Н.И. Фомичев

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ НАУЧНЫХ ИССЛЕДОВАНИЙ

Министерство образования Российской Федерации
Ярославский государственный университет им. П.Г. Демидова

Н.И. Фомичев

АВТОМАТИЗИРОВАННЫЕ СИСТЕМЫ НАУЧНЫХ ИССЛЕДОВАНИЙ

Ярославль 2001

ББК 72.4(2)я73
Ф76
УДК 536

Рецензенты:
кафедра физики Ярославского государственного технического университета;
М.Н. Преображенский, канд. физ.-мат. наук

Фомичев Н.И.

Ф76 Автоматизированные системы научных исследований: Учеб. пособие / Н.И. Фомичев; Яросл. гос. ун-т. - Ярославль, 2001. - 112 с.
ISBN 5-8397-0156-4

Приведены основные сведения по общим вопросам построения АСНИ. Дано описание составных частей, принципов построения и типовой конфигурации АСНИ. Показаны возможности современных ЭВМ и особенности их использования в АСНИ. Освещены вопросы структуры и методов доступа в локальные вычислительные сети. Дан обзор интерфейсов, используемых при построении АСНИ. Представлены рекомендации по применению программного обеспечения и его специфике для разработки автоматизированных систем научных исследований.

Пособие предназначено для студентов физического факультета, изучающих дисциплину «Информатика».

ББК 72.4(2)я73
Ф76
УДК 536

ISBN 5-8397-0156-4

Учебное издание

Фомичев Николай Иванович

Автоматизированные системы научных исследований

Редактор, корректор В.Н. Чулкова
Компьютерная верстка И.Н. Ивановой

Лицензия ЛР № 020319 от 30.12.96.
Подписано в печать 28.12.2001. Формат 60 x 84/16. Бумага тип.
Печать офсетная. Усл. печ. л. 6,51 Уч.-изд. л. 6,99. Тираж 100 экз. Заказ

Оригинал-макет подготовлен в редакционно-издательском отделе
Ярославского государственного университета
Отпечатано на ризографе
Ярославский государственный университет им. П. Г. Демидова
150000 Ярославль, ул. Советская, 14.

Введение

Автоматизированные системы научных исследований (АСНИ) прочно заняли свое место в современном научном эксперименте. Применение АСНИ значительно повышает эффективность исследований, производительность труда, сокращает сроки обработки информации по результатам экспериментов, открывает пути к получению качественно новых результатов.

Для повышения эффективности научных исследований необходимо решить две проблемы.

Первая - создание современной базы автоматизации и планирования научных исследований, основанной на широком внедрении ЭВМ, измерительно-информационных систем, стандартизации аппаратного и программного обеспечения.

Вторая - подготовка инженерных и научных кадров, способных создать, освоить и внедрить автоматизированные системы.

С решением второй проблемы и связано наличие в программе высшей школы дисциплин «Автоматизированные системы научных исследований» и «Автоматизация эксперимента».

Дисциплина «Автоматизированные системы научных исследований» является составной частью курса «Информатика» и включает в себя общие концепции построения АСНИ, организации работ по их созданию.

Цель данного пособия заключается в том, чтобы помочь студентам, не имеющим достаточного опыта автоматизации измерений, овладеть практическими знаниями, необходимыми для самостоятельной работы с АСНИ.

С появлением персональных ЭВМ (ПЭВМ) и современной аппаратуры стало возможным создание простых по структуре, но гибких и легко адаптируемых систем объектного уровня, способных выполнять полной цикл сбора и обработки опытных данных на лабораторных экспериментальных установках.

В пособии рассмотрены общие принципы построения АСНИ, вопросы выбора эффективных алгоритмов обработки результатов исследований и управления научным экспериментом, приемов их реализации с учетом возможностей технических средств и программного обеспечения различных подсистем АСНИ.

Научные исследования как объект автоматизации

Целью любого научного исследования является построение модели, наилучшим образом отражающей свойства реального объекта (процесса или явления). Создание модели осуществляется сопоставлением теории и эксперимента. Это сопоставление носит, как правило, итерационный характер, что можно отобразить в виде алгоритма (рис. 1). На каждом шаге итерации происходит уточнение модели, что ведет обычно к ее усложнению.

На теоретической стадии возникает необходимость применения все более сложного математического аппарата, широкого использования ЭВМ с высокой производительностью.

На экспериментальной стадии научного исследования усложнение модели объекта приводит к увеличению потоков информации и требует, начиная с некоторого момента, создания АСНИ.

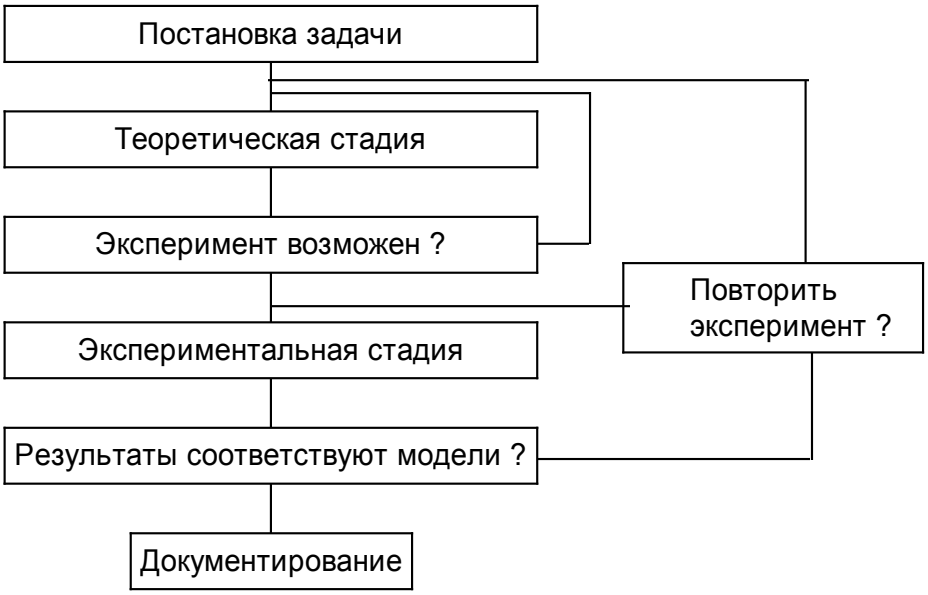


Рис. 1.

При создании АСНИ преследуется достижение ряда основных целей:

1. Повышение эффективности и качества научных исследований на основе получения и уточнения более полных моделей исследуемых объектов.

2. Получение качественно новых научных результатов, достижение которых невозможно без использования АСНИ.

3. Снижение сроков и трудоемкости научных исследований.

Автоматизированная система научно-технических исследований представляет собой аппаратно-программный комплекс на базе средств вы-

числительной техники, предназначенный для получения, уточнения и апробации математических моделей исследуемых объектов, явлений, процессов.

Отметим, что в определении представлены три существенных момента, характерных именно для АСНИ:

- ключевая роль средств вычислительной техники (практически никакая АСНИ не обходится без нее);
- единство аппаратных и программных средств;
- целевое назначение АСНИ, ее ориентация на получение математических моделей в виде формул, графиков, таблиц и т.д.

Особенности научных исследований как объекта автоматизации

Для того чтобы автоматизировать тот или иной объект, необходимо ясно представить его основные особенности. Для автоматизации научных исследований целесообразно выделить некоторые их *главные черты*. К ним относят следующие.

1. Многогранность исследовательской деятельности. Научные исследования включают в себя элементы разного характера: постановка научной задачи, разработка теории, проведение научных расчетов, моделирование, систематизация и поиск научной информации, разработка методологии эксперимента, проведение эксперимента, обработка, накопление и отображение информации, интерпретация результатов, принятие решений и т.д. Различные стороны научной деятельности в неодинаковой степени поддаются формализации и реализации в рамках АСНИ. Сравнительно просто автоматизируются процедуры, связанные с проведением эксперимента (регистрация, обработка, накопление, отображение информации и т.п.). Вместе с тем, такие стороны научной деятельности, как постановка задачи исследования, разработка теории, интерпретация результатов, требуют обязательного творческого участия человека-исследователя.

2. Существенная роль человеческого фактора. Человек остается главной, ключевой фигурой исследования и при наличии автоматизированной системы. С точки зрения разработчика АСНИ, это означает необходимость создания максимальных удобств пользователю при работе с системой. Как следствие, в современных АСНИ наблюдается широкое использование диалогового режима работы, средств графического представления информации.

3. Высокий уровень априорной неопределенности хода и результатов исследования. Научные исследования всегда проводятся для получения некоторой новой информации о свойствах объекта исследований. Разработчики АСНИ вынуждены работать при дефиците априорной информации.

Это одно из принципиальных отличий АСНИ от автоматизированных систем других классов (АСУТП, АСУП, САПР). Данная особенность требует таких технических решений при создании АСНИ, которые позволяют сделать систему максимально гибкой, легко модернизируемой с учетом новой информации об объекте исследования, полученной в ходе отработки системы автоматизации.

4. Непрерывность процесса научного исследования. Исследовательская деятельность носит, как правило, непрерывный характер, так как любой исследователь по завершении некоторого этапа работ обычно намечает их дальнейшее развитие, формулируя новую программу работ (новую цель, задачу и т.п.) для того же или другого объекта. Это означает необходимость непрерывного развития, совершенствования соответствующей АСНИ.

5. Уникальность научного исследования. Каждое научное исследование имеет определенные особенности, отличающие его от других аналогичных исследований. Эти особенности могут проявляться в разных исследованиях в неодинаковой степени, однако в той или иной мере их присутствие обязательно, поскольку иначе такое исследование уже не может относиться к категории научного. Черты уникальности могут быть связаны с особенностями самого объекта исследования, постановки задачи, метода экспериментирования, используемого оборудования и т.д.

В связи с уникальностью научных исследований каждая АСНИ, как правило, обладает специфическими чертами, присущими только данной системе в части ее технического, программного или научно-методического обеспечения.

6. Многообразие исследовательских задач. Человек стремится познать окружающий мир во всей его сложности, во взаимосвязи всех сторон явления, поведения и свойств частных процессов. Выделим некоторые основные моменты, позволяющие более конкретно указать, в чем же состоит многообразие исследовательской работы. К их числу обычно относят:

- многообразие объектов исследования в различных отраслях наук, в отдельной отрасли и даже в пределах одной предметной области;
- разнообразие физических процессов, характеризующих поведение сколь угодно сложного объекта исследования;
- разномасштабность (пространственная и временная) изучаемых объектов и соответствующего оборудования;
- разнообразие условий реализации исследований (от стационарных лабораторий до исследований с помощью средств, находящихся на подвижных носителях в неблагоприятных условиях на значительном удалении от исследователя);
- разнохарактерность исследований (от фундаментальных исследований до рутинных измерений);

- разнообразие целевых установок при проведении исследований (от исследований, ориентированных на совершение научных открытий, до сугубо утилитарных, предназначенных для решения конкретной частной задачи в узкой предметной области).

С точки зрения разработки АСНИ многообразие исследовательских задач затрудняет использование стандартных решений, требуя учета специфических черт данного научного исследования.

Анализ перечисленных основных черт научных исследований с позиций создания АСНИ свидетельствует об их сложности как объекта автоматизации. Поэтому целесообразно выделять классы научных исследований по совокупности определенных однотипных свойств, например, по отраслям наук (исследования в области физики, химии, биологии и т.д.).

Составные части АСНИ

Принято выделять некоторые основные составные части, входящие в АСНИ. К их числу обычно относят:

1. Научно-методическое обеспечение (НМО) включает в себя различного рода методы, способы, методики, алгоритмы проведения эксперимента, обработки и представления экспериментальных данных. Можно выделить частные виды НМО, характерные для достаточно узкой предметной области, и НМО общего применения, которое одинаково успешно может использоваться в различных предметных областях.

2. Техническое обеспечение АСНИ включает в себя комплекс используемых технических средств: ЭВМ, измерительную аппаратуру, экспериментальную установку, устройства связи с объектом и другие устройства, обеспечивающие функционирование АСНИ и ее отдельных частей.

3. Программное обеспечение АСНИ содержит документы с текстами программ, программы на машинных носителях, различные эксплуатационные документы, позволяющие реализовать основные функции АСНИ, различные режимы ее работы, эффективное взаимодействие пользователей с техническими ресурсами АСНИ.

4. Информационное обеспечение АСНИ включает в себя различного рода базы и банки данных, информационно-поисковые системы, справочные и обучающие системы, а также программные средства, обеспечивающие работу с имеющейся и вновь поступающей информацией, и соответствующие информационные документы.

5. Метрологическое обеспечение АСНИ содержит дополнительную аппаратуру, методические материалы, инструкции и т.д., предназначенные для обеспечения необходимых метрологических характеристик системы, точности и достоверности измерительной информации.

6. Организационно-правовое обеспечение включает в себя методические и руководящие материалы, положения, приказы, квалификационные требования, инструкции для пользователей, которые регламентируют взаимодействие пользователей с системой, порядок эксплуатации и развития АСНИ, способы организации доступа отдельных исследователей и групп исследователей к ресурсам коллективного пользования.

В дальнейшем основное внимание будет уделяться первым трем составным частям АСНИ, т.е. научно-методическому, техническому и программному обеспечению АСНИ.

Принципы построения АСНИ

В основу АСНИ положены принципы обмена информацией между исследователем и экспериментальной установкой в реальном масштабе времени.

При этом АСНИ осуществляет:

- сбор измерительной информации, ее первичную обработку (в соответствии с алгоритмом процесса исследования);
- обмен управляющей информацией между экспериментальной установкой и ЭВМ;
- хранение информации и обмен ею с другими ЭВМ.

Современные АСНИ строятся с использованием определенных основополагающих принципов. Наиболее важными из них являются следующие:

1. Комплексность, т.е. изначальная направленность АСНИ на решение всего комплекса задач, стоящих перед исследователем; реализация всех основных функций, возлагаемых на такого рода системы; обеспечение возможности применения АСНИ на различных этапах исследований.

2. Многоуровневая организация. В соответствии с этим принципом при построении современных АСНИ выделяется несколько структурных уровней. Каждый из них ориентирован на решение определенной группы однородных по сложности исследовательских задач, которые, в свою очередь, требуют соответствующих технических средств и организации тех или иных режимов работы. Подобная организация позволяет реализовать принцип комплексности в условиях ограничения возможных затрат на создание и эксплуатацию АСНИ.

3. Расширяемость (модульный принцип построения), т.е. использование при создании АСНИ таких технических решений, которые бы делали возможным дальнейшее быстрое развитие системы, увеличение количества пользователей, развитие функциональных возможностей системы без переделок и изменений принципиального характера.

4. Адаптируемость, которая означает достижение большей гибкости АСНИ, возможности ее подстройки и модернизации с учетом конкретных особенностей данной исследовательской задачи, данного объекта исследований.

5. Коллективность использования. В соответствии с данным принципом АСНИ строятся как системы коллективного пользования. Это означает, с одной стороны, организацию коллективного доступа к наиболее сложным и дорогостоящим системам АСНИ, а с другой - объединение усилий при создании и последующем использовании АСНИ, когда отдельные удачные разработки и результаты исследований становятся общедоступными и могут применяться всеми пользователями системы.

6. Интеграция АСНИ, включающая в себя два аспекта:

- использование технических ресурсов АСНИ для решения задач иного характера (учебных, организационно-управленческих, расчетных, фоновых и т.п.);

- тесное взаимодействие с автоматизированными системами других типов (САПР, АСУТП, АСУП). Создание комплексных систем, в первую очередь типа АСНИ-САПР, когда одни и те же средства используются и для проведения исследований научного характера, и для целей автоматизированного проектирования соответствующего технического объекта, при котором результаты исследований выступают в качестве одной из составляющих исходной информации или служат для оценки качества проектных решений.

7. Типизация инженерных решений при создании АСНИ означает разработку таких компонентов автоматизированных систем, которые могут найти применение при автоматизации основной массы научно-технических исследований в самых разных предметных областях. Такие решения способствуют проведению единой технической политики при построении АСНИ в отдельных отраслях науки.

Типизация инженерных решений охватывает целый комплекс проблем, возникающих при создании АСНИ, а именно связанные с разработкой общей структуры системы, ее технического, программного и научно-методического обеспечения, а также с выбором конкретной конфигурации системы или отдельных ее частей.

Типовая структура АСНИ

Различные элементы исследования требуют и различной технической базы в рамках АСНИ. Например, разработка теоретических вопросов часто сопровождается проведением громоздких расчетов, моделированием, поиском научной информации с помощью информационно-поисковой службы, что требует значительной мощности и объема памяти ЭВМ. С другой

стороны, обращение к этим ресурсам АСНИ производится относительно редко и необязательно с высокой оперативностью. Вместе с тем, операции, связанные с проведением автоматизированного эксперимента, всегда осуществляются в масштабе реального времени, и нет необходимости в значительных вычислительных мощностях. Для реализации в рамках АСНИ самых разных, в том числе и самых трудоемких, элементов исследований при разумных затратах на создание АСНИ современные системы строятся по многоуровневому принципу. Наиболее целесообразна структура, содержащая три уровня: *объектный, инструментальный и сервисный (базовый)*.

Объектный уровень характеризуется связью с объектом исследований. Его назначение состоит в организации процесса экспериментирования, т.е. реализации управления экспериментальной установкой, регистрации данных, их оперативной обработки, накопления и представления первичных результатов исследователю, в том числе и оказание ему помощи в интерпретации результатов эксперимента и принятии решения о дальнейшем проведении исследований. На объектный уровень также возлагают операции, связанные с проверкой и тестированием экспериментального оборудования, текущей регистрацией и документированием данных.

Инструментальный уровень предназначен для проведения достаточно сложных видов обработки экспериментальных данных, научных расчетов и моделирования, если они не требуют слишком больших мощностей вычислительного оборудования. Здесь осуществляется накопление и длительное хранение информации, полученной в результате исследований, формируются архивы и банки данных по отдельным проблемам исследований. На инструментальном уровне осуществляется отработка различных алгоритмов и программ, составленных пользователем, в том числе и программ, используемых на объектном уровне.

Базовый (или сервисный) уровень используется для осуществления наиболее сложных и громоздких научных расчетов, моделирования, обработки и представления информации, формирования крупных банков и баз данных, создания информационно-поисковой системы.

Трехуровневая организация современных АСНИ позволяет, с одной стороны, предоставить исследователю необходимые средства вычислительной техники и автоматизации на всех этапах исследования, а с другой - сократить затраты на создание системы, уменьшить количество ЭВМ, периферийного оборудования и т.д.

Необходимо подчеркнуть, что для АСНИ *наиболее важным является объектный уровень,* так как именно на этом уровне фигурирует исследователь, роль которого является ключевой. Именно на объектном уровне в первую очередь регистрируется новая информация об изучаемом явлении или объекте. Поэтому АСНИ, являясь многоуровневыми системами, не относятся к категории иерархических систем. Можно считать, что верхние

этажи этой организации - инструментальный и базовый уровни - являются вспомогательными, оказывающими дополнительные услуги при извлечении полезной информации, разработке и проверке теоретических положений на основе экспериментальных данных.

Типовые конфигурации АСНИ

В зависимости от области деятельности, в которой проводятся экспериментальные исследования, конфигурация экспериментальной установки будет различной, но во всяком экспериментальном исследовании можно выделить несколько основных функциональных частей.

Прежде всего имеется *экспериментальная установка* с объектом (рис. 2), воспроизводящая исследуемый процесс или явление. Это могут быть ускоритель элементарных частиц с мишенью, аэродинамическая труба с моделью самолета и т.п. Процесс должен воспроизводиться при определенных значениях определяющих его параметров. Для задания и удерживания этих параметров объект снабжается *системой управления*.

Обязательной частью любого автоматизированного экспериментального комплекса является *измерительная система (ИС)*. Измеряемыми величинами в экспериментальных исследованиях являются физические величины (напряжение, ток, температура, давление, линейные и объемные перемещения и др.). Первоначальными источниками информации о значении измеряемых величин служат датчики (Д). Их основная задача состоит в преобразовании измеряемого параметра в электрический сигнал. Датчики чаще всего выдают сигнал в аналоговой форме. В случае большого числа датчиков может быть использован коммутатор ($К_М$). Для обеспечения работы последующих блоков комплекса сигналы с датчиков могут быть усилены в масштабаторе (УМ) и переданы на измерительное устройство. Измерительные устройства, используемые в АСНИ, имеют, как правило, цифровую индикацию и цифровое представление результатов на выходе. Узел, переводящий информацию из аналоговой формы в цифровую, носит название аналого-цифрового преобразователя (АЦП).

Для управления экспериментальной установкой информация из цифровой формы, как правило, преобразуется в аналоговую с помощью цифро-аналоговых преобразователей (ЦАП).

Следующей частью системы является *узел обработки*. Он включает в себя процессор, запоминающее устройство и систему математического обеспечения. Чаще всего таким устройством является ЭВМ (ПК).

На рис. 2 показана типовая конфигурация АСНИ. В конкретных реализациях могут быть незначительные отклонения от приведенной схемы. Например, может быть предусмотрено только цифровое управление, тогда становится ненужным цифро-аналоговый преобразователь. В качестве уст-

ройства вывода графической и текстовой информации может использоваться один дисплей и т.п.

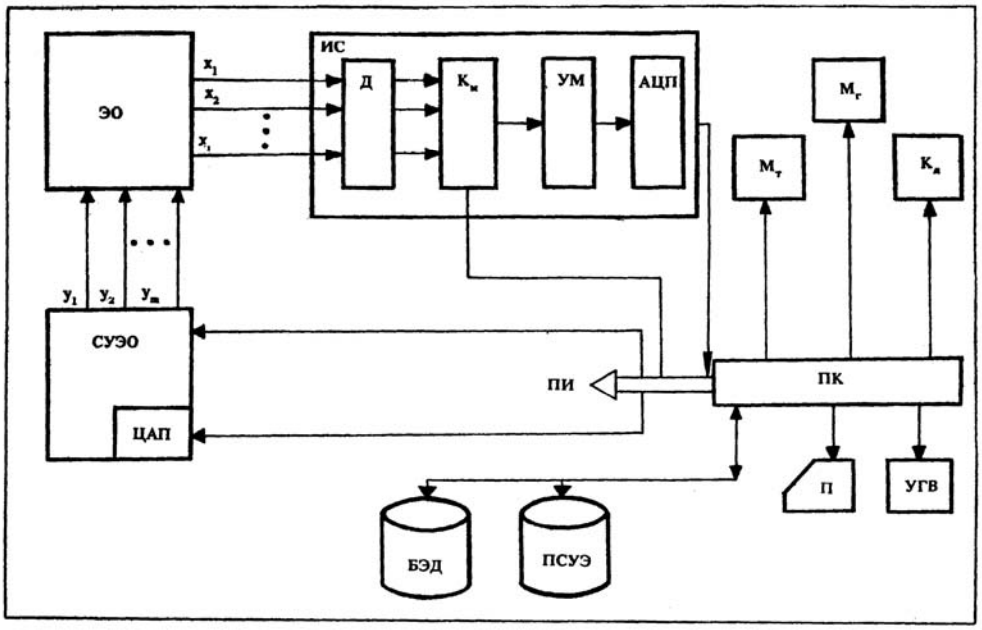


Рис. 2

ЭО - экспериментальный объект; $У_1$ $У_2$,...,$У_m$ - управляющие сигналы; X_1, X_2,...X_N - измеряемые параметры; ПК - персональный компьютер; ИС - измерительная система; Д – датчики; $К_М$ - коммутатор; УМ - усилитель-масштабатор; АЦП - аналого-цифровой преобразователь; ЦАП - цифро-аналоговый преобразователь; ПИ - приборный интерфейс; СУЭО - подсистема управления экспериментальным объектом; $М_т$ - монитор текстовый; $М_г$ - монитор графический; $К_л$ - клавиатура; УГВ - устройство графического вывода (графопостроитель); П - печатающее устройство; БЭД - база экспериментальных данных; ПСУЭ - программная система управления экспериментом

Современные ЭВМ, в частности персональные компьютеры, обладая высокими техническими характеристиками, позволяют использовать их в таких приложениях, как измерительные приборы, осциллографы и т.д., путем простого программирования и подключения соответствующих дополнительных устройств.

