливали лабораторные стенды, учились и проводили исследования. Их трудом построили и благоустроили студенческий городок КПИ – КГТУ.

Родичев Геннадий Михайлович Окончил Красноярский государственный педагогический институт в 1952 году. Прошел путь от ассистента до профессора кафедры физики. С 1970 по 1973 год заведовал кафедрой физики. Имеет 56 научных трудов и 6 изобретений.

Зубков Анатолий Иванович, студент первого набора, председатель профкома института, помогал ректору в организации быта студентов на период строительства КПИ. Проявил большую работу по созданию благоприятных условий для коллектива политехников в самое трудное время – становления ВУЗа.

Главный корпус (Г) Красноярского политехнического института (КПИ) по улице Киренского, 26 – основное место, где за учебой, работой, культурный деятельностью и спортом проводит свое время студенты, сотрудники и преподаватели университета. Построен в 1963 году.

Штатный формуляр профессорско-преподавательского состава на 1956/1957 учебный год по состоянию на 15 ноября 1956 года.

Кафедра высшей математики
Воробьева Татьяна Ивановна
Борде Бернгард Исаакович
Артамонов Валентин Васильевич
Гнетулло Сергей Михайлович

Кафедра иностранных языков
Овсянникова Галина Калининовна
Предтеченская Нина Николаевна
Захарченко Нина Васильевна

Кафедра марксизма - ленинизма
Бут Лидия Кирилловна
Тетерева Александра Яковлевна
Груш Давид Бенционович

Кафедра начертательной геометрии и черчения
Хейн Сергей Иннокентьевич
Васильев Юрий Николаевич
Ходоренко Мария Кондратьевна
Кузнецов Виталий Александрович
Поляков Владимир Константинович

Кафедра общей и технической электротехники
Жилин Анатолий Николаевич

Кафедра технической механики
Борисов Василий Николаевич
Томашевич Нина Аркадьевна
Вошакин Венедикт Федорович

Кафедра технологии металлов и материалов
Афонский Иван Филиппович
Алексовский Юрий Михайлович

Кафедра физвоспитания и спорта
Шестаков Геннадий Георгиевич
Медзяникова Тамара Иосифовна

Кафедра физики и химии
Втюрин Николай Иванович
Волков Валентин Ефремович
Чудновская Сима Наумовна
Родичев Геннадий Михайлович

Приложение 3

Таблица П3.1

Перечень работ, включенных в координационные планы ГКНО СССР (Минвуза), ГКНВШ РСФСР (Росминвуза) и отраслевых министерств, выполненных под руководством автора, и акты использования работ

Номер п/п	Номер документа	Наименование документа	Наименование отчета	Результаты работы
1	Проблема 080.250-Д1 утв. СМ СССР	Министерство цветной металлургии СССР Тематическая карта Минцветмета 18-69-032	Отчет по теме 59 «Устройство вычисления и аналого-цифрового преобразования приведенного напряжения и сопротивления электролизера», КПИ, 1971, 180 с.	П1, П2 Акты об испытании и эффекте внедрения на алюминиевых заводах отрасли в 1970–1980 гг.
2	Приказ Минвуза № 595 от 27.06.1971, п. 13	О развитии в вузах НИР в области оптимизации и автоматизации исследований	Разработка и исследование устройства кодирования видеосигналов «КОД4». Многоканальный цифро-аналоговый преобразователь с запоминающим устройством «КОД5»	Акты об испытании и эффекте от 19.12.1974, 13.12.1975. Акты об испытании и эффекте от 16.04.1976. КБ МРЗ г. Москва
3	Приказ Росминвуза № 394 от 17.09.1976	Автоматизация проектирования. П. 09.07.03. Разработка и исследование системы автоматизации обработки результатов испытаний	Разработка и исследование системы автоматизации обработки результатов испытаний. Отчет по НИР. КПИ, Красноярск, № ГР 78082237, Инв. № Б996696, М: ВНИТИЦЕНТР, 1981, 123 с. (92 с.)	Акт о внедрении НИР от 4.03.1981
4	Темплан МинГеологии СССР от 26.12.1977 XII-020 313-3	Исследование и разработка аналого-цифровой системы обработки данных физических исследований природных минералов «КОД8»	Отчет по НИР. Система обработки данных исследования природных минералов на базе микроЭВМ КПИ, Красноярск, №ГР 75055070, Инв. № Б625323, М.: ВНИТИЦЕНТР, 1977 г. 277 с. (60 с.)	Акт о внедрении НИР «КОД8» от 15.11.1982