Обмен информацией в графической форме является исключительно эффективным средством для представления объектов со сложной структурой. На экране дисплея возможно формирование целой системы приборных шкал (вольтметров, амперметров, омметров, фотометров и многих других измерительных приборов), регистрирующих те или иные параметры экспериментального объекта.

Содержание экспериментальных исследований

Экспериментальным исследованием - *экспериментом* (от латинского слова *experimentum* - проба, опыт) - называется метод познания, при помощи которого исследуются реальные явления действительности, реальные функциональные связи, характеризующие состояние изучаемого объекта. Важной задачей эксперимента является проверка гипотез и предсказаний теоретической физики. Это так называемые решающие эксперименты (*experimentum crucial*). Исходя из общего утверждения, что критерием истины является практика, можно сказать, что только эксперимент может быть критерием истины в научных исследованиях.

Формально под экспериментальным исследованием можно понимать поиск функциональной зависимости между параметрами, описывающими состояние системы и феноменологическое описание или объяснение обнаруженных новых закономерностей.

Пусть исследуемая система, состояние которой описывается набором параметров $X_1, X_2, X_n; X, Y,$ помещена в квазиадиабатическую оболочку, на которую извне действуют силы $q_1, q_2,...,q_k$, как это условно изображено стрелками на рис. 3.

Рис. 3

Запишем априори неизвестное состояние системы в виде

$$\varphi(X_1, X_2, \ldots, X_n; X, Y; q_1, \ldots, q_k) = 0 .$$

Для полного изучения системы необходимо в первую очередь установить функциональную связь между всеми параметрами X_i, X, Y.

В реальных исследованиях обычно в силу причин, о которых сказано ниже, приходится такой полный эксперимент разбить на ряд этапов. Вначале из тех или иных соображений выбирают всего два параметра (например, X и Y) и изучают зависимость одного из выбранных параметров от другого в условиях, когда все остальные параметры поддерживаются на постоянном уровне.

Такая организация исследования обусловлена рядом причин. Прежде всего, при одновременном изменении нескольких параметров (n >2) существенно возрастает объем получаемой информации, и ее запись возможна только с помощью автоматических устройств с большим объемом памяти и налаженной системой выборки необходимых данных из огромного массива полученной информации.

Пусть, например, мы планируем провести исследование состояния реального газа, описываемого уравнением Ван-дер-Ваальса

$$(p + a/V^2)(V - b) = RT,$$

в которое входят всего три параметра: p - давление, V - объем одного моля газа, T - температура; a и b - некоторые константы, характеризующие отклонение газа от идеального. Для исследования при фиксированной температуре $T = T_0 = const$ зависимости давления газа от объема в интервале давлений от 1 до 100 *атм* с интервалом изменения давления 0,1 *атм* необходимо записать 10^3 пар чисел. Если же одновременно будет изменяться и температура скачками 0,1°K в интервале от 200°K до 300°K, то объем информации возрастет до 10^6 троек чисел. Возникает проблема и на этапе записи и особенно на этапе обработки большого объема информации.

Эксперименты, в которых одновременно изменяется больше чем два параметра, называются многофакторными и в физических исследованиях встречаются не очень часто. Важно подчеркнуть, что многофакторные эксперименты являются фактически системой однофакторных исследований.

Зафиксируем значения n внутренних параметров X_i, внешних q_j и разрешим уравнение состояния системы относительно двух внутренних параметров X и Y, записав его в виде $Y = \phi(X; X_1,...,X_n; q_1,...,q_k)$.

В реальных исследованиях никогда не удается настолько точно стабилизировать значения параметров X_i, чтобы они не искажали результаты измерений. Точно так же любая реальная квазиадиабатическая оболочка не может полностью исключить внешние воздействия на исследуемую систему.

Найдем допустимые пределы изменения параметров в условиях хорошего эксперимента. Под термином *условия хорошего эксперимента* понимается такая его организация, при которой различные погрешности из-за

нестабильности параметров намного меньше изменений исследуемых величин X и Y.

Пусть минимальное значение изменения величины X (интервал изменения) равно ΔX, тогда изменение второй исследуемой величины Y будет равно $\Delta Y = \frac{\partial \varphi}{\partial X}\Delta X$.

Изменение Y может быть вызвано также неконтролируемым изменением параметров X_i и q_i

$$\Delta Y(X_i) = \frac{\partial \varphi}{\partial X_i}\Delta X_i \ ; \ \Delta Y(q_j) = \frac{\partial \varphi}{\partial q_j}\Delta q_j \ . \qquad (1)$$

Условия «хорошего» эксперимента можно записать в следующем виде: для всех X_i, q_i

$$\Delta Y\left(X_i\right) \square \ \Delta Y \ ; \ \Delta Y\left(q_j\right) \square \ \Delta Y \ , \qquad (2)$$

т.е. допустимы такие вариации внутренних и внешних параметров, при которых исследуемая величина Y изменяется намного меньше, чем при направленном изменении параметра X на величину ΔX.

При большом числе параметров в случае коррелированного изменения величин возмущения условие (2) может оказаться недостаточным. В этом случае сумма малых изменений $\Delta Y(X_i)$ или $\Delta Y(q_i)$ может привести к заметному изменению $\Delta Y(X_i, q_i)$, сравнимому с ΔY. Условия «хорошего» эксперимента требуют здесь выполнения дополнительных соотношений

$$\Delta Y(X) = \sum_i^n \frac{\partial \varphi}{\partial X_i}\Delta X_i \square \ \Delta Y \ ;$$

$$\Delta Y(q) = \sum_j^k \frac{\partial \varphi}{\partial q_j}\Delta q_j \square \ \Delta Y \ . \qquad (3)$$

Условия (2) или (3) могут быть основанием для выбора методов стабилизации параметров, которые обеспечивают необходимый уровень стабилизации. Конкретное исследование влияния различных факторов на результаты измерений является одной из важнейших задач экспериментальных исследований.

Для описания состояния системы необходимо знать численные значения всех параметров, которые могут быть определены только в результате измерений. Таким образом, в основе любого экспериментального исследования лежат измерения физических величин. Под физической величиной в соответствии с [1] понимается особенность физического объекта, характеризующая его свойство, состояние или происходящий в нем процесс и имеющая качественное и количественное содержание.

Аналитическое описание обнаруженной в результате измерений зависимости одних параметров от других является второй важнейшей задачей экспериментальных исследований.

Третьим этапом экспериментальных исследований должно быть феноменологическое описание обнаруженных зависимостей полученных результатов. Сам исследователь, выполнивший сложные измерения и знающий массу деталей, должен попытаться объяснить полученные результаты.

Таким образом, экспериментальное исследование состоит, по крайней мере, из трех основных этапов: измерение, аналитическое описание, феноменологическое объяснение результатов эксперимента.

Определение измерений. Типы измерений

Что же такое измерение? Согласно [1] "*измерение есть нахождение физической величины опытным путем с помощью специальных технических средств"*.

Сам процесс измерения может быть описан следующим образом: измерение - это операция, посредством которой определяется отношение одной, измеряемой, величины к другой, однородной, величине, принимаемой за единицу. Величина, выражающая такое отношение, называется численным значением измеряемой величины. В общем смысле измерение есть познавательный процесс, заключающийся в сравнении путем эксперимента исследуемой величины с некоторой однотипной физической величиной, принятой за единицу.

Результат каждого измерения представляет собой именованное (размерное) число, состоящее из наименования (размерности) измеряемой величины и числа, показывающего отношение измеренной величины к другой величине, принятой за единицу для данных измерений - так называемую эталонную величину.

Из приведенных определений процедуры измерения видно, что для выполнения измерения необходимо иметь, во-первых, эталон - единицу измерения, а во-вторых, способ сравнения исследуемой величины с эталонной.

Наиболее распространены четыре метода измерений: *прямые*, *совместные*, *косвенные* и *совокупные*.

Прямые измерения. Прямыми называются «измерения, при которых искомое значение величины находят непосредственно из опытных данных» [1]. Фактически только прямые измерения являются истинными, а остальные типы измерений - лишь различные комбинации прямых измерений. Прямыми измерениями определяются численные значения физических величин, являющихся основными единицами системы и имеющих простую размерность. Это такие величины, как длина, масса, время. Прямыми измерениями могут определяться численные значения и тех физических величин, единицы которых являются производными. Так, ускорение тела можно найти с помощью прямых измерений, определяя растяжение

пружины с грузом. В этом случае требуется предварительная градуировка растяжения пружины.

В прямых измерениях всегда предполагается наличие стандартных мер - эталонов и разработанных методов измерений.

В научных исследованиях прямые измерения используются как составные части так называемых совместных измерений.

Совместные измерения. Совместными называются одновременные измерения двух или нескольких неодноименных величин, характеризующих состояние исследуемой системы, для нахождения зависимости между величинами. Как было отмечено, совместные измерения составляют основу научного исследования.

Рассмотрим пример совместных измерений - исследование зависимости сопротивления полупроводника от температуры. Блок-схема установки для выполнения исследования представлена на рис. 4, из которого видно, что установка состоит из двух каналов для прямых измерений: для измерения температуры и измерения сопротивления. Отметим важную особенность совместных измерений: определение всех параметров должно проводиться одновременно.

Рис. 4

Косвенные измерения. Косвенными называются такие измерения, в которых исследуемая величина определяется с помощью известных соотношений между физическими величинами, найденными в результате прямых измерений. Надо отметить, что косвенные измерения не являются истинными измерениями - это вычисления физической величины на основе измерения других величин. Так, вязкость жидкости по методу Стокса вычисляется по формуле Стокса на основании результатов четырех прямых измерений: веса шарика, его диаметра, скорости установившегося падения и плотности жидкости. (Взяв плотность жидкости из справочника, мы пользуемся результатами прямых измерений, выполненных ранее другими исследователями.) Иногда очень трудно найти резкую границу между прямыми и косвенными измерениями. Например, скорость движения корабля или самолета можно определить в результате двух прямых измерений: времени и длины пройденного пути. Однако можно воспользоваться трубкой Пито, которая предварительно проградуирована в единицах скорости, и, следовательно, измерение условно может быть отнесено к прямым.

Совокупные измерения. Совокупными называются измерения нескольких одноименных величин, при которых результаты измерений находят решением системы линейных уравнений.

Виды экспериментальных исследований

Роль эксперимента при проведении научных исследований очень многообразна, начиная от проверки исходных идей и кончая испытаниями образцов новой техники в реальных условиях.

Попробуем классифицировать сами экспериментальные исследования по такой совокупности параметров, которая определяет различные аппаратные и математические средства, необходимые для их автоматизации.

Качественный и количественный эксперимент

Вначале рассортируем эксперименты на два класса - *качественный* и *количественный.*

Целью *качественного* эксперимента является установление только факта существования явления.

Например, при исследовании прохождения света через границу сред с разными показателями преломления можно легко установить факт существования преломления.

Задача *количественного* эксперимента - установление количественных связей между параметрами, описывающими состояние системы. Например, при исследовании явления преломления количественными (измерительными) экспериментами выступают опыты, в которых установлены количественные соотношения между углами падения и преломления и найдена функциональная связь между синусами углов и коэффициентом преломления.

Учитывая широкий спектр направленности и условий проведения эксперимента в ходе научно-технических исследований, удобно всю совокупность экспериментальных исследований такого рода разбить на ряд групп, выделив прежде всего исследования лабораторного и промышленного типов (рис. 5).

Лабораторный эксперимент

Лабораторный эксперимент характеризуется наличием специальной экспериментальной установки, с помощью которой объект исследования изолируется от влияния различного рода мешающих факторов («внешней среды») и которая позволяет производить регистрацию измерительной информации и целенаправленное воздействие на объект исследования. Именно в лабораторных условиях в максимальной степени могут реализоваться идеи управляемого оптимального эксперимента, когда каждый из

управляющих факторов и их совокупность в целом изменяются по определенной программе, обеспечивающей максимальную информативность поступающих данных и наиболее эффективное решение исследовательской задачи.

Рис. 5

Можно выделить несколько градаций лабораторного эксперимента.

А. *Лабораторный эксперимент для проведения фундаментальных научно-технических исследований.* В ходе таких исследований анализируются возможности применения открытий в области физики, химии, биологии и т.д. для создания объектов новой техники или технологии. Именно с их помощью находят принципиально новые подходы и решения, обеспечивающие качественный скачок в развитии технических средств, технологического оборудования, и т. п.

Ускорители элементарных частиц, реакторы - примеры сложных экспериментальных установок для исследовательского эксперимента.

Б. *Лабораторный инженерный эксперимент.* Это наиболее характерная и массовая разновидность лабораторного эксперимента в технических науках, с помощью которого научные работники и инженеры-исследо-

ватели осуществляют отработку технических решений при создании новых или совершенствовании имеющихся приборов, устройств или их отдельных частей. Такая отработка может осуществляться путем экспериментирования над реальными образцами соответствующих изделий или отдельными их частями (фрагментами, материалами). Это лабораторный *натурный* эксперимент.

В ряде случаев предпочтительным оказывается экспериментирование не над самим объектом, а над его моделью, тогда это называется *модельным* экспериментом. В зависимости от способа моделирования различают:

- *масштабное моделирование*, когда физические процессы, протекающие в объекте и модели, одинаковы, а различие между ними носит чисто масштабный характер. Но простое геометрическое подобие сооружения еще не обеспечивает подобия явления. Например, изучается водослив плотины. Размеры ее уменьшены линейно, но масса падающей воды уменьшилась в кубе масштаба. Были выработаны критерии подобия, определяющие правила моделирования по определяющему параметру;

- *физическое моделирование*, при котором реальный объект заменяется объектом иной физической природы с поведением, подобным поведению изучаемого объекта, но более удобным при экспериментировании;

- *аналоговое моделирование*, где реальный объект заменяется его электрическим аналогом, воспроизводящим дифференциальные уравнения, описывающие свойства исследуемого объекта.

Если различные явления описываются одними и теми же уравнениями, то можно одно из явлений выбрать за основу для модели, а остальные выражать через него.

Модельным выбирается то явление или процесс, в котором можно легче и точнее произвести измерения. Так как лучше всего разработаны измерения электрических величин, то и модели стараются выполнить на электрических схемах. Этот вид моделирования носит название *электромоделирование*.

Для общности в указанный перечень следовало бы включить и *математическое моделирование,* когда объект исследования полностью моделируется в цифровом виде на ЭВМ. Однако здесь центр тяжести лежит в алгоритмической области, не затрагивая задачу организации связи объекта исследования и ЭВМ.

Составляя математическую модель, нужно стремиться оставлять для рассмотрения лишь наиболее существенные параметры, делать математическое описание процесса как можно проще. Не надо забывать, что для многих известных уравнений и сегодня не найдено методов численного решения на ЭВМ.

Математическое моделирование - это специфический «чистый эксперимент» на основе полной автоматизации. Следует помнить, что при математическом моделировании всегда возникает важная проблема, связанная

с однозначным соответствием идеализированного объекта и реальной системы.

Промежуточное положение между натурным и модельным занимает *полунатурный* эксперимент, когда лишь некоторая часть исследуемого объекта заменяется той или иной моделью.

В. *Рутинный лабораторный эксперимент.* Эта разновидность эксперимента, связанная с реализацией жесткой программы экспериментирования по фиксированной методике, во многом близка к стандартным техническим измерениям, проводимым, например, при испытаниях серийных изделий, проверке их соответствия требованиям нормативных документов.

Промышленный эксперимент

Промышленный эксперимент характеризуется тем, что объект исследования изучается в условиях, соответствующих возможным реальным режимам его эксплуатации.

В промышленном эксперименте установка применяется для сложного измерительного эксперимента. В нем тип исследуемого процесса и уравнения, его описывающие, известны. Например, натурный эксперимент часто применяется для исследований прочности и усталости конструкций и механизмов [2].

От натурных испытаний в рабочих условиях (которые, по существу, также являются видом эксперимента) натурный промышленный эксперимент отличается тем, что, во-первых, подвергаются исследованию не всегда конструкции или изделия в целом, а чаще их агрегаты и отдельные части. Во-вторых, нагрузки (даже стохастические) задаются программным путем, что делает сопоставимыми два и более отдельно проведенных эксперимента. Программное задание нагрузок позволяет сократить срок проведения эксперимента на усталостную прочность и ресурс изделия.

Существующие варианты промышленного эксперимента указаны на том же рис. 5, где выделены такие его разновидности, как:

а) натурные испытания, т.е. исследования в реальных эксплуатационных условиях для различных, в том числе и критических, режимов;

б) стендовые испытания, когда хотя бы часть внешней для объекта исследования среды имитируется с помощью некоторых вспомогательных технических средств;

в) настроечный (поисковый) эксперимент, предназначенный для определения в реальных промышленных условиях оптимальных характеристик объекта или условий его функционирования с учетом некоторого выбранного критерия оптимальности;

г) рутинный промышленный эксперимент, ориентированный на массовое измерение параметров объектов по стандартным методикам (например, массовых приемо-сдаточных испытаний).

Ясно, что даже в рамках научно-технических исследований характер требований, предъявляемых к соответствующей системе автоматизации, может сильно различаться в зависимости от типа и градации исследований. В связи с этим в последующем будем в основном ограничиваться рассмотрением вопросов построения АСНИ, предназначенных в первую очередь для автоматизации лабораторного эксперимента градаций А и Б.

ЭВМ в АСНИ

При создании АСНИ большое значение уделяется использованию средств вычислительной техники. При этом повышается эффективность научных исследований, которая заключается в следующем:

1. В несколько раз сокращается цикл исследований (экспериментов) за счет ускорения подготовки и проведения эксперимента, оперативного использования результатов экспресс-анализа, проводимого в реальном масштабе времени, сокращения времени обработки и систематизации данных, уменьшения числа ошибок при измерении и обработке.

2. Увеличивается точность результатов и их достоверность, так как в АСНИ возможно использование методов, снижающих влияние накапливающихся ошибок округления при вычислении промежуточных результатов.

3. Повышается качество и информативность эксперимента за счет увеличения числа контролируемых параметров (по сравнению с "некомпьютерными" исследованиями) и более тщательной обработки данных.

4. При интерактивном взаимодействии с АСНИ достигаются усиление контроля над ходом эксперимента и возможность его оптимизации.

5. Сокращается штат участников эксперимента, повышается производительность исследователя.

6. Очень важным является то, что результаты экспериментов структурируются и выводятся оперативно в наиболее удобной для оператора форме. Например, вместо просмотра километровых таблиц данных их структурируют в виде графических объектов. Так, зависимость от двух аргументов удобно представлять средствами трехмерной графики - в одну картинку интегрируют миллионы измерений.

Возможности современных ЭВМ

Считается, что человек выполняет одно арифметическое действие над 3 - 4-разрядными числами в среднем около минуты. За рабочий день (человек не может работать беспрерывно) это составит около 500 действий. Современные ЭВМ достигают уже сотен миллионов операций в секунду. Один миллион операций - это несколько лет труда человека. ЭВМ выпол-

няет эту работу в доли секунды, при этом вычисления проводятся с большей точностью (над числами до 16 - 32 разрядов).

С использованием ЭВМ стало возможным решать задачи ядерной физики, космонавтики, аэродинамики, прогнозирования погоды, обработки изображений и т.д. Эти задачи требуют высокого быстродействия и точности вычислений.

О поколениях ЭВМ

В соответствии с техническими принципами и исторически различают пять поколений ЭВМ.

Основным активным элементом машин *первого* поколения являлась электронная лампа. Компьютеры такого типа появились в пятидесятых годах нашего века. Типичные представители ЭВМ этого поколения среди отечественных - БЭСМ-1, Минск-1, Урал-1, Урал-2, М-1, М-3, Стрела и др. Быстродействие их не превышало 2 - 3 тыс. операций в секунду, емкость оперативной памяти – 2 048 машинных слов (длина слова - 48 разрядов).

В ЭВМ *второго* поколения в качестве элементной базы использовались транзисторы, что существенно увеличило емкость оперативной памяти, надежность и быстродействие. К машинам этого класса относятся Урал-14, Урал-16, Минск-22, Минск-32, БЭСМ-3, БЭСМ-4, М-220, БЭСМ-6, МИР-2, Наири и др. Быстродействие БЭСМ-4, М-220 порядка 20 - 30 тыс. операций в секунду, а оперативная память - соответственно 8 194, и 16 384 слов. У БЭСМ-6 быстродействие около миллиона операций в секунду и память до 128 Кслов.

Основа машин *третьего* поколения - интегральные схемы. К ним относятся все ЕС ЭВМ – от ЕС-1010 (быстродействие до 10 тыс. операций в секунду, объем оперативной памяти от 8 до 64 Кб), включая ЕС-1020, ЕС-1040, ЕС-1060, до ЕС-1066 (более 2 млн. операций в секунду, 8 192 Кб), а также Электроника-60, Электроника-100/125, СМ-3, СМ-4 и др. Машины третьего поколения оперируют произвольной буквенно-цифровой информацией, единица адресации памяти - байт, а не слово (длина слова стала 4 байта, используются полуслова и двойные слова). У машин третьего поколения появилась возможность параллельной работы устройств и, как следствие, возможность работы нескольких пользователей в режиме разделения времени.

Элементная база компьютеров *четвертого* поколения - БИС. Примером может служить многопроцессорный вычислительный комплекс "Эльбрус". Эльбрус-1КБ имел быстродействие до 5,5 млн. операций с плавающей точкой в секунду, а Эльбрус-2 - производительность до 120 млн. операций в секунду, емкость оперативной памяти до 144 Мб, или 16 Мслов (слово 72 разряда), максимальная пропускная способность каналов ввода-вывода – 120 Мб/с.

Появление микропроцессоров в 70-е годы привело к созданию множества ПК от первых 8-разрядных до 64-разрядных и более в наши дни.

Особенности использования ЭВМ в АСНИ

Отличительной особенностью использования ЭВМ в АСНИ является выполнение большинства (или всех) операций в реальном масштабе времени. Термин *"реальное время"* используют в том случае, когда требуется оперативно реагировать на входные сигналы, причем задержка реакции должна быть конечной и не превышать определенного значения. В различных приложениях этот термин понимается по-разному - от долей секунд до минут. Главное, чтобы ответ пришел не позднее того времени, когда еще можно внести коррекции в исследуемый процесс. *Для ЭВМ, используемых в АСНИ, вычисление управляющих воздействий за время, больше требуемого, приравнивается к получению неправильного результата.*

При использовании ЭВМ в АСНИ можно решать следующие задачи

- принимать информацию от датчиков о состоянии среды и объекта;
- рассчитывать в "реальном времени" управляющие воздействия и передавать их на исполнительные механизмы;
- отображать информацию о текущем состоянии системы оператору на дисплее в диалоговом режиме;
- принимать и обрабатывать оператора по изменению условий эксперимента;
- передавать или принимать информацию от других ЭВМ.

Каждая из этих задач решается с помощью своей программы, которая находится в памяти ЭВМ и выполняется по мере необходимости.

Архитектурная организация ЭВМ основных классов и типов

В современной вычислительной технике (ВТ) основой представления информации являются электрические сигналы, допускающие две формы представления - *аналоговую* и *дискретную.* В первом случае величина напряжения является *аналогом* значения некоторой измеряемой переменной: например, подача на вход напряжения в 1.942 В эквивалентна вводу числа 19.42 (при масштабе 0.1). Во втором случае - в виде нескольких различных напряжений, *эквивалентных* числу единиц в представляемом значении переменной. При *аналоговом* представлении информации значения измеряемых величин могут принимать любые допустимые значения из заданного диапазона, т.е. представляется бесконечный спектр значений измеряемой величины на заданном отрезке. При *дискретном* представлении информа-

ции значения измеряемых величин носят дискретный (конечный) характер в измеряемом диапазоне.

Сравнительный анализ обоих форм представления информации показывает, что при создании ВТ аналогового типа требуется меньшее число компонентов, но сложность ее быстро возрастает за счет необходимости различать значительно большее число (вплоть до бесконечности) состояний сигнала. Аналоговая ВТ позволяет легко интегрировать сигнал, выполнять над ним любое функциональное преобразование и т.д. За счет этого и ряда других особенностей возможно решение некоторых классов задач во много раз быстрее, чем на *дискретной* ВТ. Недостатками аналоговой формы представления информации является сложность реализации устройств для ее логической обработки, длительного хранения и высокой точности измерения. Поэтому *аналоговые вычислительные машины (АВМ)* предназначены в первую очередь для решения задач, описываемых системами дифференциальных уравнений: управление непрерывными процессами; моделирование в гидро- и аэродинамике; исследование динамики сложных объектов и др. Но АВМ не могут решать задач, связанных с хранением и обработкой больших объемов информации различного характера; задач с высокой степенью точности и др., с которыми легко справляются *цифровые вычислительные машины* (ЦВМ), использующие *дискретную* форму представления информации. Положительные черты обоих типов совмещает *гибридная* ВТ, включающая как аналоговые, так и дискретные устройства обработки информации.

Аналоговая вычислительная техника

Интенсивное развитие АВМ наступает в начале 50-х годов XX века, после создания стабилизированного операционного усилителя постоянного тока, который позволил создавать отвечающие необходимым требованиям функциональные блоки, выполняющие разнообразные математические операции: арифметические, интегрирование, дифференцирование и др.

В отличие от дискретной ЭВМ в основу АВМ заложен принцип *моделирования,* а не *счета* [3]. При использовании в качестве модели некоторой задачи электронных цепей каждой переменной величине задачи ставится в соответствие определенная переменная величина электронной цепи. При этом основой построения такой модели является подобие исследуемой задачи и соответствующей ей электронной модели.

По своим вычислительным возможностям АВМ наиболее приспособлены для исследования объектов, динамика которых описывается дифференциальными уравнениями, а также алгебраическими и некоторыми другими типами уравнений. Следовательно, относительно класса решаемых

задач АВМ носят *специальный* характер, в отличие от *универсального* характера ЦВМ. Современные АВМ можно условно разбить на три класса: специального**,** общего и персонального назначений.

Специальные АВМ ориентированы на решение отдельных задач или одного класса задач, описываемых, как правило, обыкновенными дифференциальными уравнениями. АВМ этого типа имеют *фиксированную* или *коммутируемую* архитектуру. Класс специальных АВМ составляют в основном управляющие, бортовые и ориентированные на решение отдельных задач машины [4, 5].

АВМ *общего назначения* служат для решения широкого класса задач моделирования, и их архитектура, как правило, базируется на использовании коммутационного метода, а также методов *сеток* или *сплошных сред***.** Два последних метода позволяют решать класс задач, описываемых уравнениями в частных производных (задачи гидродинамики, теплопроводности, аэродинамики, моделирования атмосферы и др.). Метод *сеток* базируется на использовании электрических сеток с сосредоточенными в узлах параметрами. При решении таких задач дифференциальные уравнения предварительно преобразуются в систему линейных алгебраических уравнений по методу конечных разностей. Метод *сплошных сред* базируется на использовании электрических процессов в некоторой сплошной проводящей среде (электропроводная бумага, электролит и др.); он достаточно точен и прост, но его использование носит более узкий характер.

АВМ *общего* назначения условно делятся на три класса по их вычислительным возможностям решать задачи, описываемые дифференциальными уравнениями n-го порядка: *малые* ($n \le 10$; МН - 10М, МПТ - 9 и др.), *средние* ($10 \le n \le 20$; АВК - 32, PACER - 600 и др.) и *большие* ($n > 20$; АВК - 2 (5), ЭМУ - 200, PACER - 700 и др.).

В общем виде принципиальную структуру современной АВМ *общего* назначения можно представить в следующем виде (рис. 6). Кратко рассмотрим назначение основных компонентов АВМ.

Под *системой* понимается совокупность средств, предназначенных для обеспечения нормального функционирования АВМ. С*лужебная* система включает электропитание, терморегуляцию и вентиляцию, остальные системы являются *основными***.** Под *блоком* понимается устройство АВМ, имеющее определенное функциональное назначение. О*сновные блоки* служат для выполнения различных математических операций.

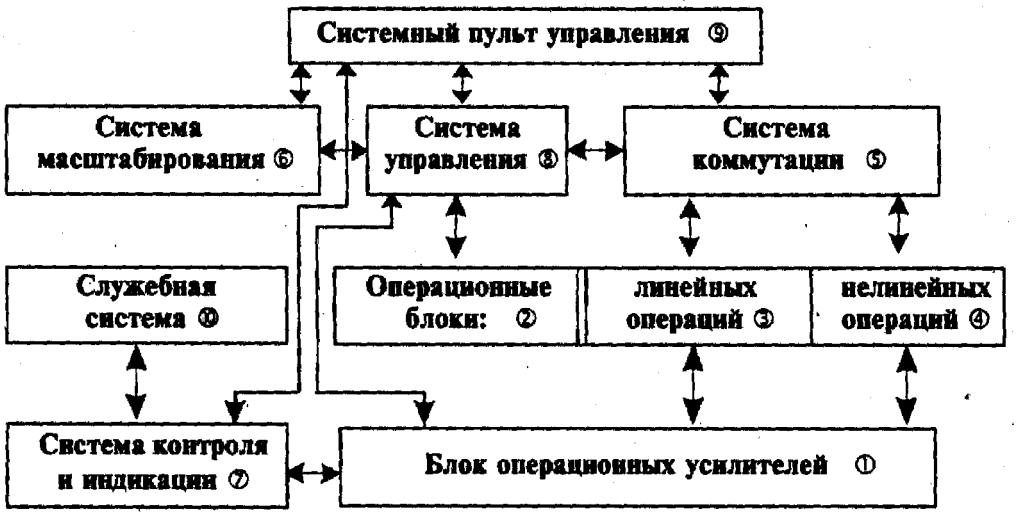


Рис. 6

Блок операционных усилителей содержит набор *операционных усилителей* (ОУ), являющихся усилителями постоянного тока с большим коэффициентом усиления. На базе ОУ в совокупности с *пассивными* элементами (резисторами, диодами, конденсаторами) выполняются практически все математические операции.

В состав *операционных блоков,* работающих совместно с ОУ, входят блоки *линейных* (суммирования, инвертирования, интегрирования) и *нелинейных* (умножения, деления, извлечения квадратного корня) *операций.* При решении задачи на АВМ операционные блоки посредством *системы коммутации* настраиваются на совместную работу с ОУ, образуя необходимые конфигурации для выполнения тех или иных функций или операций [5]. *Система масштабирования* служит для задания постоянных и/или переменных *масштабных* коэффициентов при подготовке АВМ к решению задачи; от их выбора зависит *точность* результата решения задачи. *Система коммутации* обеспечивает необходимую информационную связь между блоками АВМ посредством *кабелей,* коммутируемых программно. *Система контроля и индикации* обеспечивает контроль состояния ОУ и операционных блоков; результаты работы системы регистрируются, как правило, на осциллографах, самописцах и плоттерах. *Система управления* обеспечивает синхронизацию работы всех узлов АВМ. Пользователь через *пульт управления* инициирует ее работу в одном из двух режимов: *подготовки* и *решения* задачи. В *первом* режиме производится *настройка* всех нужных блоков АВМ и их *коммутация,* а во *втором* - *решение* задачи с заданными условиями и *вывод* результатов. С*лужебная система* поддерживает нужный терморежим для операционных блоков, а также *стабильность* напряжения и тока, необходимые для работы блоков. Так как изменение поступающих напряжений интерпретируется АВМ как изменение

значений *переменных* решаемой задачи, то к стабильности напряжения предъявляются особые требования.

Основные характеристики АВМ. В отличие от ЦВМ (точность вычислений которой определяется разрядностью) *точность* вычислений на АВМ определяется качеством изготовления элементной базы и основных узлов. Поэтому точность решения задач на АВМ находится в пределах (0.1 - 6)% или в числовом диапазоне (0.0001 - 1), т. е. на уровне большинства физических и инженерно-технических задач. Для целого класса задач производительность АВМ существенно превышает аналогичный показатель для ЦВМ. Это объясняется параллельным принципом решения задач на АВМ, когда результат решения получается мгновенно и одновременно во всех точках модели. Данная особенность делает весьма целесообразным использование АВМ в замкнутых системах автоматического регулирования и для решения задач в режиме реального времени.

Таблица 1

Показатель	**АВМ**	**ЦВМ**
Тип информации	Непрерывный	Дискретный
Изменение значений	Величиной напряжения	Числовым значением
Базовые операции	Арифметические, интегрирование	Арифметические
Принцип вычислений	Высокопараллельный	Последовательно-параллельный
Режим реального времени	Без ограничений	Ограниченные возможности
Динамическое изменение решаемой задачи	Посредством системы коммутации	В диалоговом режиме
Основные профессиональные требования к пользователю	Профессиональные знания + методика моделирования	Знание основ ПО, систем программирования, методов алгоритмизации
Уровень формализации задач	Ограничен моделью решаемой задачи	Высокий
Способность решения логических задач	Ограниченная	Высокая
Точность (φ) вычислений	$\varphi \le 10^{-4}$	$\varphi \approx 10^{-40}$
Диапазон представимых чисел	$1\text{-}10^{-4}$	Не уже 10^{-4}—10^{40}
Класс решаемых задач	Описываемые алгебро- и дифуравнениями	Любые задачи
Специальные функции	Ограниченный набор, низкая точность	Широкий класс
Уровень миниатюризации	Ограниченный	Высокий
Сферы применения	Ограниченные	Практически везде
Пользовательский интерфейс	Низкого уровня	Высокого уровня

АВМ способны выполнять лишь ограниченный набор *логических* операций (выбор минимакса, условные переходы и др.), существенно уступая ЦВМ в решении задач логического характера. Однако АВМ имеют существенные преимущества перед ЦВМ при использовании их в системах автоматического регулирования и управления, т.е. при создании широкого класса АСУТП. Процесс подготовки задачи для решения на АВМ существенно проще аналогичной работы для ЭВМ, ибо не требует специальных знаний по программированию и методам алгоритмизации.

Краткая сравнительная характеристика АВМ и ЦВМ по наиболее интересным показателям приведена в табл. 1.

Из сравнительного анализа следует, что АВМ (в отличие от ЦВМ) являются специализированной ВТ, наиболее приспособленной к работе в системах автоматического контроля и управления, а также при решении ряда важных задач математического и физического моделирования.

Дискретная вычислительная техника

По целому ряду причин именно *дискретное* представление информации определяет на сегодня лицо всей ВТ, основу которой составляют ЭВМ различных классов и типов.

Специальные ЭВМ (сЭВМ)

ЭВМ данного класса ориентированы на решение *специальных* вычислительных либо задач управления и характеризуются постоянством и периодичностью задач, решаемых в режиме реального времени.

Спектр сЭВМ лежит в весьма широком диапазоне - от простейших управляющих МП, встроенных в миниатюрную и робототехнику, до больших управляющих вычислительных комплексов, поддерживающих АСУ, АСУТП и др. В качестве сЭВМ можно рассматривать и различного типа электронные калькуляторы, предназначенные для решения несложных по логике и объему данных вычислительных задач.

В зависимости от класса сЭВМ обеспечиваются языками программирования различного уровня (от *микропрограммного* до *высокого)*. Однако они обладают ограниченными аппаратными и вычислительными ресурсами.

Дополнительную информацию по тенденциям развития и применения сЭВМ и их комплексов можно получить в [6].

Микропроцессорные ЭВМ и ПК

Создание в начале 70-х годов первых *универсальных микропроцессоров* (МП) можно считать началом эры микроЭВМ, а затем и *персональных компьютеров* (ПК). Первый из этой серии МП *Intel-4004* выполнял все функции центрального процессора ЭВМ *общего* назначения и в совокуп-

ности с четырьмя микросхемами (памяти, устройств управления и интерфейса ввода/вывода) представлял собой компьютер, не уступающий по мощности *большим* ЭВМ середины 50-х годов.

В классе микроЭВМ особое место занимают ПК, использование которых носит сугубо индивидуальный *(персональный)* характер и определяется, прежде всего, режимом эксплуатации, диапазоном решаемых задач, внешними устройствами.

Многие авторы рассматривают в качестве микроЭВМ все ЭВМ, базирующиеся на МП, не выделяя подклассов сЭВМ и ПК. В этом смысле все ЭВМ можно условно классифицировать на *четыре* больших типа: *микро (micro), мини (mini), общего назначения (mainframe)* и *супер (super)-ЭВМ*. Однако для пользователя классификация ЭВМ, как правило, носит весьма условный характер. Выделение среди микроЭВМ подкласса сЭВМ подчеркивает принципиально присущие данному типу ЭВМ *архитектуру* и *режим использования,* тем более что класс сЭВМ в *значительной* доле использует *специализированные* МП. В настоящее время под *микроЭВМ* понимается микропроцессорная ВТ, используемая двояко*:*

1) в качестве *универсального блока* обработки данных и/или управления, выпускаемого в виде одной БИС/СБИС и предназначенного для встраивания в различные специализированные системы контроля и управления и в другую технику для расширения ее функциональных и интеллектуальных возможностей;

2) в качестве малогабаритной ЭВМ персонального пользования (ПК), предназначенной для работы в интерактивном режиме, имеющей развитое ПО различного назначения и ориентированной на широкий круг пользователей.

Таким образом, первый способ использования микроЭВМ был рассмотрен в рамках сЭВМ, хотя класс сЭВМ не ограничивается только микроЭВМ. Аналогично, класс ПК не ограничивается только микроЭВМ.

Практически все современные *универсальные* (относительно набора команд) микроЭВМ отражают классическую неймановскую архитектуру, включающую следующие основные компоненты, представленные на примере схемы ПК (рис. 7). Данная схема поможет понять внутреннюю организацию некоторого типового ПК. Вместе с тем, ее элементы характерны для любой *микро-* и *мини*-ЭВМ, а также в значительной степени и для ЭВМ *общего назначения.* Так, *клавиатура* и *дисплей* составляют *консоль* ЭВМ и являются наиболее типичными устройствами ввода и отображения информации, обеспечивая *интерфейс* пользователя с ЭВМ. Большинство современных ЭВМ в своем составе имеют в качестве *внешней памяти* (ВП) накопители на различного типа магнитных носителях. ВП содержит программы и данные, которые не помещаются в оперативную память (ОП), и требуют длительного хранения. Для вывода и документирования данных и программ используются различного типа печатающие устройства (прин-

теры) и рисующие плоттеры, а для работы в системах телекоммуникации ЭВМ на основе *модемов* имеют возможность обмена информацией с удаленными телефонными абонентами.

Системная плата ПК (рис. 7) [3] содержит следующие основные компоненты: *тактовый генератор* (ТГ), *постоянное запоминающее устройство* (ПЗУ), *оперативную* память (ОП), *микропроцессор* (МП) {и, возможно, сопроцессор (МП*)}, *контроллеры* передачи данных, контроллеры ввода/вывода и порты ввода/вывода, а также шины управления, адресации и данных, образующие в совокупности *общую шину - системный* интерфейс (СИ) ПК.

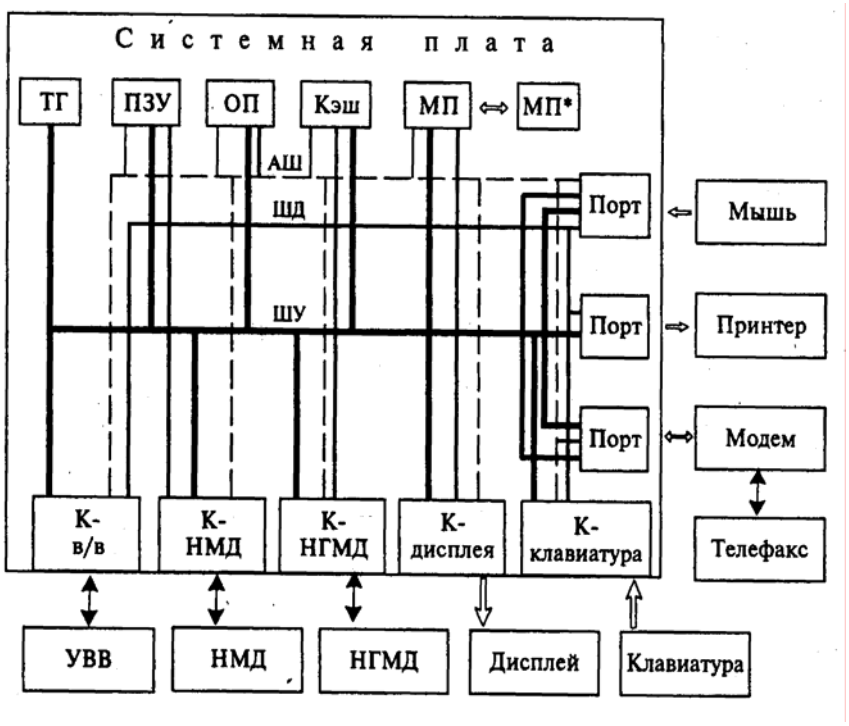


Рис. 7

К-ххх - контроллер ххх-устройства; ▬ (шина управления - ШУ); ——— (шина данных - ШД}; — — — (адресная шина - АШ/

Главной частью системной платы является МП (рис. 8) [3], управляющий работой всей системы узлов ПК и программой, описывающей алгоритм решаемой задачи. МП имеет сложную структуру, реализованную в виде системы электронных логических схем, в качестве основных его компонент можно выделить: арифметико-логическое устройство (АЛУ), устройство управления (УУ), систему прерываний (СПр), устройство управления общей шиной (УОШ) - системным интерфейсом - и специальные регистры.

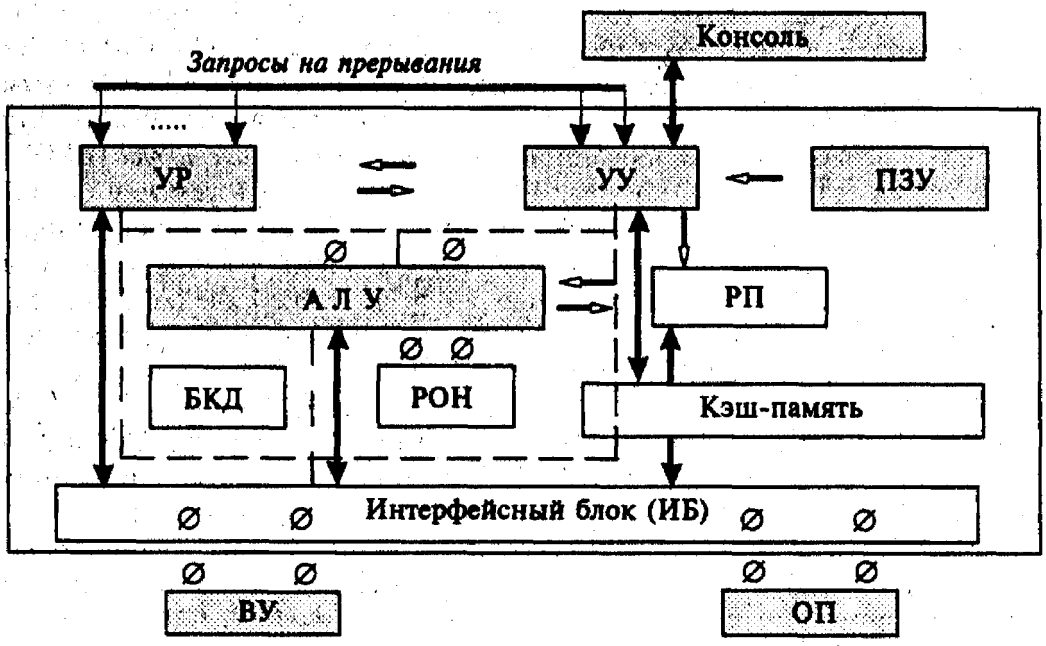


Рис. 8

УУ служит для управления работой всех узлов МП посредством выработки и передачи другим его компонентам управляющих и синхронизирующих сигналов; система синхронизации ПК базируется на кварцевом *тактовом генераторе* (ТГ). Для передачи синхро- и управляющих сигналов служит специальная *шина управления* (ШУ). АЛУ предназначено для выполнения *арифметических* и *логических* операций над данными и адресами памяти, хранящимися в *регистрах* общего назначения и специальных; регистры представляют собой *сверхоперативную* память, работающую со скоростью процессора. Вся обрабатываемая информация передается по *шине данных* (ШД), связывающей *все* основные узлы системной платы, кроме ТГ. *Система прерываний* (СПр) представляет собой *систему*, позволяющую прерывать работу МП практически в любой момент для немедленной обработки поступившего в данный момент *запроса* либо постановки его в очередь прерываний. После обработки прерывания СПр обеспечивает восстановление прерванного процесса обработки с точки, непосредственно следующей за точкой прерывания. Важный компонент МП составляет *устройство управления общей шиной - системным интерфейсом* (СИ), содержащим *шины управления, данных и адресации* (рис. 7).

Для расширения возможностей и повышения функциональных характеристик МП дополнительно на системную плату может устанавливаться специальный *сопроцессор* (МП*), как правило служащий для расширения набора команд ведущего МП. Например, математические сопроцессоры IBM-совместимых ПК расширяют возможности МП для вычислений с плавающей точкой; сопроцессоры могут расширять функции МП для работы в локальных сетях ПК (LAN-процессоры) и т.д.

ПЗУ содержит постоянную программу (сохраняемую даже при отключенном питании), которая служит для тестирования памяти, начальной загрузки ПК и инициации его *основных* блоков, находящихся в оперативном *(включенном)* состоянии. Содержимое ПЗУ *микропрограммным* способом формируется фирмой-изготовителем ПК и несет черты его индивидуальности. В случае специализации ПК в ПЗУ можно помещать наиболее часто используемое ПО.

Оперативная память служит для оперативного хранения программ и данных, сохраняемых только на период работы ПК. Объем ОП колеблется в достаточно широком диапазоне: от 64 Кбайт до 128 Мбайт и выше. Как правило, ОП имеет модульную структуру и может расширяться. МП имеет возможность прямого доступа к ОП, минуя систему ввода/вывода. Логическая организация ОП определяется функционирующим на ПК *системным* ПО (СПО). *Кэш-память (Кэш)* с малым временем доступа служит для временного хранения промежуточных результатов и содержимого наиболее часто используемых ячеек ОП и регистров МП. Объем *Кэш-памяти* зависит от модели ПК и для большинства моделей составляет 256 – 512 Кбайт.

Контроллер ввода/вывода является необязательным и обычно применяется в многопользовательских системах; он берет на себя управление некоторыми операциями по вводу/выводу, которые, при его отсутствии, выполняются самим МП. *Контроллеры* внешних устройств, как правило, используются для устройств быстрого обмена данными с ОП (НГМД, НМД, дисплей и др.), а также для обеспечения работы в *групповом* и *сетевом* режимах.

Порты ввода/вывода служат для обеспечения обмена информацией ПК с *внешними,* не очень быстрыми устройствами (клавиатура, мышь, телефонная сеть и др.). *Порты* бывают *входными, выходными* и *универсальными* (ввод/вывод), а также *последовательными* и *параллельными. Последовательный* порт ведет *побитный, а параллельны*й - *побайтный* обмен информацией; поэтому принтер подключается к *параллельному* порту, а мышь - *к последовательному.* Большинство современных ПК имеют один *параллельный* и два *последовательных* порта ввода/вывода. *Информация,* поступающая через порт, направляется сначала в МП, а затем в ОП, и наоборот.

Все узлы системной платы связаны *системным интерфейсом* (СИ) типа «*общая шина»,* организация которого зависит от модели и типа ПК и микроЭВМ. Он представляет собой систему линий передачи адресов, данных и управляющих сигналов различного типа и назначения. В настоящее время большинство стандартных СИ для микроЭВМ и ПК оптимизируются под конкретные типы МП и совершенствуются в направлении увеличения разрядности линий адреса и данных, линий запросов на прерывания и контроллеров, функциональных и диагностических возможностей. Наиболее распространенными стандартами СИ для микроЭВМ и ПК являются: *Multibus* (Intel), *VMEbus* (Motorola), *Q-bus* (DEC), *EISA* (Compaq) и др.

Малые (мини-) ЭВМ

Малые (мини-) ЭВМ составляют достаточно массовый класс, занимающий промежуточное положение между микроЭВМ (ПК) и ЭВМ общего назначения. *Мини-ЭВМ* имеют ОП объемом порядка от 100 Мбайт до нескольких гигабайт, приближаясь по вычислительным возможностям к ЭВМ общего назначения.