Продолжение табл. П3.1

Номер п/п	Номер документа	Наименование документа	Наименование отчета	Результаты работы
5	Постановление СМ СССР по ВПК № 35 от 07.02.1978	Приказ Росминвуза 29сс от 23.03.1978	Аналого-цифровая подсистема сопряжения вычислительного комплекса с объектом. Отчет по НИР. КПИ, Красноярск, ГР81051088, Инв. № 0281.3014017, М.: ВНИТИЦЕНТР, 1981. 235 с. (30 с.)	Акт о внедрении НИР от 4.11.1981
6	Приказ Мингео № 409 от 09.09.1983	Приказ Росминвуза 131 дсп от 12.03.1979	Аналого-цифровая система обработки данных исследований природных минералов «КОД8М». Отчет по НИР. КПИ, № ГР01814004783, Инв. № 02830028481, М.: ВНИТИЦЕНТР, 1983,130 с. (23 с.)	Акт о внедрении НИР «КОД8М» от 15.11.1982
7	Приказ Мингео № 212 от 25.04.1988	О проведении приемочных испытаний автоматизированной системы «КОД10»	Аналого-цифровая терминальная система обработки данных исследования природных минералов «КОД10» Отчет по НИР. КПИ, Инв. № ГР0288000956260062 129, М.: ВНИТИЦЕНТР, 1988, 109 с. (50 с.)	Акт о внедрении НИР от 19.04.1989
8	Приказ Минвуза № 1211 от 29.12.1978 П. 2.1.1. Приказ Росминвуза 31 от 15.02.1983 П. 3.182	Разработка теоретических и алгоритмических основ автоматизации проектирования аналого-цифровых подсистем	НИР «Основы автоматизации проектирования аналого-цифровых вычислительных систем». Внедрено «Базовое программное обеспечение первой очереди УИ САПР»	Акт о внедрении НИР в учебный процесс КПИ от 1.04.1986
9	Приказ Минвуза № 195 от 16.03.1987	О развитии в 1986–1990 годах исследований по созданию УИ САПР и их подсистем в вузах	Пункт 3.2.17. Разработать подсистему автоматизации проектирования аналого-цифровых вычислительных систем.	УИ САПР, учебное пособие изданы в 1989 г., с грифом Минобразования РФ – в 1996 и 2001 гг.

Продолжение табл. П3.1

Номер п/п	Номер документа	Наименование документа	Наименование отчета	Результаты работы
10	Гранты Краевого фонда науки 2000, проект «Интеграция» 2001 г. А0020/2.1		Учебное пособие «Основы САПР неоднородных вычислительных систем» с грифом Минобразования, 2001 г.	ПМК на CD «Основы САПР неоднородных вычислительных систем» в 2001, 2006 гг.
11	Проект SITE Евросоюза	IST mobile Summit	Моделирование систем с мобильными компонентами	Дрезден 2005 г.
12	Приказ СФУ, проект 001/3 2007 г.	Инновационные образовательные программы (УМКД)	ПМК на CD и УМКД «Моделирование неоднородных вычислительных систем»	Программа курса, лекции, лабораторные и курсовые работы. Акты использования СФУ 38-1810.10 от 14.05.2008

Таблица П3.2

Основные события в развитии информатики и СФУ

События в КПИ КГТУ СФУ	Хранение и передача информации	События в развитии вычислительной техники	Этапы (годы)
Создание КПИ / КПИ переданы здания по пр. Мира, 49 и Ленина, 70. В осеннем семестре занятия проводились на дневном и вечернем факультетах. Б. И. Борде был принят в КПИ 15.09.1956	Конференция «Пути развития советского математического машиностроения и приборостроения»	НПО «Светлана» выпустило первые транзисторы. Серийный выпуск настольной АВМ МН7 с наборами диодов для непрерывной логики	1956
Создание лаборатории вычислительной техники (ВТ) в составе кафедры высшей математики. Ректор В. Н. Борисов и зав. каф. ВМ Т. И. Воробьева. В 1960 году Б. И. Борде сдал кандидатский экзамен акад. С. А. Лебедеву в ИТМиВТ РАН	Обучение основам вычислительной техники и программированию всех студентов КПИ. Доклад на конференции КПИ по комплексу компонент для ЭВМ, включая транзисторы в ключевых режимах. Повышение квалификации в МЭИ в 1959 г.	Создание в лаб. ВТ КПИ макета ЭВМ на транзисторах и приобретение нескольких машин МН7. Машины МН7 предназначались для решения нелинейных дифференциальных уравнений до 6-го порядка и были очень эффективны	1957–1960