Типичную классическую архитектуру мини-ЭВМ представляют ЭВМ *PDP-серии* фирмы *DEC* (США) и отечественные СМ-серии [7], в частности *СМ-1700.*

Современное архитектурное развитие мини-ЭВМ строится на использовании идей мультипроцессорности, параллельной обработки и *RISC-подхода - (RISC - Reduced Instruction Set Computer* - ЭВМ с *сокращенным набором* команд). При этом большое внимание уделяется развитию *системных интерфейсов* (СИ). СИ обеспечивает объединение основных блоков ЭВМ в единую информационную систему с *центральным процессором* (ЦП), ОП, контроллерами и *портами* внешних устройств. По типу реализуемой системы ввода/вывода СИ подразделяются на четыре основных класса: *сосредоточенные* (Microbus, Multibus II), *локально-сосредоточенные* (Q-bus), *локальные* (Unibus, Общая Шина СМ ЭВМ и др.) и *локально-распределенные* (Bitbus, micro NOVA). Наиболее распространенным типом СИ для мини-ЭВМ является Unibus-стандарт (или ОШ СМ ЭВМ [7]), реализующий единую систему адресации ячеек ОП *и регистров* контроллеров *внешних устройств* (ВУ).

Достаточно детальный анализ вопросов эффективного применения мини-ЭВМ и используемого ими периферийного оборудования представлен в [8, 3].

ЭВМ общего назначения

Данный класс [3] обладает значительно большим быстродействием и вычислительными возможностями, чем мини-ЭВМ, и составляет основу крупных ВЦ и ВЦКП. Основная память ЭВМ общего назначения имеет объем от сотен мегабайт до гигабайт, имея возможности для наращивания. Их производительность измеряется десятками и сотнями миллионов операций в секунду, и они могут поддерживать работу с тысячами удаленных терминалов и/или рабочих станций.

По производительности модели серий ЭВМ общего назначения принято делить на *младшие, средние* и *старшие.* В настоящее время *старшие* модели имеют производительность свыше 10 млн. оп/с, предельно допустимые для серии объемы ОП и большое число автономных систем ввода/вывода.

В качестве типичных примеров ЭВМ общего назначения (в дальнейшем именуемых просто ЭВМ) можно назвать серии *IBM/360, IBM/370* (США) и *IBM*-совместимые отечественные серии *ЕС ЭВМ* [9].

Универсальность применения ЭВМ определяют следующие характерные черты: *универсальность, совместимость, развитое* ПО, *агрегатность* технических средств, высокая *технологичность*. Под *универсальностью* понимается возможность эффективно решать задачи различных классов и типов из всех областей человеческой деятельности. *Совместимость* реализуется на аппаратно-программном уровне и предполагает наличие единого системного и прикладного ПО, совместимого снизу вверх для всех моделей одной и той же серии ЭВМ. При этом СПО ЭВМ характеризуется наличием развитых операционных систем, являющихся программным расширением аппаратных средств ЭВМ. Прикладное ПО ориентировано на самый широкий круг приложений. Агрегатный принцип организации технических средств, стандартный интерфейс ввода/вывода, позволяющий подключать различные периферийные устройства широкой номенклатуры, совместно с развитым ПО позволяют создавать разнообразные вычислительные комплексы, наиболее отвечающие конкретным приложениям.

СуперЭВМ

СуперЭВМ занимают самую правую позицию в спектре современной ЦВТ и обладают производительностью, достигающей 10^{11} оп/с и выше. Такие вычислительные системы могут не только удовлетворительно решать сложнейшие научно-технические задачи, требующие огромного объема вычислений, но и обеспечивать работу более чем с 10 000 отдельных рабочих станций, для чего им требуются в качестве координатора системы ввода/вывода специальные мини-ЭВМ или ЭВМ общего назначения. Типичными областями применения суперЭВМ являются: научные исследования, прогнозирование погоды, проектирование авиационной и космической техники, ядерные исследования и другие области, требующие быстрой обработки очень большого количества данных. Определенную картину по использованию *суперЭВМ* дает сводный анализ по США [3]: оборонные проекты (45%), нефтяные компании (18%), университеты (13%), космические исследования (10%) и другие (14%).

Максимальная производительность (а целый ряд задач уже сейчас требует производительности не менее 10^{12} оп/с) может быть получена только в рамках не-неймановской параллельной архитектуры ЦВТ. Аппаратно современный параллелизм ЭВМ поддерживается на четырех основных уровнях: 1) многомашинном, 2) мультипроцессорном, 3) однопроцессорном с несколькими исполнительными устройствами и 4) конвейеризацией обработки данных.

Примером организации мультипроцессорной суперЭВМ может служить отечественная Эльбрус-3.

СуперЭВМ Эльбрус-3 содержит до 16 ЦП, исполнительные устройства, которые работают в максимально допустимом режиме распараллеливания, обеспечивая быстродействие ВС порядка 5 млрд. оп/с. Каждый из процессоров суперЭВМ обеспечивает эффективную трансляцию и выполнение программ за счет аппаратной реализации наиболее массовых алгоритмов языков высокого уровня и операционной системы; мультипрограммный режим с аппаратной защитой данных пользователя и операционной системы.

Базовое СПО суперЭВМ *Эльбрус* базируется на машинном языке *ЭЛЬ-76* высокого уровня и ОС *Эльбрус*. Дополнительно реализованы ОС *UNIX и* языки программирования *Prolog* и *ADA*.

Более детально с организацией отечественных суперЭВМ семейства Эльбрус можно ознакомиться в [10]. Из наиболее известных зарубежных мультипроцессорных суперЭВМ можно отметить такие серии, как: *Cray-3, Cray-4* фирмы *Cray Research; AFP* фирмы *CDC; FACOM, VP-200* фирмы *Fujitsu; HEP* фирмы *Denelcor; iPSC* фирмы *Intel*.

Представление информации в ЭВМ

Вопросы представления информации в ЭВМ включают два основных аспекта: используемая базовая *система счисления* (с.с.) и собственно представление числовой и алфавитно-цифровой информации. Цифровые элементы на основе *двоичной* с.с. *обеспечивают максимальную* производительность ЭВМ. Поэтому наиболее распространенными с.с. являются системы с $2^к$-основанием (к=1, 2, 4), это же относится и *к разрядности* ЭВМ, кратной двум $\{2^p \mid 3 \le p \le 7\}$.

Системы счисления (с.с.). Под с.с. понимается метод обозначения количества, являющийся *позиционным* и *непозиционным.* В *непозиционных* с.с. символы, обозначающие то или иное количество, не меняют своего значения в зависимости от места в изображении количества, например в *римской* с.с. Однако такие системы характеризуются сложными способами записи чисел и правилами выполнения арифметических операций. В *позиционных* с.с. значение символа зависит от *его места* в изображении количества. Для определения некоторой позиционной с.с. необходимо ввести *алфавит* $A=\{a_1, a_2, ..., a_n\}$, с помощью которого можно представлять любое количество. *Основанием* такой с.с. является число a_n+1. Чтобы не усложнять изложение, будем рассматривать алфавит A={0 ÷ 9, A, B, C, D, E, F}, обеспечивающий использование с.с. с *основаниями* ≤ 16. Изображение любого количества называется *числом,* а символы *A*-алфавита - *цифрами.* Тогда любое число N представимо как:

$$N_{(p)} = a_k p^k + a_{k-1} p^{k-1} + \ldots + a_0 p^0 + a_{-1} p^{-1} \ldots = \sum a_n p^n \quad k \le n \le (-m)\},$$

где p $(p=a_n+1)$ есть *основание* с.с. и $N_{(p)}$ - число N в с.с. с p-основанием. Существует бесконечное число позиционных с.с. Рассмотрим наиболее употребительные из них и вопрос перевода из одной системы в другую.

Пусть имеются две с.с. с основаниями p и d. Рассмотрим сначала перевод *целых чисел* из p-ичной с.с. в d-ичную. Любое целое число в этих с.с. можно представить соответственно в виде:

$$N_{(p)} = a_k p^k + a_{k-1} p^{k-1} + \ldots + a_1 p^1 + a_0 p^0 \ldots = \sum a_n p^n \;\; a_n \in \{0,\ldots, p-1\} \;\; (n-0,\ldots,k)$$

$$N_{(d)} = b_t d^t + b_{t-1} d^{t-1} + \ldots + b_1 d^1 + b_0 d^0 \ldots = \sum b_n d^n \quad b_n \in \{0,\ldots, d-1\} \;\; (n-0,\ldots,t)\,.$$

(A) Алгоритм перевода состоит в следующем. На первом шаге делим в с.с. с p-основанием $(p$-с.с.$)$ число $N_{(p)}$ на d-основание: $N_{(p)}/d=N_{1(p)}+a_0$; на втором шаге делим число $N_{1(p)}$ на d-основание: $N_{1\{p)}/d=N_{2(p)}+a_1$ и т. д. Последний результат деления будет: $N_{m-1(p)}/d=N_{m(p)}+a_{m-1}$ при условии $N_{m(p)}, a_{m-1} < d$. Тогда из представлений чисел $N_{(p)}$ и $N_{(d)}$ нетрудно заметить, что число $N_{(d)}$ должно иметь вид:

$$N_{(d)} = N_{m(p)} d^m + a_{m-1} d^{m-1} + \ldots + a_1 d^1 + a_0 d^0;$$

$$N_{m(p)}, a_k \in \{0,\ldots, d-1\} \quad (k = 0,\ldots, d-1).$$

Таким образом, $N_{(p)} \equiv N_{(d)}$.

Рассмотрим теперь перевод чисел, меньших 1. В этом случае числа $N_{(p)}$ и $N_{(d)}$ имеют следующий вид:

$$N_{(p)} = 0.a_{-1} p^{-1} + a_{-2} p^{-2} + a_{-3} p^{-3} + \ldots = \sum a_n p^n \;;\; a_n \in \{0,\ldots, p-1\},$$

$$N_{(d)} = 0.\beta_{-1} d^{-1} + \beta_{-2} d^{-2} + \beta_{-3} d^{-3} + \ldots = \sum \beta_n d^n;\; \beta_n \in \{0,\ldots, d-1\};(n = -1,-2,-3,\ldots).$$

(B) Алгоритм перевода состоит в следующем. На первом шаге умножаем в p-с.с. число $N_{(p)}$ на d-основание: $N_{(p)} \cdot d = \beta_1 \cdot N_{1(p)}$; на втором шаге умножаем число $N_{1(p)}$ на d-основание: $N_{1(p)} \cdot d = \beta_2 \cdot N_{2(p)}$ и т.д.

Из представлений чисел $N_{(p)}$ и $N_{(d)}$ нетрудно заметить, что число $N_{(d)}$ должно иметь вид:

$$N_{(d)} = 0.\,\beta_1 d^{-1} + \beta_2 d^{-2} + \ldots = \sum \beta_n d^n;\; \beta_n \in \{0,\ldots, d-1\};(n = 1,2,3,\ldots).$$

Для перевода *смешанных* чисел следует переводить *отдельно* их *целые* и *дробные* части согласно соответственно алгоритмам **(A)** и **(B)**. Процедуру перевода чисел из p-с.с. в d-с.с. будем называть $(p \to d)$-переводом.

Вопросы перевода из одной с.с. в другую играют весьма важную роль для программистов, использующих языки программирования низкого уровня (микропрограммные, машинные, ассемблеры и др.).

Приведем пример перевода числа из одной с.с. в другую:

$(10 \to 8)$-перевод: $1942.55_{(10)} = 3626.4(3146)_{(8)}$

(A)

```
 1   9   4   2 | 8
-1   6         |-----
     3   4     | 2   4   2 | 8
    -3   2     |-2   4   0 |-----
         2   2 |       2   | 3   0 | 8
        -1   6 |     -2  4 |-------| 3
             6 |           |   6
               |           |
```

(B)

0.55	0.40	0.20	0.60	0.80	0.40
x 8	x 8	x 8	x 8	x 8	x 8
4.40	3.20	1.60	4.80	6.40	3.20

В процессе умножения обнаружен период, равный $(3146)_{(8)}$.

Современная ВТ использует в основном 2-с.с. для хранения команд и информации при выполнении арифметических и логических операций. Системы счисления с основаниями 8 и 16 используются для более компактной записи команд и содержимого памяти, а также при некоторых специальных форматах представления чисел.

Представим простейшие операции *двоичной* арифметики.

Сложение (вычитание) бинарных чисел производится поразрядно с переносом (заниманием) единицы в старший (старшем) разряд(е), например:

```
  1001110100111.1001          11011001.011
+ 10001011.1001             -  100110.100
------------------          ---------------
  1010000110010.1111          10110010.111
```

Как и в случае десятичных чисел, *умножение* бинарных чисел производится путем поразрядного умножения с последующим суммированием; положение десятичной точки определяется также аналогично. Сказанное с очевидными изменениями относится и к случаю деления бинарных чисел. Следующие примеры иллюстрируют сказанное:

```
    1 0 1 1 1                1 0 0 0 1 0 0 1 1 0 | 1 1 0 0 1
  x 1 0 1 0 1                  1 1 0 0 1         | 1 0 1 1 0
  ---------                  -------------       |---------
    1 0 1 1 1                    1 0 0 1 0 1
  0 0 0 0 0                      1 1 0 0 1
---------                      -------------
  1 0 1 1 1                        1 1 0 0 1
0 0 0 0 0                          1 1 0 0 1
---------                        -------------
1 0 1 1 1                                  0
---------
1 1 1 1 0 0 0 1 1
```

Нетрудно заметить, что операция *умножения* бинарных чисел сводится к операциям *сдвига* и поразрядного *сложения* по (mod 2); деление использует последовательное *вычитание.* Простота правил бинарных сложения, вычитания и умножения позволяет упрощать схемы арифметических устройств ЭВМ.

Прямой, обратный и дополнительный коды. *С* целью упрощения выполнения арифметических операций для представления числовой ин-

формации применяются специальные коды, позволяющие упрощать определение знака результата операции и заменять вычитание сложением. При этом облегчается выработка признаков переполнения разрядной сетки. В дальнейшем все бинарные числа рассматриваются относительно некоторого *n*-разрядного регистра.

Прямой код любого бинарного *N*-числа определяется следующей формулой:

$$N_{\text{пр}} = \begin{cases} N, & \text{если } N \geq 0 \\ |N| + \alpha, & \text{если } N < 0 \end{cases},$$

где $\alpha=1$ и $\alpha=2^{n-1}$ соответственно для *дробных* и *целых* чисел; диапазон представления чисел *прямым* кодом есть $0 \leq N < \alpha$. Из определения следует, что числа $N \geq 0$ представляются кодами $0 \leq N_{пр} < \alpha$, а числа $N < 0$ - кодами $\alpha \leq N_{\text{пр}} < 2\alpha$. Признаком знака является наличие нуля (+) или единицы (-) в старшем разряде регистра, называемом знаковым. Например, в 12-разрядном регистре числа N=1996 и N′= -54 будут иметь соответственно коды $N_{пр}$=0 | 11111001100 и $N_{пр}$=1 | 00000110110. Нетрудно убедиться: сложение в *прямом* коде чисел с *одинаковыми* знаками весьма просто - числа складываются, а содержимое знакового разряда не меняется. В случае чисел *с разными* знаками операция сложения в *прямом* коде значительно сложнее, поэтому для представления чисел со знаком используются *обратный* и *дополнительный* бинарные коды, определяемые для произвольного бинарного *N*-числа соответственно как:

$$N_{\text{обр}} = \begin{cases} N, & \text{если } N \geq 0 \\ \beta - |N|, & \text{если } N < 0 \end{cases} \text{ и } N_{\text{доп}} = \begin{cases} N, & \text{если } N \geq 0 \\ \gamma - |N|, & \text{если } N < 0 \end{cases},$$

где $\beta=2-2^{-(n-1)}$, $\gamma=2$ и $\beta=2^n-1$, $\gamma=2^n$ соответственно для *дробных* и *целых* чисел. Диапазон представляемых *обратным* кодом чисел такой же, как и в случае *прямого* кода. В случае *дополнительного* кода он зависит от знака числа: для положительных он составляет $0 \leq N < \alpha$, для отрицательных - $0 < |N| \leq \alpha$, где α - величина из определения.

При использовании двух последних кодов операция сложения чисел с различными знаками сводится к операции сложения при помощи *обратного* и *дополнительного* кодов. Например, числа N=1996 и N'= -54, представленные в бинарных *прямом, обратном* и *дополнительном* кодах согласно определению принимают соответственно следующий вид:

$N_{\text{пр}}$=	0|11111001100	$N_{\text{обр}}$=	0|11111001100
$N_{\text{пр}}$=	1|00000110110	+$N_{\text{обр}}$=	1|11111001001
			0|11110010110
$N_{\text{доп}}$=	0|11111001100		1996
+$N_{\text{доп}}$=	1|11111001010	+	-54
	0|11110010110		1942

Из примера хорошо видно, что для *положительного* бинарного числа значения всех трех кодов совпадают, тогда как *обратный* код отрицательного целого числа получается из его *прямого* кода путем *инверсии* всех его цифровых разрядов, а *дополнительный* - из *обратного* путем добавления к младшему разряду единицы. При сложении бинарных чисел, представленных в *обратном (дополнительном)* коде, производится сложение по "mod 2" всех *n* разрядов регистра, включая знаковый. При этом в случае возникновения *переноса* в знаковом разряде 1 *добавляется (не добавляется)* к младшему разряду *обратного (дополнительного)* кода. На приведенном примере это видно. Из определений кодов легко получить следующие соотношения между ними для произвольного бинарного N-числа соответственно *целого* и *дробного:*

Целое: $N_{\text{пр}} + N_{\text{доп}} = 3 \times 2^{n-1}$; $N_{\text{пр}} + N_{\text{обр}} = 3 \times 2^{n-1} - 1$; $N_{\text{доп}} = N_{\text{обр}} + 1$.

Дробное: $N_{\text{пр}} + N_{\text{доп}} = 3$; $N_{\text{пр}} + N_{\text{обр}} = 3 - 2^{-(n-1)} - 1$; $N_{\text{доп}} = N_{\text{обр}} + 2^{-(n-1)}$.

Следовательно, *дополнительный код* отрицательного бинарного числа получается путем прибавления 1 к *младшему* разряду *обратного* кода. При работе с *прямым* и *обратным* кодами может возникать *отрицательный* нуль, отсутствующий в *дополнительном* коде. Подробнее о принципах представления бинарных чисел можно узнать в [11, 12].

Формы представления чисел. Любая информация в ЭВМ представляется в виде *бинарных* (двоичных) кодов *фиксированной* или *переменной* длины. Наименьшие элементы бинарного кода *(единицы информации),* принимающие значения 0 или 1, называются *битами,* или *разрядами.* В качестве основной единицы информации в современных ЭВМ служит *байт,* состоящий из 8 бит.

В ЭВМ применяются *две формы* представления чисел: с *фиксированной* и *плавающей* точкой. В первом случае положение точки *фиксируется* в определенном месте относительно разрядов числа, как правило, перед *старшим* или после *младшего;* в *первом* случае представляются числа $|N| < 1$, во *втором* - только *целые* числа (рис. 9).

2^{n-1}	2^{n-2}	…	…	…	2^j	…	…	2^2	2^1	2^0
знак	…	Р	А	З	Р	Я	Д	Ы	…	…
n-1	n-2	…	…	…	j	…	…	2	1	0

Рис. 9

Нумерация бинарных разрядов *(битов)* в суперЭВМ и ЭВМ общего назначения ведется *слева направо,* в ЭВМ остальных типов - наоборот. *Знаковый* разряд является, как правило, крайним слева. На рис. 9 показан общий формат для чисел с точкой, зафиксированной после младшего разряда. В этом формате представимы целые числа с точностью до $2^{(n-1)}$. В случае использования прямого кода диапазон представления чисел состав-

ляет $1 \le |N| \le 2^{(n-1)} - 1$; дополнительный код позволяет использовать числа в диапазоне $-2^{n-1} \le N \le 2^{(n-1)} - 1$, что при *n=32* примерно соответствует диапазону десятичных целых чисел $1 \le |N| \le 10^9$.

Если точка фиксируется справа от младшего разряда, то регистром целых чисел *со знаком* можно представлять *нуль, положительные* и *отрицательные* целые бинарные числа. В зависимости от модели ЭВМ используются два формата представления целых чисел с фиксированной точкой: *со знаком* и *без знака*; в последнем случае все разряды регистра служат для представления модуля числа.

В *универсальных* ЭВМ основной является форма представления чисел *с плавающей точкой*, не требующая масштабирования данных.

В общем случае представление *N*-числа в форме с плавающей точкой имеет следующий вид: $N=A^pM$, где *M - мантисса; A - основание* характеристики и *p* - ее *порядок.* Как правило, величина A^p представляет целую степень двойки. *Мантисса* (*M;* является дробью со знаком) и *порядок (p;* целое со знаком) представляются в *A*-с.с. Знак *N*-числа совпадает со знаком *M*-мантиссы; *p*-порядок определяет положение точки в представлении *N*-числа. На рис. 10 представлен общий формат числа с плавающей точкой:

Знак числа	П	О	Р	Я	Д	О	К	М	А	Н	Т	И	С	С	А
(n+k+1)	*(n+k)*	…	…	…	…	…	*(n+1)*	*n*	…	…	…	…	*2*	*1*	*0*

Рис. 10

В таком формате, как правило, крайний левый бит определяет *знак мантиссы,* следующая за ним группа битов - *порядок со знаком* и остальные биты - *модуль мантиссы.* Действия над числами в форме с плавающей точкой требуют выполнения операций как над мантиссой, так и над порядком.

Так как под *мантиссу* отводится фиксированное число битов, то для получения максимальной точности используются *нормализованные* числа, для которых выполняется условие $A^{-1} \le |M| < 1$. В некоторых ЭВМ используется другое условие *нормализации* **-** $1 \le |M| < A$, т.е. старший бит мантиссы в *A*-с.с. отличен от нуля.

Диапазон представимых чисел в форме с плавающей точкой зависит от *A*-с.с. и числа битов, отведенных под *p*-порядок, тогда как *точность* вычислений определяется числом битов *M*-мантиссы. С увеличением разрядности мантиссы растет точность вычислений, но уменьшается скорость выполнения арифметических операций.

В мини- и микроЭВМ представление чисел в форме с плавающей точкой имеет особенность: используется 2-с.с. (*A=2*) и два формата - *короткий* (32 бита) и *длинный* (64 бита). Под *p*-порядок отводится 8 бит, а под *M-мантиссу* - 23 бита *(короткий)* и 55 бит *(длинный)*. Числа пред-

ставляются только в *нормализованном* виде, когда *старший* бит мантиссы является единичным (поэтому старший бит мантиссы не фиксируется, его единичное значение подразумевается**).** Следовательно, обеспечивается фактическая точность представления чисел, соответствующая мантиссам с 24 и 56 битами.

Представление символьной информации и десятичных чисел. Наряду с числовой, ЭВМ обрабатывают и *символьную* (алфавитно-цифровую) информацию. *Совокупность* всех символов, используемых на конкретной ЭВМ, образует ее *алфавит**,*** и для их *бинарного* кодирования применяется какой-либо метод, когда каждому символу ставится в соответствие отдельное бинарное слово-код. При выборе метода кодирования руководствуются объемом и способами обработки символьной информации. Так как многие типы информации содержат в значительном объеме *цифровую* информацию, то применяются две системы кодирования: *символьной* информации и *десятичных* чисел. Эволюция электронной информационной технологии вызвала появление *8-битовой* единицы информации - *байта**,*** биты в котором нумеруются, как правило, справа налево 0, ..., 7.

Наибольшее распространение имеет представление *символьной* информации с помощью байтов: *байт* $\leftrightarrow$ *символ**.*** Так как байт содержит 8 битов, то таким способом можно закодировать максимум 256 символов. Более того, байт вмещает две десятеричные или шестнадцатеричные цифры, что очень важно при представлении *десятичных* чисел. Современные ЭВМ и дискретные системы управления производят *пословную* обработку информации, где *слово* состоит из последовательности байтов. Слова бывают *фиксированной* (1, 2, 4, 8, 16, ... байтов) и *переменной* длины. Длина используемых ЭВМ слов во многом определяет ее производительность. В дальнейшем *объем* символьной информации будем измерять в единицах: бит, 1 байт (б) $\equiv$ 8 бит, 1 килобайт (Кбайт) $\equiv$ 1024 байт, 1 мегабайт (Мбайт) $\equiv 1024^2$ байт, 1 гигабайт (Гбайт) $\equiv 1024^3$ байт. Например, объем одного печатного листа текста (40 000 символов) составляет $\approx$ 39.1 Кбайт.

Каждая ЭВМ имеет свою *внутреннюю* таблицу кодов, реализующую тот или иной метод бинарного кодирования символьной информации в *оперативной* и *внешней* памяти. В настоящее время наиболее распространенными являются следующие коды - *американский стандартный код обмена информацией* (ASCII), при необходимости расширяемый за счет символов национальных алфавитов (например, кириллицы), и *двоичный код* (ДКОИ) обмена информацией. Каждый пользователь легко может уточнить используемый его ЭВМ *внутренний* код бинарного представления символьной информации. С другой стороны, программные системы могут использовать и собственные системы кодирования символьной информации; существуют средства для перекодирования информации.

Символьная информация представляется словами *переменной* длины, содержащими нужное число байтов. Для упрощения обработки символь-

ной информации бинарному коду каждого символа присваивается *вес,* представляющий собой десятичный *эквивалент* бинарного кода-числа. Тогда имеется возможность легко производить на основе *весов* кодов, например, сортировку *символьной* информации. Например, в *ASCII*-коде нижеследующие заглавные латинские буквы имеют соответственно следующие коды-представления и веса:

Символ	**16-с.с.**	**2-с.с.**	**Вес**
пробел	20	0010|0000	32
5	35	0011|0101	53
A	41	0100|0001	65
G	47	1000|1110	71
K	4B	0100|1011	75
S	53	0101|0011	83
V	56	0101|0110	86

Выражение “Tallinn 1996" в шестнадцатеричной системе счисления имеет 12-байтовое представление следующего вида:

54	61	6C	6C	69	6E	6E	20	31	39	39	36

В отличие от *символьной* для *десятичной* цифровой информации при байтовой организации в целях экономии памяти и удобства обработки используются специальные форматы кодирования десятичных чисел - *зонный* и *упакованный.* При этом *десятичные* числа рассматриваются как десятичные со знаком, имеющие переменную длину; для представления отрицательных чисел используется *прямой* код.

Цифры 0, ..., 9 могут представляться в бинарно-десятичном *8421-коде, в* котором каждая цифра представляется ее 4-битным бинарным эквивалентом, а шесть сочетаний <1010>...<1111> служат для кодирования знаков и служебных символов. Код *8421* удобен для выполнения машинных преобразований из десятичной с.с. в двоичную с.с., и наоборот, но его использование имеет свои особенности [11].

В заключение следует отметить, что рассмотренные способы представления информации относятся к ЭВМ классической архитектуры [13].

Структура памяти

Оперативная память представляет собой электронное устройство, включающее в себя большое количество запоминающих элементов для записи бит информации, а также схем управления ими. Биты группируются в байты и слова. У каждого слова и каждого байта есть адрес - номер, по ко-

торому к ним обращается ЦП. Все байты оперативной памяти нумеруются начиная с нуля (рис. 11*а*). Слова нумеруются четными числами (рис. 11*б*), причем четные байты считаются младшими в слове и занимают разряды 0,. ..,7 слова (рис. 11*в*); нечетные байты слова занимают старшие разряды 8,. ..15 (рис. 11*г*). В микроЭВМ предусмотрена возможность обращаться как к словам, так и к байтам.

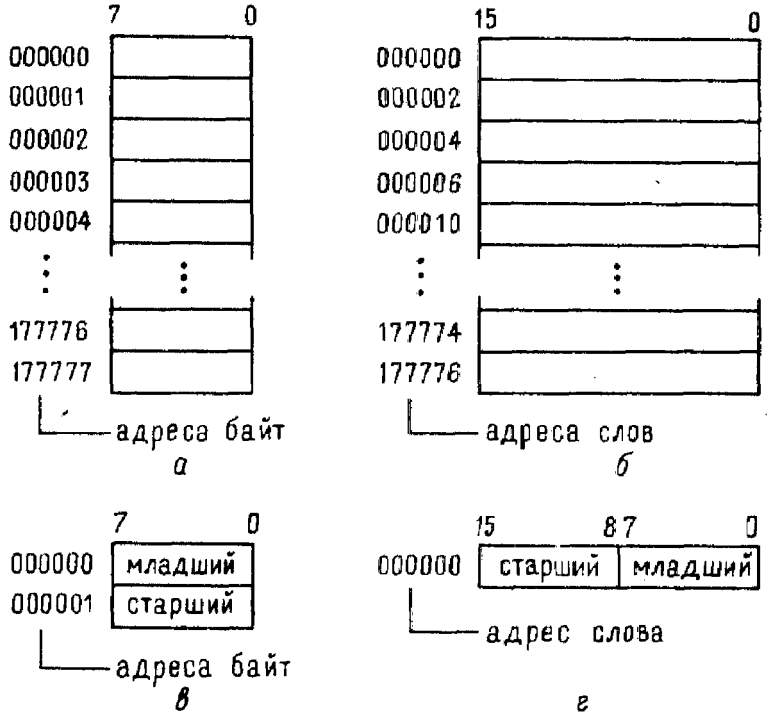


Рис. 11

Разрядность адреса памяти определяет максимальную область памяти или адресное пространство памяти, доступное для процессора. Так, 16-разрядный адрес микроЭВМ позволяет обращаться к 64К байт или к 32К слов (буква К используется для обозначения числа, равного $1024(10) = 2^{10}$) [14].

В настоящее время в качестве оперативной памяти микроЭВМ чаще всего используются *динамические запоминающие устройства (ЗУ). Динамические ЗУ* позволяют записывать и считывать данные по любому адресу в произвольный момент времени и относятся к RAM-памяти (Random Access Memory). Динамическим ЗУ присущ существенный недостаток**:** для сохранения содержимого необходимо периодически считывать и записывать элементы памяти, этот процесс называется регенерацией памяти. В зависимости от конкретного исполнения ЗУ доступ к нему может блокироваться на время регенерации или быть «прозрачным» для ЦП. При работе ЭВМ совместно с экспериментальной установкой (ON LINE) это необходимо учитывать. Тем не менее три достоинства: высокая степень интеграции, малое потребление, малая удельная стоимость бита информации - определяют широкое применение динамических ЗУ в ЭВМ.

Режимы работы вычислительных систем

Аппаратные средства ЭВМ совместно с программным обеспечением образуют вычислительную систему (ВС). В зависимости от класса ЭВМ и вида операционной системы ВС могут работать в однопрограммном или мультипрограммном режимах.

В *однопрограммном режиме* работы в памяти ЭВМ находится и выполняется только одна программа. Такой режим обычно характерен для микроЭВМ и персональных ЭВМ, т.е. для ЭВМ индивидуального пользования.

В *мультипрограммном режиме* (многопрограммном) режиме работы в памяти ЭВМ находится несколько программ, которые выполняются частично или полностью между переходами процессора от одной задачи к другой в зависимости от ситуации, складывающейся в системе. В мультипрограммном режиме более эффективно используются машинное время и оперативная память, так как при возникновении в выполняемой задаче каких-либо ситуаций, требующих перехода процессора в режим ожидания, процессор переключается на другую задачу и выполняет ее до тех пор, пока в ней также не возникнет подобная ситуация, и т.д. При реализации мультипрограммного режима требуется определить очередность переключения и выбирать моменты переключения, чтобы эффективность использования памяти и машинного времени была максимальной. Мультипрограммный режим обеспечивается аппаратными средствами ЭВМ и средствами операционной системы. Он характерен для персональных ЭВМ на процессорах типа 386, 486, Pentium, одновременно следящих за выполнением нескольких задач и повышающих тем самым эффективность работы пользователей.

При мультипрограммном режиме работы ВС можно выделить *режим коллективного доступа*, когда каждый пользователь ставит свою задачу на выполнение в произвольный момент времени, т.е. для каждого пользователя в такой ВС как бы реализуется режим индивидуального пользования. Это осуществляется обычно с помощью квантования машинного времени, когда каждой задаче, находящейся в оперативной памяти ЭВМ, выделяется квант времени. По окончании кванта времени процессор переключается на другую задачу или продолжает выполнение прерванной в зависимости от ситуации в ВС. Вычислительные системы, обеспечивающие коллективный доступ с квантованием машинного времени, называют *ВС с разделением времени.*

Сети ЭВМ и средства телекоммуникационного доступа

Тенденции развития средств автоматизации эксперимента отслеживают тенденции развития вычислительной техники. На начальном этапе, когда число компьютеров во всем мире было невелико, возможности их ограничены, а стоимость нередко превышала стоимость обслуживаемой установки, системы автоматизации развивались по принципу централизации вычислительных мощностей. Схематически такая система изображена на рис. 12. В ней центральный компьютер обслуживает одну или несколько экспериментальных установок.

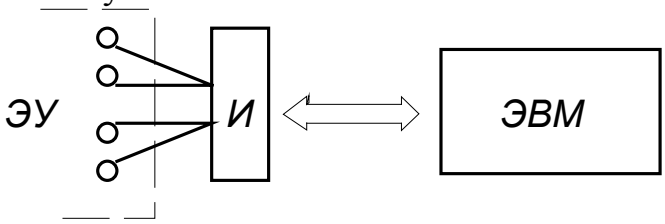


Рис. 12

При стремительном развитии микропроцессорной техники относительная стоимость компьютера стала резко падать. Появились “интеллектуальные” приборы и устройства со встроенными микропроцессорными системами. Они позволяют автономно выполнять ряд функций в системе автоматизации эксперимента. Построенная с их применением система автоматизации эксперимента имеет структуру, приведенную на рис. 13. В нее входят компьютеры и “интеллектуальные” приборы, каждый из которых выполняет свою функцию сбора, обработки данных и управления экспериментом.

Контроль за работой отдельных узлов децентрализованной системы автоматизации осуществляет, как правило, один из компьютеров, наделенный соответствующими “полномочиями”. Компьютеры системы автоматизации обмениваются между собой данными и командами управления при помощи аппаратуры связи.

Для современного этапа развития средств вычислительной техники характерно использование сравнительно дешевых персональных ЭВМ, обладающих достаточно большими вычислительными возможностями. Поэтому естествен переход к распределенным системам обработки информации на базе многопроцессорных и многомашинных ВС (ММВС), а также сетей ЭВМ.

ММВС позволяют получить высокую производительность. Их применение целесообразно при необходимости интенсивного обмена большими массивами данных. Объединение ЭВМ в ММВС производят с помощью стандартных интерфейсов.

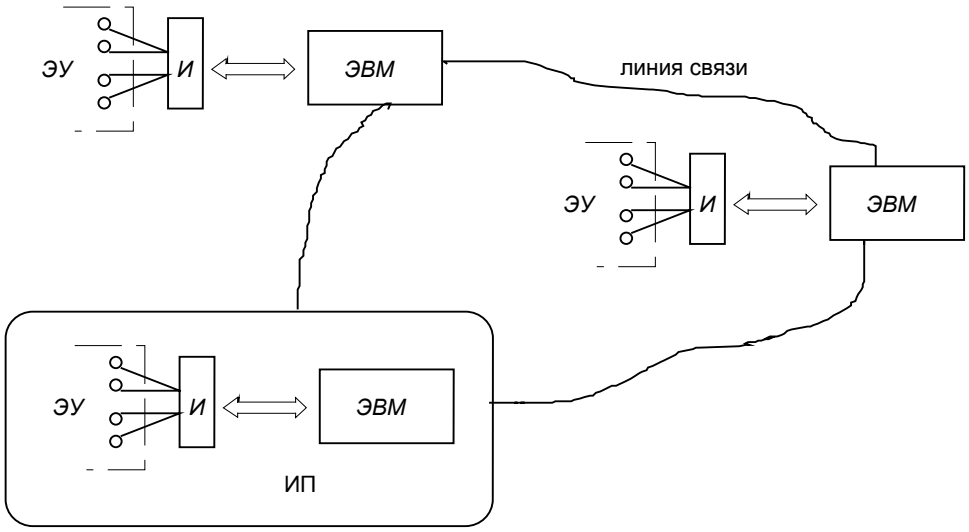


Рис. 13

ЭУ - экспериментальная установка (или отдельные ее узлы), И - интерфейсы связи ЭВМ с экспериментальной установкой, ИП - интеллектуальный прибор. Датчики и узлы управления условно изображены кружками.

Под вычислительной сетью (сетью ЭВМ) понимают объединение достаточно большого числа независимых ВС, удаленных друг от друга на расстояние от нескольких метров до нескольких тысяч километров и связанных специальным каналом передачи данных, с целью коллективного использования аппаратных, программных и информационных ресурсов. Наиболее перспективными при проектировании и использовании АСНИ являются *локальные* вычислительные сети (ЛВС) с расстояниями между отдельными ЭВМ (узлами сети), не превышающими 2.5 ... 3.0 км [15].

Глобальные вычислительные сети объединяют ЭВМ в пределах больших географических регионов, располагают значительными вычислительными мощностями и позволяют получить доступ к информационным ресурсам всей планеты (например, сеть InterNet).

Классификация вычислительных сетей возможна по различным признакам. *По типу ЭВМ*, объединенных в сеть, различают *однородные* вычислительные сети, объединяющие программно-совместимые ЭВМ, и *неоднородные*.

По распределению функций управления сетью могут быть *централизованные* вычислительные сети, управляемые центральной ЭВМ, и *децентрализованные*.

По принципу передачи данных между узлами различают сети ЭВМ:

- с некоммутируемыми каналами передачи данных, используемые для передачи больших объемов информации с малым временем установления связи между ЭВМ;

- с коммутируемыми каналами передачи данных, имеющие специальные переключатели каналов связи;

- *с коммутацией сообщений*;

- *с коммутацией пакетов*, в которых все сообщения разбиваются на части - пакеты, передаваемые по отдельности и собираемые в узле назначения в единое сообщение;

- сети ЭВМ *со смешанной коммутацией*.

Значительное влияние на характеристики вычислительной сети оказывает ее конфигурация, или структура.

Топологии локальных сетей

Сегодня наиболее часто используются три топологии локальных сетей [17, 18] (ЛВС): «звезда», «кольцо» и «шина». Ограниченное распространение получила иерархическая топология ЛВС, где абоненты соединяются в виде «дерева» с разветвляющимися ветвями (соединение многих «звезд»). Основные три топологии существенно различаются областями применения, часто не могут заменить одна другую, и поэтому их использование оправдано.

Топология сети очень сильно влияет на методы управления в ней, на ее отказоустойчивость и даже на ее стоимость. Поэтому выбор метода управления или среды передачи диктует выбор топологии.

Кратко остановимся на основных особенностях наиболее распространенных топологий.

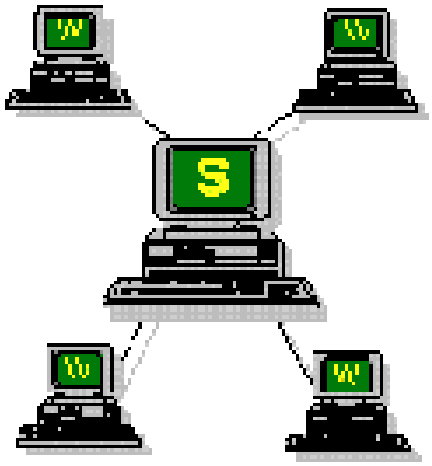


Рис. 14

1. Вычислительная сеть типа "звезда" - принципиально централизованная топология, в которой всегда есть четко выделенный центральный абонент, который осуществляет все управление обменом в сети и через который идет вся информация в сети (рис. 14). Любое жесткое централизованное управление по своей сути бесконфликтно, но нарушения в работе центра приводят к выходу из строя всей системы. Такая сеть не будет ра-

ботать при любой неисправности центрального абонента, поэтому центральный компьютер должен быть гораздо надежнее остальных. К тому же центральный компьютер будет сильно загружен работой с сетью и не сможет заниматься другими задачами. Сеть с такой конфигурацией малочувствительна к выходу из строя соединительного кабеля. Разрыв кабеля в любом месте всегда нарушает связь только с одним абонентом, вся остальная сеть остается работоспособной. К недостаткам данной топологии относится ограниченное число абонентов (редко превышает 16) и невозможность его увеличения без специальных программных решений.

Название «звезда» применяется для двух различных типов топологий: «активная звезда» и «пассивная звезда».

«Активная звезда» - это истинная «звезда» (в центре есть абонент, как на рис. 14), а «пассивная звезда» фактически представляет собой «шину», где абоненты соединены в виде звезды. В центре этой звезды нет абонента, все кабели там просто соединены между собой. Там также могут находиться повторители сигналов, некие пассивные преобразователи и т.д. В любом случае в центре «пассивной звезды» нет никакой обработки информации и никакого управления обменом.

2. Вычислительная сеть типа «кольцо» (рис. 15) основана на использовании однонаправленного высокоскоростного канала связи, образующего замкнутое кольцо или петлю.

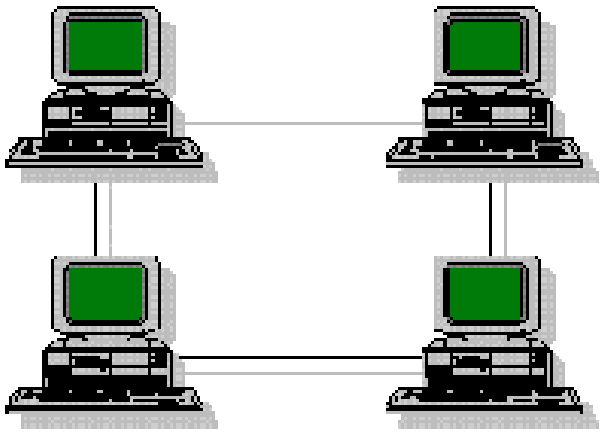


Рис. 15

Вся передаваемая информация проходит через всех абонентов, поэтому выход из строя любого из них (во всяком случае, его адаптера) нарушает работу всей сети в целом. Разрыв кабеля в любой точке нарушает целостность кольца и выводит из строя всю сеть. Для предотвращения такой ситуации иногда применяют дублирование кабеля. Управление может быть как централизованным, так и децентрализованным, оно не так жестко зависит от топологии, как в случае «звезды». Все адаптеры должны быть одинаковы, но иногда один из них выполняет функцию диспетчера сети. Топология обычно допускает значительное число абонентов (1024 и вы-

ше). В кольце автоматически производится усиление передаваемых сигналов каждым абонентом, поэтому размеры его могут быть очень большими (до нескольких десятков километров) и ограничены только временем прохождения сигнала по всему кольцу.

3. Магистральная вычислительная сеть «шина» (рис. 16) строится на основе одного общего канала связи и коллективного использования его в режиме разделения времени.

Рис. 16

«Шина» ориентирована на полное равноправие всех абонентов и идентичность их адаптеров. Это не значит, что управление обменом не может быть централизованным, правда, центр будет заниматься только управлением обменом, а не перераспределением информации, как в случае «звезды». В «шине» в отличие от «звезды» и «кольца» чрезвычайно важны вопросы электрического согласования используемых линий связи, так как при любом повреждении кабеля возникают отражения и наложения сигналов, полностью нарушающие работу сети. К выходу из строя компьютеров «шина» нечувствительна: нарушается только обмен с неисправным компьютером, а вся остальная сеть остается работоспособной. Максимально допустимое количество абонентов в «шине» примерно такое же, как и в «кольце». В «шине» легко можно менять количество подключенных абонентов, иногда даже прямо в процессе работы. Сложность аппаратуры адаптеров в «шине», как правило, выше, чем в других топологиях, это связано с необходимостью работы на большое число нагрузок, а также со сложностью децентрализованного управления обменом по сети. Однако децентрализованное управление гораздо надежнее централизованного и лучше приспосабливается к изменяющимся внешним условиям.

Примером такой сети может служить сеть “Ethernet”, разработанная фирмой “Xerox Corp.”

4. Иерархическая вычислительная сеть

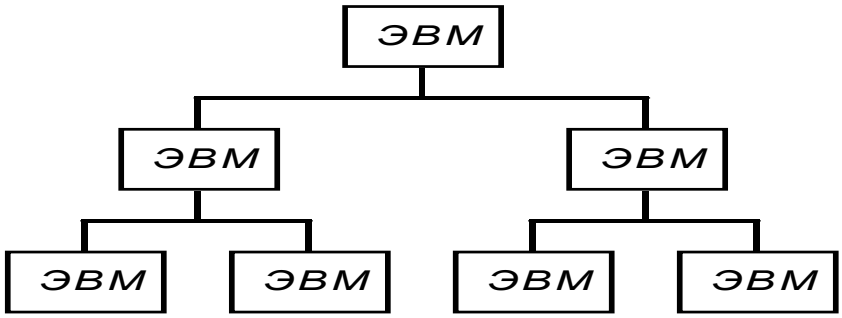


Рис. 17

Возможности ЭВМ в такой сети (рис. 17) увеличиваются от нижних уровней к верхним. На надежность сети основное влияние оказывает ЭВМ верхнего уровня.

Среды передачи информации

Информация в локальных сетях передается в последовательном коде, бит за битом. Передача на большие расстояния при любом типе кабеля требует сложной передающей и приемной аппаратуры: надо формировать мощный сигнал на передающем конце и детектировать слабый сигнал на приемном конце.

Наиболее часто в локальных сетях используются следующие типы линий связи, или, как их часто называют, сред передачи информации: электрический кабель (витая пара проводов или коаксиальный кабель), оптоволоконный (он же волоконно-оптический) кабель, радиоканал и инфракрасный канал (последние два типа не требуют проводов). Кратко перечислим их основные особенности [17].

1. Витая пара - самый дешевый тип соединительных проводов. Представляет собой скрученные между собой два провода в диэлектрической изоляции (рис. 18). Часто витой парой называют также не перекрученные, а параллельно идущие провода в общей изоляции («лапша»), хотя характеристики у него несколько иные. Витая пара характеризуется исключительной простотой монтажа и ремонта повреждений. Как правило, витая пара используется для передачи на скорости до 10 Мбит/с. К недостаткам витой пары относятся низкий уровень защищенности от электрических и магнитных помех и большой уровень собственных излучений, а также возможность простого несанкционированного подключения к сети. Иногда используется экранированная витая пара, которая свободна от подобных недостатков. Витая пара обычно используется для связи на расстояниях до нескольких сот метров. Затухание сигнала на частоте 10 МГц составляет около 1,0-3,0 дБ/м. Задержка сигнала в витой паре обычно не превышает 8-

12 нс/м. Как и любой длинной электрической линии связи, витой паре требуется согласование на концах с целью уменьшения отражений сигнала.

Рис. 18

2. Коаксиальный кабель (рис. 19) представляет собой центральный проводник в изоляции, помещенный в гибкий металлический коаксиальный экран. Наличие экрана сильно увеличивает помехозащищенность и снижает собственное излучение. Несанкционированное подключение к коаксиальному кабелю сложнее, чем к витой паре, но все равно возможно. В режиме модуляции высокочастотного сигнала пропускная способность может достигать 500 Мбит/с, а в немодулированном режиме она редко превышает 50 - 100 Мбит/с. Стоимость коаксиального кабеля в несколько раз выше стоимости витой пары. Затухание сигнала на частоте 10 МГц составляет порядка 0.1 - 1.0 дБ/м. Задержка распространения сигнала - от 4 до 5 нс/м. При использовании коаксиального кабеля важно обеспечить гальваническую развязку и согласование на концах линии. Стоит отметить, что в случае применения электрического кабеля (коаксиального или витой пары) не требуются никакие сложные преобразователи сигналов на приемном или передающем конце, так как передается электрический сигнал.

Рис. 19

3. Волоконно-оптический кабель - это качественно иной тип среды передачи информации. Сигнал по нему передается не электрический, а световой, что требует соответственно преобразования электрического сигнала в световой на передающем конце и обратного преобразования на приемном конце. Это увеличивает стоимость аппаратуры. Но уникальные характеристики волоконно-оптических кабелей обеспечивают им все большее распространение. Свет с длиной волны 0.85 мкм или 1.2 мкм передается по тонкому (порядка 10 мкм) стекловолокну, заключенному в оболочку, которая имеет значительно меньший коэффициент преломления. Это приводит к эффекту полного внутреннего отражения, так как свет проходит по кабелю, не выходя наружу. Главное достоинство этого типа кабеля - чрезвычайно высокий уровень помехозащищенности и отсутствие излучения. Невозможно несанкционированное подключение. Максимальная длина кабеля без ретрансляции может достигать нескольких десятков километров. Скорость передачи до 3 Гбит/с, задержка сигнала ~ около 5 нс/м.