Продолжение табл. П3.2

События в КПИ КГТУ СФУ	Хранение и передача информации	События в развитии вычислительной техники	Этапы (годы)
Размещение в 1963 году в главном корпусе КПИ 7 учебных лабораторий на 4 этаже в центральной части главного корпуса. В 1967 году в объединенном Совете СОАН Б. И. Борде защитил кандидатскую диссертацию	Приказом ректора КПИ В. Н. Борисова № 76 от 16.02.1963 года организована самостоятельная дисциплина «Вычислительная техника» на базе 7 лабораторий. Приказом Министра В. Столетова от 23.06.1969 г. в КПИ организована кафедра вычислительной техники в составе РТФ	Приобретение ЭВМ «МИР» для инженерных расчетов в 1969 г. Выпуск учебных пособий по АВМ МН7 и программированию на ЭВМ «МИР»	1961–1970
Освоение машин «МИР» шло легко, так как входной язык машин «МИР» был Алгол-60, переведенный на русский язык программирования АЛМИР	Проектирование и ввод в учебный процесс зала ЭВМ «МИР» для самостоятельной работы студентов КПИ в 1973 г. Проектирование, монтаж и ввод в эксплуатацию вычислительного центра КПИ на базе ЕСЭВМ в 1976 году со штатом в 16 единиц. Начальником ВЦ назначен В. П. Нуждин. С 1979 года начальником ВЦ КПИ назначен канд. техн. наук Г. М. Цибульский, избранный деканом ФИВТ и в 2006 году директором ИКИТ СФУ	Приобретение терминальных станций ЕС7906 в 1976 году и затем ЕС7920. Начало безбумажных технологий	1971–1980
Доступ к международному программному обеспечению для IBM 360-370 позволил КПИ обеспечить доступ к международным достижениям программных комплексов	Приобретение персональных ЭВМ (ПЭВМ). Доступ к сетевым сервисам с ПЭВМ или терминалов	Организация глобальной сетевой структуры с серверами	1981–1990
Для таких систем и появились САПР (системы автоматизированного проектирования) в различных отраслях	Программное обеспечение и САПР увеличивали производительность инженерного труда	Повсеместный доступ к глобальной сети. Начало беспроводных сетей ЭВМ	1991–2000

Окончание табл. П3.2

События в КПИ КГТУ СФУ	Хранение и передача информации	События в развитии вычислительной техники	Этапы (годы)
Обучать студентов САПР можно только на открытых системах. Поэтому появился приказ министра по учебно-исследовательским САПР № 195 от 16.03.1987. Пункт 3.2.17 утвердил УИ САПР НВС COD	Создание дата-центра СФУ и доступ к обучающему контенту из любой точки мира. Начало массовых онлайн-курсов. Обучение без границ	Массовый выпуск мобильных беспроводных устройств доступа к сети	2001–2010
Появление облачных САПР, доступ к которым обеспечен из любой точки сети, наример Autodesk InfraWorks	Создание уникальных дата-центров в России. Пример – дата-центр Сбербанка в Сколково, проект которого создавали в САПР REVIT. Централизация хранения и обработки данных. Завершить переход на новую технологию BIM планируется на 2024 г.	Создание центров и структур по цифровой экономике. Новые своды правил по BIM обязательны с 1 марта 2018 г.	2011–2024

Научное издание

Борде Бернгард Исаакович

МЕТОДЫ АВТОМАТИЗАЦИИ ПРОЕКТИРОВАНИЯ НЕОДНОРОДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ И ИНФОРМАЦИОННЫХ МОДЕЛЕЙ ОБЪЕКТОВ

Монография

Редактор *М. В. Саблина*
Корректор *З. В. Малькова*
Компьютерная верстка *Н. Г. Дербенёвой*

Подписано в печать 05.06.2020. Печать плоская. Формат 60×84/16
Бумага офсетная. Усл. печ. л. 13,25. Тираж 500 экз. Заказ № 7781

Библиотечно-издательский комплекс
Сибирского федерального университета
660041, Красноярск, пр. Свободный, 82а
Тел. (391) 206-26-16; http://bik.sfu-kras.ru
E-mail: publishing_house@sfu-kras.ru