На частотах более 200 МГц волоконно-оптические кабели имеют несомненное преимущество перед любыми электрическими кабелями.

Оптоволоконные кабели допускают разветвления и ответвления, но они существенно увеличивают затухание, поэтому предпочитают использовать однонаправленные кабели. Это сразу определяет возможные топологии сети: «звезда» (с двумя разнонаправленными кабелями между центральным абонентом и каждым из периферийных) или «кольцо» (с одним однонаправленным кабелем). Недостатки данного типа кабеля - его меньшая механическая прочность, чувствительность к ионизирующим излучениям (снижается прозрачность оптоволокна). При использовании волоконно-оптического кабеля автоматически решается задача гальванической развязки и не требуется никакого согласования кабеля.

4. Радиоканал удобен прежде всего тем, что абоненты ничем не связаны друг с другом, они могут легко изменять свое расположение. В данном случае не нужно прокладывать кабель и следить за его сохранностью. Радиоканал может поддерживать связь на десятки километров и обеспечивать предельные скорости передачи до десятков мегабит в секунду. Но широкого распространения в локальных сетях радиоканал не получил из-за сравнительно высокой стоимости приемных и передающих средств (необходимо преобразовывать электрический сигнал в радиосигнал и обратно), а также низкой помехозащищенности передаваемой информации.

5. Инфракрасный канал, как и радиоканал, не требует соединительных проводов, что является его большим достоинством.

В отличие от радиоканала он нечувствителен к электромагнитным помехам, и это позволяет использовать его в производственных условиях. К недостаткам инфракрасного канала относятся высокая стоимость приемников и передатчиков, где требуется преобразование электрического сигнала в инфракрасный и обратно, а также низкие скорости передачи (не выше 5 Мбит/с). В условиях прямой видимости инфракрасный канал может обеспечить связь на расстояниях несколько километров, но наиболее удобен он для связи компьютеров, находящихся в одной комнате, где отражение от стен комнаты дает устойчивую и надежную связь. Наиболее естественный тип топологии здесь - «шина» (т. е. переданный сигнал одновременно получают все абоненты). Имея такое количество недостатков, инфракрасный канал не смог получить широкого распространения.

Методы кодирования информации

Передаваемая по линии связи информация обычно подвергается специальному кодированию, которое способствует повышению надежности передачи.

Наиболее часто в локальных сетях используются следующие коды передачи информации (рис. 20).

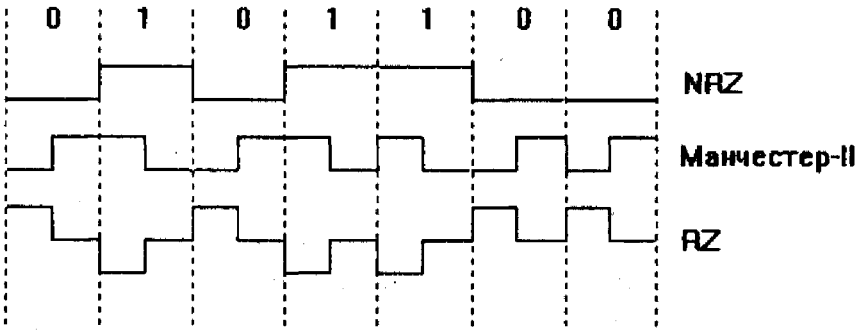


Рис. 20

1. NRZ (*Non Return to Zero* - без возврата к нулю) - простейший код, представляющий собой обычный цифровой сигнал (правда, код может быть преобразован на обратную полярность или изменены уровни, соответствующие нулю и единице). К несомненным достоинствам кода *NRZ* относятся его очень простая реализация (исходный сигнал не надо ни кодировать на передающем конце, ни декодировать на приемном конце), а также минимальная, среди других кодов требуемая при данной скорости передачи пропускная способность линии связи.

Самый большой недостаток данного кода - отсутствие синхросигнала информации, позволяющего приемнику синхронизовать прием информации из сети с передачей ее передатчиком. В этом случае приемник может выбирать данные из сети не в нужный момент, если его частота приема несколько отличается от частоты передачи (уход часов приемника). Особенно это критично для больших блоков (пакетов) информации (1-2 килобайта и более). К концу принимаемого пакета теряется взаимная синхронизация передачи и приема, следовательно, возможна потеря данных. Поэтому код *NRZ* используется только для передачи короткими пакетами (обычно до 1 Кбита).

Для синхронизации начала приема пакета используется стартовый служебный бит, чей уровень отличается от пассивного состояния линии связи (например, пассивное состояние линии при отсутствии передачи - 0, стартовый бит - 1). Наиболее известное применение кода *NRZ* - стандарт RS-232C, последовательный порт персонального компьютера, который вполне может использоваться для организации небольших и медленных локальных сетей. Передача информации в нем ведется байтами (8 бит), сопровождаемыми стартовым и стоповым битами.

2. RZ (*Return to Zero* - с возвратом к нулю) - этот трехуровневый код получил такое название потому, что после значащего уровня сигнала в первой половине передаваемого бита информации следует возврат к некоему «нулевому» уровню (например, соответствующему нулевому потенциалу).

Переход к нему происходит в середине бита. Логическому нулю при этом соответствует положительный импульс, логической единице - отрицательный (или наоборот). Особенностью кода *RZ* является то, что в центре бита всегда есть переход (положительный или отрицательный), следовательно, из этого кода приемник может выделить синхроимпульс (строб). Здесь возможна привязка не только к началу пакета, как в случае кода *NRZ*, но и к каждому отдельному биту. Поэтому рассинхронизации не произойдет даже при очень длинных пакетах. Такие коды, несущие в себе строб, получили название *самосинхронизирующихся*. Недостаток кода *RZ* состоит в том, что требуется вдвое большая полоса пропускания канала при той же скорости передачи по сравнению с *NRZ***,** так как здесь на 1 бит приходится 2 изменения уровня напряжения. Для скорости передачи информации 10 Мбит/с требуется пропускная способность линии 10 МГц. Наиболее часто код *RZ* используется в оптоволоконных сетях. Правда, в них нет положительных и отрицательных уровней сигнала, поэтому используется три уровня: отсутствие света, «слабый» свет, «сильный» свет. Это очень удобно: даже когда нет передачи информации, свет все равно есть, что позволяет легко определить целостность оптоволоконной линии связи.

3. Код Манчестер-II, или *манчестерский код*, получил наибольшее распространение в сетях, использующих электрические кабели. Он также относится к самосинхронизирующимся кодам, но в отличие от кода *RZ* имеет не три, а только два уровня, что способствует его лучшей помехозащищенности. Логическому нулю соответствует положительный переход в центре бита, т.е. первая половина битового интервала - низкий уровень, вторая половина - высокий, логической единице соответствует отрицательный переход в центре бита, или наоборот. Обязательное наличие перехода в центре бита позволяет легко выделить синхросигнал, что дает возможность передавать информацию сколь угодно большими пакетами без потерь из-за рассинхронизации. Допустимое расхождение часов приемника и передатчика может достигать 25%. Как и в случае кода *RZ*, здесь требуется пропускная способность линии в два раза выше, чем при использовании *NRZ*. Для скорости передачи 10 Мбит/с требуется полоса пропускания 10 МГц.

Большое достоинство манчестерского кода - отсутствие постоянной составляющей в сигнале (половину времени сигнал положительный, другую половину - отрицательный). Это дает возможность применять для гальванической развязки импульсные трансформаторы. При этом не требуется дополнительного источника питания для линии связи (как в случае использования оптронной развязки), резко уменьшается влияние низкочастотных помех, не проходящих через трансформатор, легко решается проблема согласования.

Так же, как и при использовании кода *RZ*, при манчестерском кодировании очень просто определять занятость сети. Для этого достаточно кон-

тролировать, есть ли изменение сигнала в течение битового интервала. Код *Манчестер-II* используется как в электрических кабелях, так и в оптоволоконных. В самой распространенной локальной сети Ethernet используется именно этот код.

Помимо рассмотренных здесь, в локальных сетях используются и другие коды, например, логической единице соответствует один импульс на битовом интервале, а логическому нулю - два импульса. Применяются также различные методы модуляции: амплитудной, частотной и фазовой.

Форматы пакетов локальных сетей

Информация в локальных сетях передается отдельными порциями, называемыми в различных источниках пакетами, кадрами или блоками. Использование пакетов связано с тем, что в сети одновременно может происходить несколько сеансов связи (при топологиях типа «шина» и «кольцо»). В течение одного и того же интервала времени могут идти два процесса или более передачи данных между абонентами. Пакеты как раз и позволяют разделить во времени сеть между передающими информацию абонентами. Если бы вся требуемая информация передавалась непрерывно, без разделения на пакеты, то это привело бы к монопольному захвату сети одним из абонентов на довольно продолжительное время.

Структура пакета определяется, прежде всего, аппаратурными особенностями данной сети, выбранной топологией и типом среды передачи информации. Она также существенно зависит от используемого протокола (порядка обмена информацией). Типичный пакет содержит в себе следующие основные поля (части):

- стартовая комбинация (не обязательна) - обеспечивает настройку аппаратуры адаптера или другого сетевого устройства на прием и обработку пакета;

- сетевой адрес (идентификатор) принимающего абонента - индивидуальный или групповой номер, присвоенный принимающему абоненту в сети, - позволяет приемнику распознать пакет, адресованный ему или всем абонентам сети;

- сетевой адрес (идентификатор) передающего абонента - индивидуальный или групповой номер, присвоенный передающему абоненту, - информирует принимающего абонента, откуда пришел данный пакет;

- служебная информация - указывает на тип пакета, его номер, на то, что с ним надо делать, и т. д.;

- данные - это собственно информация, ради передачи которой используется данный пакет;

- контрольная сумма пакета - числовой код, формируемый передатчиком по определенным правилам и содержащий в свернутом виде информа-

цию обо всем пакете, - используется для проверки правильности передачи пакета на приемном конце;

- стоповая комбинация (не обязательна) - информирует принимающего абонента об окончании пакета, обеспечивает выход аппаратуры приемника из состояния приема.

На рис. 21 приведена реальная структура кадра (пакета), принятая в наиболее распространенной сети Ethernet. В начале передается преамбула (стартовая комбинация), затем сетевые адреса получателя (приемника) и отправителя (передатчика), затем следуют управляющие байты, байты данных и контрольная сумма (цифры показывают количество байт). При необходимости увеличить количество управляющей информации под нее отводится нужное число байт данных.

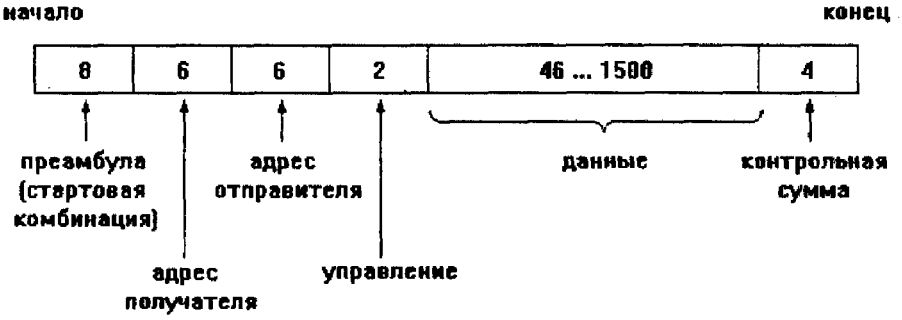


Рис. 21

Пакеты делятся на два основных типа: управляющие (не содержат поля данных) и информационные (поле данных присутствует). Управляющие пакеты служат для решения вспомогательных задач по обмену в сети, например, для установления соединения между абонентами, для завершения соединения после прекращения сеанса обмена, для подтверждения приема информационного пакета. Простейший пример сеанса связи между двумя абонентами представлен на рис. 22.

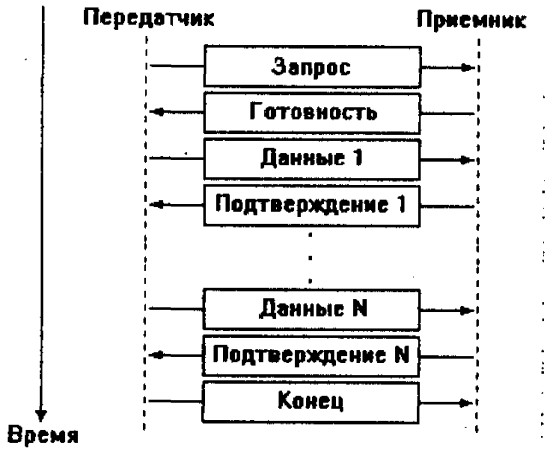


Рис. 22

Передающий абонент сначала запрашивает с помощью управляющего пакета принимающего абонента о готовности принять данные. Прини-

мающий отвечает управляющим пакетом о своей готовности. Затем следует собственно передача данных, причем на каждый информационный пакет от передатчика приемник отвечает соответствующим пакетом подтверждения приема. После окончания передачи данных передающий абонент заканчивает сеанс связи управляющим пакетом.

В заключение отметим, что существуют два термина, относящихся к сетям вообще: коммутация пакетов и коммутация каналов (цепей). При коммутации пакетов по одному каналу идут пакеты разного назначения с временным разделением. Каждый абонент отлавливает в сети те пакеты, которые адресованы ему. При коммутации каналов (цепей) производится физическое переключение различных каналов, линий связи (пример - обычная телефонная сеть). Коммутация в данном случае осуществляется в узлах коммутации (телефонные станции).

Доступ в локальные вычислительные сети

В локальную сеть всегда входит несколько абонентов, причем каждый из них, как правило, работает самостоятельно и в любой момент может обратиться к сети. Поэтому требуется управление обменом с целью упорядочения использования сети различными абонентами, предотвращения или разрешения конфликтов между ними. В противном случае возможно искажение передаваемой информации.

Обеспечение доступа в сетях с общим каналом передачи данных (кольцевая и магистральная сети) связано с проблемой распределения времени для использования линий связи. Для управления обменом (или управления доступом к сети, или арбитража сети) используется множество различных методов, все многообразие которых можно свести к четырем основным:

- случайные методы доступа;
- интервальные методы доступа;
- маркерные методы доступа;
- интервально-маркерные методы доступа.

Конкретный метод доступа в значительной степени определяется выбранной топологией сети.

Управление обменом в сети типа «звезда»

Отметим, что здесь речь пойдет об активной, истинной «звезде». При данной топологии (рис. 14) все периферийные абоненты могут решить передавать информацию одновременно.

Чаще всего центральный абонент может производить обмен только с одним из периферийных абонентов, поэтому в любой момент надо выделить только одного абонента, ведущего передачу.

Здесь возможны два решения.

Первое из них можно назвать «активный центр». В этом случае центральный абонент посылает запросы (управляющие пакеты) по очереди всем периферийным абонентам. Периферийный абонент, который хочет передавать (первый из опрошенных), посылает ответ или же сразу начинает передачу. Сеанс обмена проводится именно с ним. После окончания этого сеанса центральный абонент продолжает опрос по кругу. Периферийные абоненты имеют в данном случае так называемые географические приоритеты: максимальный приоритет у того из них, кто ближе расположен к последнему абоненту, закончившему обмен. Если же хочет передать центральный абонент, он передает безо всякой очереди.

Второе возможное решение - «пассивный центр». В этом случае центральный абонент по очереди не опрашивает, а слушает всех периферийных абонентов (т. е. принимает пакеты только от одного из них). Те периферийные абоненты, которые хотят передать, периодически посылают запросы и ждут на них ответа. Когда центр принимает запрос, он отвечает запросившему абоненту (разрешает ему передачу), и тот передает. Приоритеты здесь такие же, как и в первом случае.

Как в первом, так и во втором случае управление обменом централизованное. Все решения о порядке обмена принимает единый центр (центральный абонент).

Преимущества централизованного управления достаточно очевидны: принципиальная невозможность любых конфликтов между абонентами (все решения принимаются в одном месте), а также гарантированное время доступа, т.е. время, проходящее от момента возникновения желания передавать до момента начала передачи. Недостатки централизованного управления связаны с низкой устойчивостью этого метода к отказам (если центр выходит из строя - обмен невозможен), а также с недостаточной гибкостью (центр всегда работает по жестко заданному алгоритму) *и* низкой скоростью управления (даже если все время передает только один периферийный абонент, ему приходится ждать, пока центр опросит всех).

Управление обменом в сети типа «шина»

В этой топологии возможно точно такое же централизованное управление, как и в «звезде» (т.е. физически сеть - «шина», но логически - «звезда»). При этом один из абонентов («центральный») посылает всем остальным («периферийным») запросы, выясняя, кто хочет передать, затем разрешает передачу одному из них. Единственное принципиальное отличие состоит в том, что центр здесь не перекачивает информацию от одного абонента к другому, как в «звезде», а только управляет доступом.

Однако гораздо чаще в «шине» реализуется децентрализованное управление, так как аппаратные средства абонентов одинаковы. При этом все абоненты имеют равный доступ к сети, и решение, когда можно пере-

давать, принимается каждым абонентом на месте, исходя из анализа состояния сети. Следовательно, возможны конфликты между ними и искажения передаваемых данных из-за наложения пакетов.

Выбор алгоритмов доступа зависит от скорости передачи в сети, от длины шины, загруженности сети (интенсивности обмена или трафика сети).

1. Простейший метод, используемый при выбранном коде обмена *NRZ* в сравнительно медленных сетях, - *децентрализованный кодовый приоритетный арбитраж.* Его смысл состоит в распознавании столкновения двух или более пакетов в начале передачи и прекращении в случае столкновения передачи всеми абонентами, кроме одного. Здесь важно отметить, что при использовании кода *NRZ* очень непросто определить, идет передача или нет, т. е. занята сеть или свободна. Если в случае применения кодов *RZ* и *Манчестер-II* это сделать довольно просто исходя только из текущего анализа состояния сети (грубо говоря, посмотреть, есть изменения сигналов или нет), то здесь возникает сложная проблема. Если в передаваемом пакете присутствуют длинные последовательности «0» или «1», которые совпадают с пассивным состоянием сети (в отсутствие передачи), то, исходя из анализа состояния сети, сделать вывод о ее занятости невозможно. Поэтому обычно используется такой подход.

Во-первых, любой пакет начинается стартовым битом, уровень которого отличается от пассивного состояния сети.

Во-вторых, передача ведется пакетами только стандартной длины (например, 1 Кбит) независимо от их функционального назначения. В результате все абоненты, обнаружив стартовый бит пакета после пассивного состояния сети, начинают отсчет временного окна, длительность которого равна длительности пакета (рис. 23). А дальше любой абонент начинает передачу только тогда, когда сеть свободна.

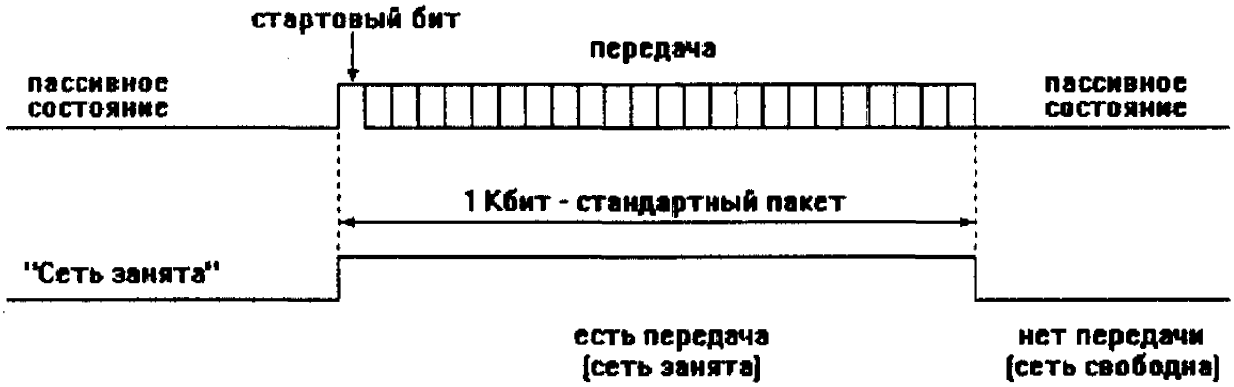


Рис. 23

Вероятность столкновения при малой интенсивности обмена в сети ничтожно мала - для этого должны одновременно начать передачу два или более абонентов. Однако при большой интенсивности обмена в сети си-

туация резко усложняется. Как при этом избежать искажения информации?

В начале любого пакета все абоненты передают свой сетевой адрес. И одновременно с этой передачей они проверяют соответствие того, что они хотят передать, тому, что есть в сети на самом деле (в пределах каждого бита). Если обнаруживается расхождение между желаемым и действительным, то абонент прекращает передачу.

В данном случае победу одержит (т.е. не обнаружит расхождения и продолжит передачу) тот абонент, у которого в коде сетевого адреса будет больше единиц (кроме этого, важен еще и порядок единиц и нулей). Максимальным приоритетом будет обладать абонент с сетевым адресом 111...1. Те же абоненты, которые проиграют (т.е. обнаружат расхождение и прекратят передачу), будут ждать освобождения сети и затем повторят свою попытку.

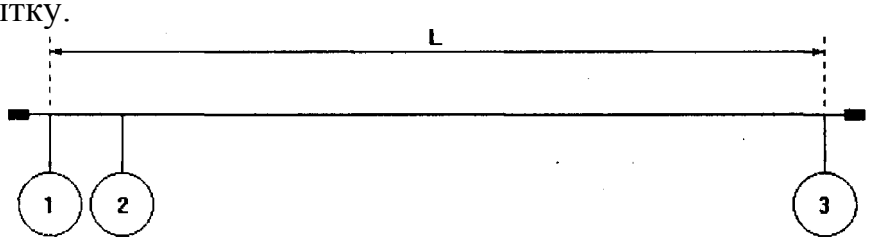


Рис. 24

При этом возникает вопрос: какова должна быть длительность бита, чтобы все абоненты точно обнаружили столкновение? Или в более общей формулировке: в каком случае можно говорить об одновременности событий в сети? Ведь скорость распространения сигнала по любому кабелю ограничена, поэтому информацию о произошедшем событии не все абоненты получают одновременно.

Пусть L - полная длина сети, V - скорость распространения сигнала в используемом кабеле (рис. 24). Допустим, абонент 1 закончил свою передачу, а абоненты 2 и 3 захотели передавать во время передачи абонента 1. После освобождения сети абонент 3 узнает об этом событии через временной интервал прохождения сигнала по всей длине сети, т.е. через L/V, а абонент 2 - сразу после освобождения. Но абонент 2 узнает о том, что абонент 3 тоже узнал об освобождении, еще через время L/V. Поэтому в общем случае говорить об «одновременности» событий в сети можно даже тогда, когда между ними временной сдвиг, меньший, чем $2L/V$, т.е. удвоенное время распространения сигнала по полной длине сети. То же самое будет происходить и на каждом битовом интервале. Следовательно, длительность бита при данном типе арбитража должна быть не менее $2L/V$, чтобы можно было точно обнаруживать все столкновения. Здесь происходит побитное сравнение, поэтому данная величина ограничивает именно длительность бита.

В сетях данного типа довольно легко реализуется подтверждение получения пакета приемником в течение времени передачи того же пакета: последний бит пакета передатчик оставляет пассивным, а приемник, если он успешно принял пакет, передает этот бит обратно активным уровнем. При других кодах это невозможно, там для подтверждения приема всегда используется отдельный управляющий пакет.

Рассмотренный метод арбитража характеризуется низкой скоростью передачи, но высокой надежностью. Все абоненты имеют свои собственные приоритеты, которые могут динамически изменяться в зависимости от важности информации.

2. Второй метод, используемый в шине, - *децентрализованный временной приоритетный арбитраж.* Основная идея данного метода состоит в том, чтобы свести вероятность столкновения к пренебрежимо малой величине.

Идея рассматриваемого здесь метода (рис. 25) состоит в том, чтобы полностью исключить столкновения после освобождения сети. Для этого предлагается следующий алгоритм. Сначала все абоненты следят за состоянием сети. Если она свободна, то передача начинается сразу после возникновения заявки на нее. Если же сеть занята, то сразу после ее освобождения все абоненты отсчитывают свой собственный уникальный временной интервал, пропорциональный коду сетевого адреса данного абонента. Таким образом, абонент с нулевым адресом начинает передачу сразу, абонент с первым адресом - через время t_0, абонент со вторым адресом - через время $2t_0$ и т.д. Если к концу этого временного интервала сеть все еще остается свободной, то абонент начинает передачу. В противном случае - снова ждет освобождения сети.

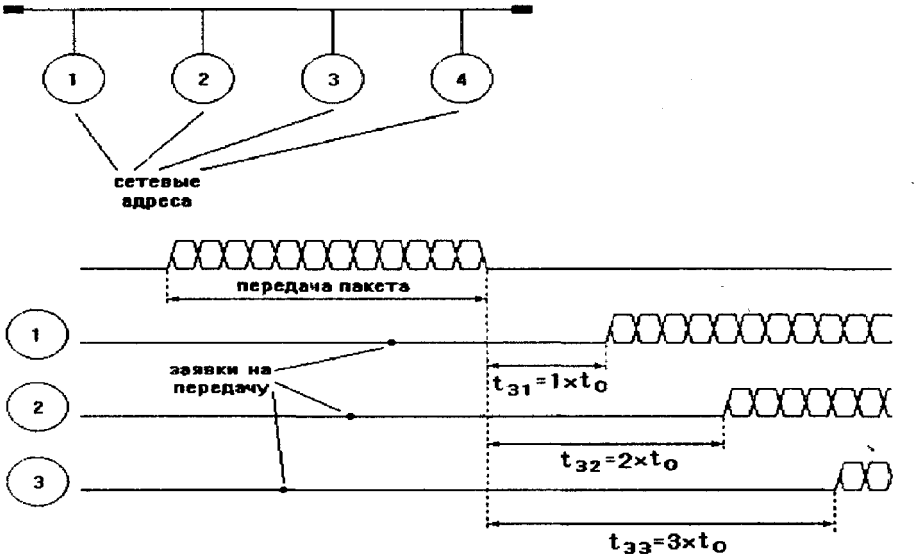


Рис. 25

Особенности данного метода следующие. При малой интенсивности обмена в сети все абоненты равноправны, а при большой - все они имеют

свои приоритеты, причем максимальный приоритет - у абонента, имеющего минимальный сетевой адрес (минимальное время задержки начала передачи). Поэтому абоненты с малыми приоритетами могут довольно долго ждать. О гарантированном времени доступа к сети здесь, как и в случае предыдущего метода, говорить не приходится. Несмотря на очень малую вероятность столкновений, полностью они не исключаются (например, при свободной сети заявки на передачу могут возникнуть одновременно).

При выборе данного метода управления обменом минимальное значение величины дискрета задержки t_0 не должно быть меньше, чем $2L/V$, где L - полная длина сети, V - скорость распространения сигнала в используемом кабеле.Так как столкновения при данном методе все-таки возможны, а обнаружение их не предусмотрено, эта функция переносится на более высокий уровень обмена. Например, вывод о столкновении делается приемником на основании анализа контрольной суммы полученного пакета.

Этот метод управления обменом быстрее, чем предыдущий (он допускает более высокие скорости обмена, так как здесь нет ограничения на длительность каждого бита). Недостаток метода состоит и в том, что в случае большой длины сети и большого количества абонентов задержки становятся очень большими. Пример: при длине сети 1 км, задержке сигнала в кабеле 4 нс/м и количестве абонентов 256 минимальная задержка составит t_0=8 мкс, следовательно, для абонента с сетевым адресом 255 задержка будет уже равна 255 х 8 мкс = 2040 мкс, т.е. около 2 мс. Для сравнения: если пакет имеет размер 1 Кбайт, то при скорости передачи 10 Мбит/с его длительность будет всего 0.8 мс.

Данный метод управления обменом, в отличие от предыдущего, не имеет такой жесткой привязки к коду передачи**.** Главное - выбрать его таким, чтобы можно было определять занятость сети.

3. Третий метод, получивший довольно широкое распространение, можно считать развитием второго. Называется он CSMA/CD *(Carrier-Sense Multiple Access/Collision Detection)* - метод доступа с контролем несущей и обнаружением коллизий (столкновений). Идея метода состоит в том, чтобы уравнять в правах всех абонентов в любой возможной ситуации, т.е. добиться, чтобы не было больших и малых фиксированных приоритетов (как в случае второго метода). Для этого используются времена задержки, вычисляемые каждым абонентом по определенному алгоритму.

Суть метода состоит в следующем.

А. Абонент, желающий передавать, следит за состоянием сети и в случае ее занятости ждет освобождения (это и называется контролем несущей, так как здесь обычно используется код *Манчестер-II*, следовательно, можно говорить о несущей частоте). Если сеть свободна, то передача начинается безо всякого ожидания.

Б. После освобождения сети абонент сразу начинает передавать и одновременно контролирует состояние сети (это тот самый контроль столк-

новений или обнаружение коллизий). Если столкновений не обнаруживается, передача доводится до конца.

В. Если столкновение обнаружено, то абонент усиливает его (передает еще некоторое время) для гарантии обнаружения всеми передающими абонентами, а затем отключается (прекращает передачу). Точно так же поступают другие передававшие абоненты.

Г. После прекращения неудачной передачи абонент выдерживает случайно выбираемый промежуток времени, а затем повторяет свою попытку передать, при этом контролируя столкновения. Так как этот промежуток времени случайный, вероятность повторного столкновения пакетов будет довольно мала.

В наиболее распространенной локальной сети Ethernet используется именно этот метод доступа.

К достоинствам метода CSMA/CD можно отнести полное равноправие всех абонентов, т.е. ни один из них не может надолго захватить сеть. Метод достаточно надежен: ведь в течение всего времени передачи пакета идет контроль столкновений. Основной недостаток метода - негарантированное время доступа: нельзя сказать наверняка, за сколько времени пакет точно дойдет до приемника.

Управление обменом в сети типа «кольцо»

Кольцевая топология имеет свои особенности при выборе метода управления обменом. Важным фактором является то, что любой пакет, посланный по кольцу, последовательно пройдя всех абонентов, через некоторое время возвратится в ту же точку (топология замкнутая). Здесь нет одновременного распространения сигнала в две стороны, как в «шине». Отметим, что сети типа «кольцо» бывают однонаправленными и двунаправленными. Мы будем рассматривать только однонаправленные, как более распространенные. В сети типа «кольцо» можно также использовать различные методы управления - логическую «звезду», логическую «шину», но чаще выбирают специфические методы, в наибольшей степени соответствующие именно особенностям «кольца».

1. Маркерный метод управления относится, как и методы опроса (централизованные), к детерминированным методам. В отличие от случайных методов (которые чаще используются при шинной топологии, например, CSMA/CD) детерминированные методы принципиально исключают любые конфликты в сети, так как в них предусмотрен механизм временного распределения сети между абонентами. При случайных методах все абоненты могут начать передачу в любой момент, поэтому там конфликты неизбежны.

Идея метода маркерного управления состоит в том, что по «кольцу» запускается специальный пакет, называемый маркером, который отмечает время возможного начала пакета (играет роль своеобразной временной

метки). Этот маркер непрерывно ходит по «кольцу», синхронизируя работу абонентов сети. Алгоритм управления предполагает следующую последовательность действий (рис. 26).

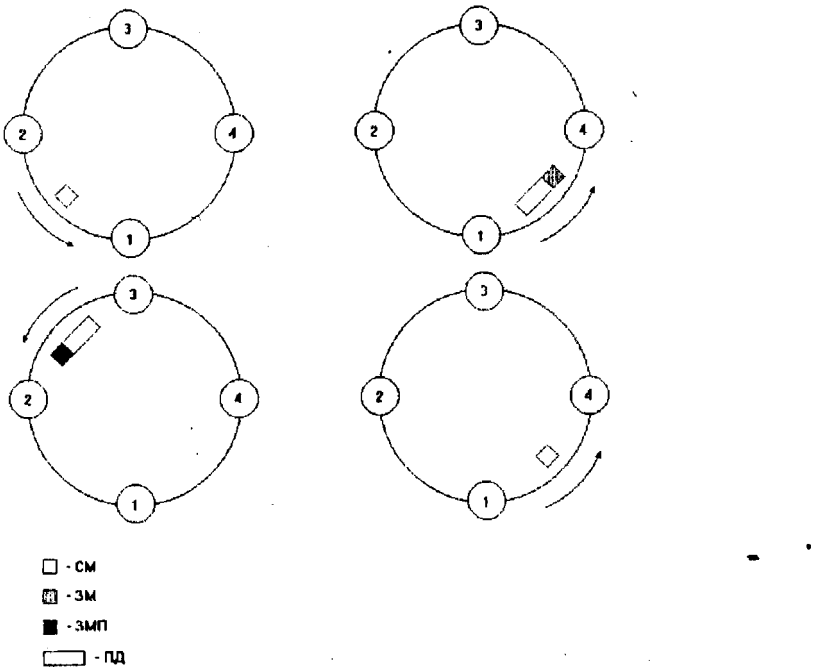


Рис. 26

СМ - свободный маркер, ЗМ - занятый маркер,
ЗМП - занятый маркер с подтверждением, ПД - пакет данных

А. Абонент 1, желающий передать, ждет так называемый *свободный* маркер (маркерный пакет, помеченный в специально выделенном поле как свободный). Получив такой маркер, абонент 1 помечает его как занятый (изменяет соответствующие биты), добавляет к нему свой пакет и отправляет полученную связку (маркер + пакет) дальше в «кольцо».

Б. Каждый абонент (2, 3, 4), получивший этот «паровозик», проверяет, ему ли адресован пакет. Если пакет не его, абонент отправляет его дальше по «кольцу». В данном случае пакет обязательно должен быть принят каждым абонентом и только потом отправлен (или не отправлен) дальше. Здесь мы наблюдаем последовательную, эстафетную передачу пакета от абонента к абоненту.

В. Абонент, распознавший пакет, который адресован ему (в нашем случае это абонент 3), принимает этот пакет, устанавливает в маркере специально выделенный бит подтверждения и отправляет «паровозик» из маркера и пакета дальше.

Г. Передававший абонент 1 получает обратно свою посылку, прошедшую через все кольцо, затем освобождает маркер (помечает его как свободный) и снова посылает маркер в сеть. При этом из анализа бита подтверждения ему уже известно, принят его пакет адресатом или нет.

Приоритет при данном методе управления - географический, т.е. право передачи после освобождения сети переходит к следующему по направлению «кольца» абоненту от последнего передававшего абонента. Но система приоритетов работает только при большой интенсивности обмена. При малой интенсивности обмена все абоненты равноправны, и время доступа к сети определяется текущим положением маркера.

Отметим, что явно выделенного центра управления здесь не существует. Однако некий «центр» все-таки должен присутствовать: один из абонентов (или специальное устройство) должен следить, чтобы маркер не потерялся (например, из-за помех или сбоя в работе какого-то абонента). В противном случае механизм доступа работать не будет. Следовательно, надежность управления в данном случае снижается (выход «центра» из строя приводит к полной дезорганизации обмена).

Основным преимуществом данного метода перед CSMA/CD является гарантированное время доступа. Здесь его величина составит $N \cdot t_{пк}$, где N - число абонентов, $t_{пк}$ - время прохождения пакета по «кольцу». Вообще маркерный метод управления обменом гораздо эффективнее случайных методов при очень большой интенсивности обмена в сети.

2. Метод кольцевых сегментов (слотов). Примером сети, использующей данный метод управления, может служить сеть Cambridge Ring. Основное отличие этого метода от маркерного состоит в том, что нескольким абонентам разрешена передача одновременно и в любой момент (при маркерном методе всегда передает только один абонент). Вместо одного маркера в сети используются несколько так называемых слотов (обычно от 2 до 8), которые выполняют, по сути, ту же самую функцию, что и маркер, - функцию временных меток. Эти слоты идут по «кольцу» довольно часто, временной интервал между ними невелик, и поэтому между ними может уместиться немного информации (обычно от 8 до 32 байт). При этом каждый слот может находиться в свободном или занятом состоянии. Алгоритм доступа к сети, реализуемый при данном методе, включает в себя следующие этапы.

А. Абонент, желающий передавать, разбивает свою информацию на слоты (маленькие пакеты) установленного размера.

Б. Затем он ждет прихода свободного слота и загружает его первой частью своей информации, потом ждет следующего свободного слота и помещает в него вторую часть, и так до полной передачи всего объема информации. В каждом слоте существует бит, определяющий, свободен или занят данный слот, поле сетевого адреса приемника и адреса передатчика, а также бит признака конца передачи. Время при данном методе дискретизируется, и поэтому конфликты не происходят.

В. Абонент, которому адресована информация, выбирает слоты, содержащие адресованную ему информацию, и устанавливает в принятом

слоте бит подтверждения (еще одно поле слота). И так продолжается до последнего адресованного ему слота.

Г. Передающий абонент получает свои слоты обратно по «кольцу» и освобождает их (помечает как свободные). При этом он сразу же имеет подтверждение приема (из анализа бита подтверждения).

Очевидно, при данном методе передачу могут вести сразу несколько абонентов. При этом совсем не обязательно каждому передающему абоненту занимать соседние слоты: слоты, находящиеся рядом, могут содержать совершенно разную информацию, относящуюся к разным абонентам (рис. 27).

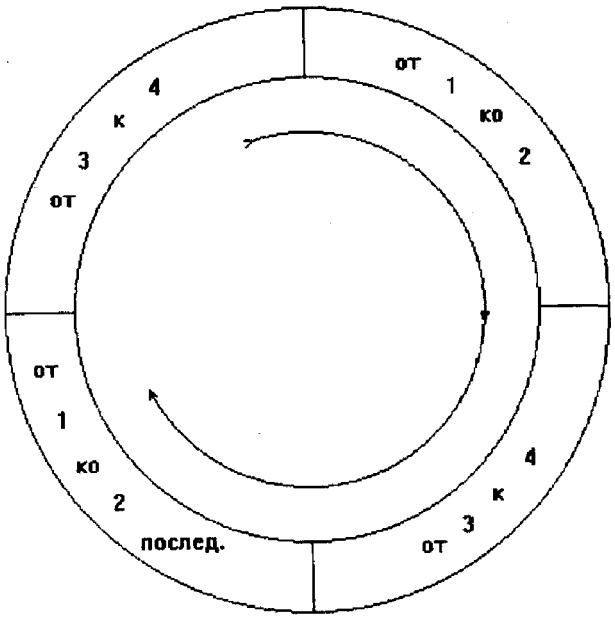


Рис. 27

Как и в случае маркерного метода управления, здесь нужен «центр», который следит за прохождением слотов и восстанавливает их в случае их исчезновения. Преимущество данного метода по сравнению с маркерным состоит в том, что сеть занимается одновременно несколькими абонентами. Время доступа при этом методе гарантировано (в наихудшем случае оно составит время передачи пакета, умноженное на количество абонентов в сети).

Контроль правильности передачи

При передаче информации в локальных сетях возможны ошибки, т.е. искажения передаваемой информации. Эти ошибки необходимо выявлять и исправлять.

Можно выделить несколько основных причин возникновения ошибок.

1. Ошибки из-за плохих контактов в сети, возникающих из-за неисправностей кабеля или неисправности разъема. Данный тип ошибок устраняется осмотром аппаратуры, заменой дефектных частей, очисткой контактов.

2. Ошибки из-за внешних факторов (действие электромагнитных наводок от сильноточных приборов, радиопомехи). Такие ошибки можно устранить снижением излучений источника помех, правильной прокладкой кабеля, выбором типа кабеля (с двойной оплеткой, оптоволоконный), но полной гарантии от них не будет никогда.

3. Ошибки из-за рассогласования кабеля (имеется в виду только электрический кабель). Устраняются подбором согласующих концевых резисторов (терминаторов), согласованием в месте подключения абонентов, разветвлениях кабеля сети и т. д.

4. Ошибки из-за конфликтов в сети (столкновений пакетов). Устраняются выбором метода управления (арбитража), организацией диалога абонентов в пределах сеанса связи (использование пакетов подтверждения, повторные передачи пакетов и т.д.).

Контроль правильности передачи может осуществляться как приемником, так и передатчиком.

1. Контроль передатчиком может быть *побитный*, *побайтный* и *пакетный*.

1. *Побитный метод* сводится к сравнению каждого передаваемого бита с истинным состоянием линии. В этом случае гарантируется стопроцентная проверка правильности.

2. *Побайтный метод* состоит в том, что передача ведется отдельными байтами, а приемник возвращает обратно передатчику каждый полученный им байт. Передатчик сравнивает полученный байт с переданным и на основании этого делает вывод о правильности передачи. Метод выявляет практически 100% ошибок, но может выявить ошибку, которой не было (когда искажается ответный байт при неискаженном переданном байте).

3. *Пакетный метод* применяется тогда, когда передача ведется пакетами, и приемник возвращает обратно передатчику весь принятый им пакет, изменяя в нем только адрес приемника и передатчика. Как и в случае предыдущего метода, здесь выявляется 100% ошибок, но можно обнаружить ошибку, которой не было. Оба эти метода замедляют обмен как минимум в два раза.

2. Контроль приемником в отличие от контроля передатчиком не требует практически никакого дополнительного времени. Идея обнаружения ошибки состоит в следующем: к информации добавляются дополнительные контрольные биты, в которые входит в свернутом виде характеристика всех информационных битов. Приемник проверяет соответствие принятых информационных и контрольных битов и на основании этого делает вывод

о наличии ошибок. Затем он сообщает передатчику, выявил он ошибки или нет, а передатчик при необходимости повторяет свою передачу.

Контроль приемником бывает *побайтный* и *пакетный*.

1. *Побайтный метод* предполагает, что каждый передаваемый байт дополняется битом четности (или нечетности), т.е. в случае, когда количество единиц в передаваемом информационном байте четное, то бит равен «0», а если нечетное, - то «1». Метод может применяться как при байтовой передаче (в сети типа «звезда»), так и при пакетной передаче. Избыточность передаваемой информации здесь довольно велика (добавляется лишний бит на каждые передаваемые 8 бит). Вероятность того, что ошибка не будет обнаружена, также значительна. К этому может привести наличие двух или более неправильных битов в байте, а также одновременное искажение информационного и контрольного битов.

2. *Пакетный метод* сводится к тому, что в конце каждого передаваемого пакета добавляется контрольная сумма (длиной 8, 16 или 32 бита), которая включает в себя информацию о всех информационных битах пакета. Метод подсчета контрольной суммы выбирается так, чтобы, с одной стороны, ее просто было вычислить, а с другой стороны, чтобы она достаточно надежно выявляла ошибки. Обычно используются контрольные суммы трех видов:

а) сумма по модулю 2 всех байтов (слов) пакета;

б) арифметическая сумма всех байтов (или слов) пакета;

в) циклическая контрольная сумма. Этот метод часто называют «циклическим контролем по избыточности» (сокращенно CRC - *Cyclic Redundancy Check*). Смысл его состоит в следующем.

Весь пакет рассматривается как *N*-разрядное двоичное число, где *N* - количество бит в пакете. Для вычисления контрольной суммы это число делится на некоторое постоянное число, выбранное специальным образом. Частное от этого деления отбрасывается, а остаток используется в качестве контрольной суммы.

Если выявлено наличие ошибок, то приемник запрашивает у передатчика повторную передачу пакета. В локальных сетях это оказывается быстрее, чем исправлять ошибки на месте.

Уровни сетевой архитектуры

При обмене информацией между компьютерами сеть выполняет множество функций, которые можно разделить на группы по тем или иным признакам. Пользователю неважно, каким образом осуществляется связь. Он из своей прикладной программы обращается к сетевому ресурсу (например, к удаленному диску или к удаленной задаче) и получает доступ к нему. Как это происходит - пользователю безразлично. Однако при этом

его запрос преобразуется в нужный для сети формат, осуществляется захват сети (арбитраж), передаваемая информация разбивается на пакеты, кодируется, преобразуется в нужные электрические сигналы и передается в сеть. На приемной части происходит обратное преобразование. Для пользователя это должно быть незаметно или, как еще говорят, прозрачно.

Существует модель открытой системы обмена информацией OSI (*Open System Interchange*), которая формирует требования, в том числе и к локальным сетям.

Модель OSI разбивает все функции по взаимодействию открытых систем на семь уровней (это не значит, что разбить их можно только так). Каждый более высокий уровень опирается на все нижестоящие и использует их в своих целях. Самые нижние уровни при этом относятся к аппаратуре связи систем, а верхние – к прикладным программам пользователя (табл. 2).

Таблица 2

Уровни модели взаимодействия открытых систем OSI

7. Уровень приложений
6. Уровень представления данных
5. Сеансовый уровень
4. Транспортный уровень
3. Сетевой уровень
2. Уровень управления линией передачи
1. Физический уровень

Уровни модели OSI получили следующие названия.

1. Физический уровень. К нему относятся функции преобразования передаваемых данных в электрические (или световые) сигналы, распространяющиеся по кабелю, а также обратное преобразование. К этому уровню относятся такие аппаратные средства, как соединительные кабели, приемники и передатчики сетевых адаптеров.

2. Уровень управления линией передачи (канальный уровень) включает в себя функции управления доступом к сети (арбитраж), формирования пакетов (кадров), обнаружения ошибок передачи. Обычно все эти функции реализует аппаратура сетевых адаптеров и их программные драйверы.

3. Сетевой уровень включает в себя функции коммутации и маршрутизации в сложных сетях, состоящих из нескольких простых сетей, а также соответствующую буферизацию данных и регулирование потока пакетов.

4. Транспортный уровень. К нему относятся функции сетевой адресации, нумерации передаваемых пакетов и контроль порядка их следования, а также согласование различных сетей между собой: например, двух локальных сетей, локальной сети и глобальной сети.

5. Сеансовый уровень. К нему относятся функции преобразования имен абонентов в сетевые адреса, управления доступом к сети на основе заданных прав доступа, а также взаимодействия абонентов, участвующих в сеансе связи.

6. Уровень представления данных включает в себя функции трансляции форматов и синтаксиса прикладных программ в форму, удобную для сети, шифрование данных и их сжатие (в случае необходимости).

7. Уровень приложений - это функции поддержки прикладного программного обеспечения конечного пользователя.

Таким образом, уровни 1 и 2 обычно реализуются аппаратно. На этих уровнях определяется физическая скорость передачи и топология сети. Именно к этим уровням относятся названия Ethernet, Arcnet, Token-Ring и т.д., т.е. тип конкретных аппаратных средств. Более высокие уровни не работают напрямую с конкретной аппаратурой, но уровни 3, 4, 5 учитывают ее особенности.

Уровни 3, 4, 5 часто объединяют в отдельную группу, так как они управляют аппаратурой. Эти уровни обеспечивают взаимодействие передающего и принимающего абонентов, формируют то, что называется виртуальным каналом связи (в отличие от физического канала), т.е. канал связи, работающий временно, но который воспринимается пользователем как реальная линия связи. Эти уровни в основном решаются средствами сетевой операционной системы или сетевой оболочки, хотя иногда отдельные функции возлагаются и на аппаратуру.

Уровни 6 и 7 уже не имеют к аппаратуре вообще никакого отношения. Они обслуживаются средствами взаимосвязи прикладных задач. Это позволяет объединять самые разные компьютерные средства от терминалов до больших ЭВМ, дает возможность работать с самыми разными операционными системами (VMS, UNIX, MS-DOS, OS/2, Apple Macintosh) и со всеми сетевыми средствами (Netware, Ethernet, FDDI, Token Ring и т. д.).

Как и любая универсальная модель, модель OSI является избыточной. Она содержит в себе все возможные функции, которые в данной конкретной системе и данной локальной сети могут просто не использоваться. Более того, в конкретной сети может даже не существовать такого разделения функций на уровни, которое предусмотрено OSI, а выделить отдельные функции нередко просто невозможно. Тем не менее модель OSI оказывается очень полезной при выработке правил согласования работы отдельных систем, в частности при соединении нескольких локальных сетей в единую систему.

Взаимодействие "Объект исследования - ЭВМ"

Появление интерфейса радикально изменило принципы построения сложных систем сбора и обработки информации, автоматизации эксперимента - измерительных информационных систем, послужило основой создания измерительно-вычислительных комплексов. Задача настоящего раздела - сообщить сведения, необходимые для понимания того, какие задачи решаются интерфейсом, на каких принципах он строится, что представляет собой его схема, в чем заключаются интерфейсные функции.

Агрегатный принцип построения измерительных систем

Наиболее рациональный принцип построения систем обработки информации, в том числе измерительно-вычислительных комплексов, - модульный принцип построения, или *принцип агрегатирования.* Его сущность заключается в том, что система выполняется как агрегат, состоящий из независимых функциональных блоков - модулей. Каждый модуль имеет конструктивную законченность. В качестве примеров функциональных блоков можно назвать аналого-цифровой и цифро-аналоговый преобразователи, цифровой вольтметр, цифровой частотомер, измерительный генератор и т.п.

Многообразие систем, построенных на агрегатном принципе, достигается путем использования различных сочетаний, комбинаций модулей. Предусматривается возможность наращивания структуры системы в процессе эксплуатации. Иногда модули объединяют в группы, называемые крейтами (например, в системе КАМАК). Управление работой системы осуществляют контроллеры, координирующие и контролирующие действия отдельных устройств.

При построении модульных систем должны быть решены две основные задачи: *совместимости* и *сопряжения* модулей (как между собой, так и с внешними устройствами).

Применительно к измерительным информационным системам различают пять видов совместимости [23]:

- *информационную* - согласованность входных и выходных сигналов модулей по видам и номенклатуре, информативным параметрам, уровням. Для информационного взаимодействия модулей применяют сигналы нескольких разновидностей: информационные, управляющие, программные, адресные, специальные;

- *энергетическую* - согласованность напряжений и токов, питающих модули; линий сети переменного тока, связывающих модули с центральным блоком питания;

- *метрологическую* - сопоставимость результатов измерений, рациональный выбор и нормирование метрологических характеристик модулей, а также согласование входных и выходных цепей;

- *конструктивную* - согласованность конструктивных параметров, механического сопряжения модулей при совместном использовании, а также согласованность эстетических требований. При этом система или ее часть должна представлять единое целое;

- *эксплуатационную* - согласованность характеристик модулей по надежности и стабильности, а также характеристик, определяющих влияние внешних факторов.

- Преимущества принципа модульности наиболее полно проявляются, если любые модули системы можно состыковать и объединить в систему без конструктивных изменений. Для этого должно быть унифицировано сопряжение между модулями. Такое сопряжение модулей между собой и с устройствами обработки достигается посредством интерфейса.

Классификация интерфейсов

Интерфейс (от англ. *interface* - сопрягать, согласовывать) - это стандарт на сопряжение двух блоков, определяющий число сопрягаемых линий, назначение каждой линии, содержание информации, передаваемой по каждой линии, направление передачи, кодировку информации, временные и амплитудные характеристики сигналов по каждой линии, типы разъемов.

Интерфейсом называется также и само сопряжение, выполненное по стандарту. Если сопрягаются программные блоки, то определение остается в силе и для них, кроме части определения, касающейся электрических характеристик сигналов.

Существует ряд стандартов на разработку технических средств сопряжения [18, 19, 20, 21]. Все множество интерфейсов в зависимости от назначения можно разделить на три типа: машинные, системно-модульные и системно-приборные.

Под машинными интерфейсами подразумевают такие интерфейсы, которые решают задачу соединения центрального процессора ЭВМ данного типа с другими ее функциональными блоками, а также подключения периферийных устройств, в том числе устройств связи с объектом. Особенности ЭВМ практически полностью определяют как электрические и конструктивные характеристики интерфейса, так и принципы его функционирования.

Системно-модульные интерфейсы решают задачу унификации сопряжения модулей, предназначены для работы в системе, что определяет и их конструктивные особенности. Модули, выполненные с учетом применения подобного интерфейса, как правило, не рассчитаны на использование в качестве автономных приборов, которые могут работать отдельно, вне системы.

Интерфейсы, предназначенные для измерительной техники, иногда сокращенно называют измерительными интерфейсами, подобно тому, как генераторы сигналов для измерений характеристик различных испытуемых объектов называются измерительными генераторами. Измерительные интерфейсы, решая задачу сопряжения, обеспечивают пять перечисленных видов совместимости и взаимодействие ЭВМ с измерительной и периферийной аппаратурой, которая также взаимодействует с устройствами сбора, накопления, регистрации и обработки информации.

Стандартные интерфейсы можно классифицировать в зависимости от схемы соединения модулей между собой и с центральным модулем (устройством обработки) системы. Различают три основные схемы соединений: каскадную, радиальную и магистральную.

Каскадная схема. Эта схема (рис. 28*а*) применима, когда общий поток информационных сигналов в каждый данный момент времени связывает между собой только один объект исследования, один источник испытательных сигналов и один измерительный прибор (на рисунке надпись $И_к$ обозначает интерфейс каскадный).

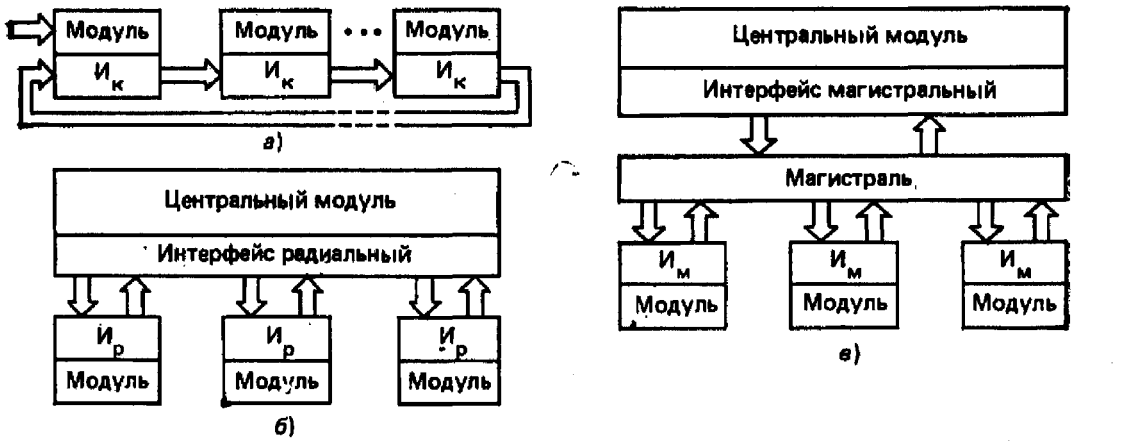


Рис. 28

Радиальная схема. Как видно из рис. 28*б*, такая схема характерна для случая, когда к центральному модулю (устройству обработки измерительной информации) необходимо подключать несколько модулей. При этом модули присоединяются непосредственно, поскольку центральный модуль располагает достаточным числом каналов для обмена данными.

Магистральная схема. В последнее время наибольшее распространение получили интерфейсы, в которых информация передается от одного

устройства (модуля) к другому по многопроводной магистрали (см. рис. 28*в*), соединяющей все устройства. При каждом обращении к центральному модулю подключается только тот модуль, адрес которого вызывается программой. При этом можно отметить следующие достоинства магистральной схемы сопряжения:

- создание гибких измерительных комплексов, которые можно легко наращивать и модернизировать;
- создание банков совместимых компонентов системы - модулей, контроллеров, разрабатываемых в различных организациях и странах;
- независимость структуры комплекса от типа ЭВМ (смена ЭВМ приводит к необходимости замены только одного элемента системы - контроллера).

Для соединения функциональных блоков или приборов между собой служат цепи, называемые линиями интерфейса. Группу линий, предназначенных для выполнения определенной функции в программно-управляемом процессе передачи данных, называют шиной. Назначение отдельных шин и линий, их перечень и взаимное расположение (топология) играют основополагающую роль при рассмотрении работы интерфейса.

Интерфейс МЭК - 625

Практика развития автоматизированных систем показала, что внешний информационный обмен между отдельными устройствами удобно производить побайтно, т.е. по 8 бит одновременно, а скорость передачи должна регулироваться самими устройствами, участвующими в обмене. В итоге возникла концепция интерфейса с бит-параллельным, байт-последовательным, асинхронным способом обмена информацией.

В настоящее время эта концепция интерфейса известна под названием МЭК-625, или магистраль IEEE-488, или интерфейсная магистраль Hewlett-Packard (HP-IB). Встречается также название интерфейса GPIB (General Purpose Interface Bus—интерфейсная шина общего назначения). У нас в стране этот интерфейс известен как «Канал общего пользования» (КОП) и регламентируется [22].

Стандарт приборного интерфейса определяет магистраль, по которой происходит обмен информацией между приборами или устройствами, принципы взаимодействия, синхронизации, управления, условия информационной и электрической совместимости. Требования конструкционной совместимости касаются только разъемов кабелей интерфейсной магистрали для соединения приборов или устройств.

Интерфейс разработан для программируемых и непрограммируемых электронных измерительных приборов. Он ориентирован на сопряжение устройств, располагаемых относительно друг друга на расстоянии до 20 м.

Обеспечивает унификацию конструктивных, электрических и функциональных характеристик независимо изготовленных приборов, что создает возможность сопряжения их, организации измерительной системы. Интерфейс позволяет иметь в системе приборы различной сложности, допускает прямой обмен информацией между ними, дистанционное и местное управление приборами. Описываемый интерфейс [20, 21, 23] относится к магистральным.

Структура и принцип действия интерфейса. На рис. 29 показана схема интерфейса. Он представляет собой пассивную шестнадцатиразрядную двунаправленную систему связи, называемую магистралью, к которой можно подключить параллельно до 15 приборов (и в том числе контроллер).

Функционально линии, образующие магистраль, группируются в три шины: данных, синхронизации (согласования передачи) и управления.

Конструктивно интерфейс состоит из кабеля, разъемов и печатных плат. Функция кабеля заключается в параллельном соединении всех устройств между собой. Соединение осуществляется таким образом, что данные от конкретного устройства могут передаваться либо одному, либо нескольким другим устройствам, входящим в систему.

Узлы устройств, с помощью которых производится обмен информацией, называются интерфейсными картами (ИКАР). Интерфейсные карты программно-управляемых приборов выполняются по двум вариантам:

- в виде схем, в том числе БИС, реализованных и конструктивно оформленных внутри прибора как его составная часть, с установкой стандартного разъема на задней панели прибора. Этот вариант преимущественно применяется в новых приборах, выпускаемых по стандарту МЭК;

- в виде отдельно выполненных интерфейсных модулей, подключаемых к серийно выпускаемым или ранее выпущенным цифровым приборам и устройствам. Это, по существу, адаптеры, т.е. переходные устройства между выходом прибора и стандартным входом в магистраль приборного интерфейса.

Рассмотрим состав и назначение каждой из трех шин.

Шина данных (Data Bus) состоит из восьми линий, обозначаемых DIO (Data input/output) с соответствующим номером линии, например DIO 3, или ЛД (линия данных) - соответственно ЛД3. По этим линиям осуществляется обмен информацией бит-параллельным, байт-последовательным способом. Иначе говоря, по восьми линиям передаются данные в форме параллельных битов и последовательных байтов. Шина данных служит для передачи (приема) основных данных: результатов измерений, адресных, программных, управляющих данных и данных состояний.

Обмен информацией может происходить между передающими («говорящими») приборами, принимающими («слушающими») приборами или между контроллером и подчиненными приборами. По характеру взаимодействия модулей с шиной стандарт разделяет их на четыре группы: уст-

ройство-контроллер, устройство передающее и принимающее, устройство только передающее, устройство только принимающее (рис. 29**).**

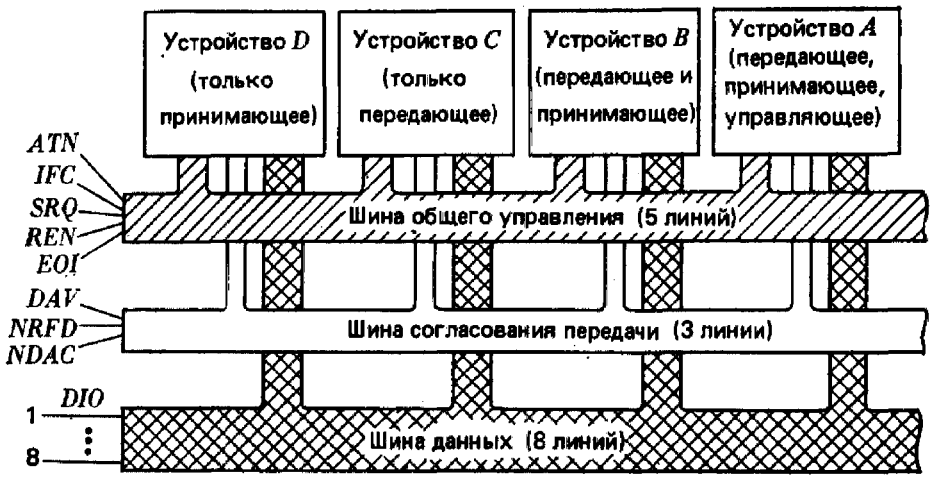


Рис. 29

Каждое устройство, имеющееся в составе системы (измерительный прибор, контроллер, вспомогательный модуль), должно выполнять, по крайней мере, одну из функций: быть «говорящим», «слушающим» или управляющим. Данные от «говорящего» устройства передаются через шину к другим устройствам, например к «слушающему» устройству, принимающему информацию «говорящего». Некоторые устройства могут выполнять обе функции. Например, программируемый мультиметр, который принимает управляющие команды как «слушающий» прибор и передает результаты измерений (данные) в качестве «говорящего» прибора. Особенность интерфейсной шины такова, что одновременно может работать несколько «слушающих» приборов, но только один - «говорящий».

Назначение управляющего устройства - контроллера - организация взаимодействия модулей системы. Команды контроллера указывают адрес модуля; какой модуль должен передавать данные, а какой - принимать, а также характер и последовательность выполнения других операций. Вычислительный контроллер, который строится на основе микропроцессора, выполняет следующие функции: определяет программу измерений, задает согласно ей виды измерений определенным приборам, управляет процедурой измерений, интерпретирует их результаты. В составе измерительной системы, объединяемой интерфейсом, может быть устройство, способное и «говорить», и «слушать», и управлять. Таким устройством является микроЭВМ.

Линии DIO (1... 8) или ЛД (1... 8) - линии ввода-вывода данных, образующие шину данных, - служат для передачи информации, представляющей собой цифровые данные, адреса модулей и универсальные команды.

Шина согласования передачи (Data byte transfer control bus), которую иначе называют шиной синхронизации, объединяет три линии (рис. 29),

обозначаемые DAV (Data valid) или СД (сопровождение данных), NRED (Not ready for data) или ГП (готов к приему) и NDAC (Not data accepted) или ДП (данные приняты). По этим линиям передаются сигналы согласования, подтверждающие соответствие состояний приборов, что необходимо для обмена информацией, т. е. управления передачей каждого байта информации по шине данных от контроллера или «говорящего» прибора к одному или нескольким «слушающим» приборам. Эти сигналы иногда называют квитирующими.

Важной характеристикой интерфейса является вид обмена данными между модулями: синхронный или асинхронный. Интерфейс МЭК в основном рассчитан на асинхронный обмен информацией.

Для асинхронного обмена основополагающей является процедура установления соответствия. Она предполагает управляемую передачу сигналов, подтверждающих взаимное соответствие состояний приборов, участвующих в информационном объеме (метод квитирования). Возможны два варианта организации указанной процедуры. Сущность первого варианта заключается в следующем: когда завершена подготовка данных для передачи, «говорящий» прибор устанавливает флаг (сигнал готовности данных) и ждет готовности «слушающего» прибора, который должен принять эти данные. При втором варианте первоначально устанавливает флаг «слушающий» прибор, что свидетельствует о его готовности принять сообщение и ожидании готовности «говорящего» прибора к передаче. В интерфейсе МЭК принят второй вариант установления соответствия.

Названия линий, образующих шину согласования передачи, определяются передаваемыми по ним сигналами. Линия DAV (достоверность информации) служит для сигнала, указывающего на наличие, достоверность информации на шине данных, или, иначе, для установления флага только «говорящего» прибора. По линии NRED (не готов к приему информации) устанавливается флаг готовности только «слушающего» прибора; она является общей для всех принимающих приборов. Линия NDAC (информация не принята) предназначена для передачи сигнала-квитанции «слушающих» приборов: наличие низкого уровня напряжения на ней свидетельствует, что самый медленно действующий из «слушающих» приборов еще не принял информации.

Шина общего управления (General interface management bus attention) состоит из пяти линий (см. рис. 29). По ним передаются управляющие сигналы, которые циркулируют между контроллером и другими приборами, подключенными к интерфейсу. Кратко охарактеризуем функцию каждой линии.

Линия, обозначаемая ATN (Attention—внимание) или УП (управление), отведена для команды, посылаемой контроллером. Наличие такой команды (низкий уровень напряжения на линии) определяет, что все остальные устройства переходят в режим ожидания и только контроллер яв-

ляется «говорящим» прибором. При этом по шине данных передаются адреса или универсальные команды. Когда на линии устанавливается высокий уровень напряжения, то «говорят» или «слушают» те приборы, адреса которых были переданы за время противоположного состояния линии.

По линии, обозначаемой IFC (Interface clear - очистка интерфейса) или ОИ, передается сигнал контроллера, приводящий схему интерфейса и все приборы в начальное состояние. Эта команда, используемая при запуске интерфейса и устанавливающая низкий уровень напряжения в линии, прекращает передачи информации по шине данных.

Линия, обозначаемая SRQ (Service request - запрос на обслуживание) или ЗО**,** является общей для всех приборов и переходит в состояние, характеризуемое низким уровнем напряжения, когда какой-либо из приборов, подключенных к интерфейсу, посылает в контроллер сигнал запроса на обслуживание, т.е. «требует» прерывания текущего обмена в магистрали и приоритетного обслуживания данного прибора контроллером.

Назначение линии, обозначаемой REN (Remote enable - разрешено дистанционное управление) или ДУ (дистанционное управление), - передача контроллером сигналов программного управления приборами. Когда по команде в линии устанавливается низкий уровень напряжения, приборы переключаются с «местного» управления (с передней панели) на дистанционное.

Линия, обозначаемая EOI (End of identify - конец обработки, конец идентификации) или КП (конец передачи), служит для посылки команды, указывающей окончание передачи сообщений по шине данных. Низкий уровень напряжения, устанавливающийся на линии синхронно с передачей последнего байта данных, сигнализирует о том, что данных больше нет. Если низкий уровень напряжения устанавливается контроллером при параллельном опросе, то конец передачи интерпретируется как идентификация.

Пример измерительной системы на основе интерфейса МЭК. Изображенная на рис. 30 структурная схема представляет сравнительно простой вариант измерительной системы. В ее составе приборы и устройства, выполняющие различные функции.

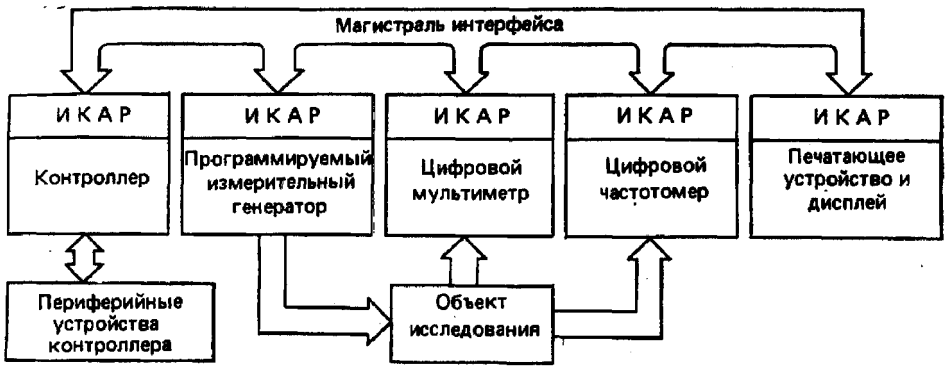


Рис. 30

Программируемый измерительный генератор - «слушающий» прибор. От интерфейса он только принимает адрес и команды, определяющие программу его работы, но данных в интерфейс не посылает. «Слушающими» устройствами являются также печатающее устройство и дисплей, фиксирующие, отображающие результаты измерений. В эти устройства из интерфейсной магистрали поступают адреса, данные измерений, команды. Цифровой мультиметр и цифровой частотомер - приборы и «слушающие», и «говорящие». Они получают адреса и команды из интерфейса и передают в него данные измерений напряжения, тока, сопротивлений резисторов (мультиметр), частоты и интервалов времени (частотомер).

Работой системы управляет контроллер, с которым связаны его периферийные устройства, например клавиатура. Он может осуществлять обработку данных, полученных результатов измерений.

В начальный момент по команде IFC (очистка интерфейса), посылаемой контроллером, интерфейсные части всех приборов системы устанавливаются в исходное состояние. Следующая команда REN переводит приборы в режим дистанционного управления (возврат к местному управлению возможен в любой момент и выполняется по соответствующей команде контроллера или по внутренней приборной команде). Управление системой сводится главным образом к передаче адресов и команд, поступающих из контроллера (при низком уровне напряжения в линии ATN). Его основная задача - указание адресов «говорящих» и «слушающих» приборов для обмена данными между ними. После формирования конкретного адреса контроллер деблокирует шины выбранного «говорящего» прибора (установив высокий уровень напряжения в линии ATN), в результате чего создается возможность передачи данных от него выбранному «слушающему» прибору. По окончании передачи данных возобновляется управляющая функция контроллера: он может формировать очередной адрес прибора и команды. Контроллер может также активно участвовать в обмене данными, играя роль «говорящего» прибора (передача программных данных) или «слушающего» прибора (прием данных измерений).

Интерфейс позволяет проводить параллельный или последовательный опрос абонентов системы (независимо от описанного процесса ее функционирования). Цель опроса - обнаружение таких ситуаций внутри системы, которые требуют определенных действий. При последовательном опросе контроллер, получив соответствующее требование, формирует команду SRQ (вызова прибора на обслуживание) и устанавливает вид обслуживания. Подобный опрос может быть программируемым, т.е. проводиться согласно соответствующей подпрограмме контроллера, обращение к которой определяется появлением требования на обслуживание. Параллельный опрос проводится независимо от поступления заявок на обслуживание.

Система КАМАК (CAMAC)

Система КАМАК - международная система модульной электроники, объединенная единым интерфейсом, ориентированная на работу с различными средствами вычислительной техники и позволяющая производить автоматические измерения, сбор данных и управление объектом или процессом [24].

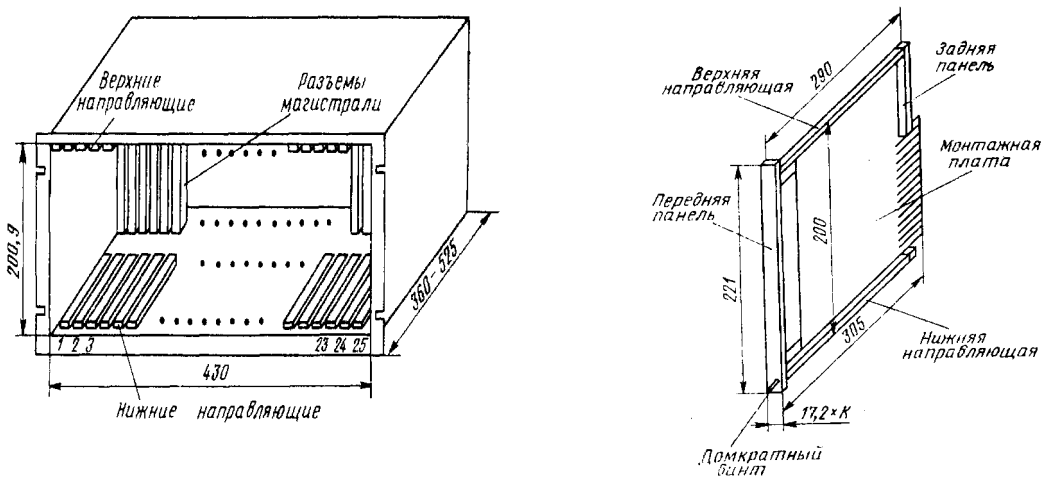
CAMAC - Computer Application to Measurement and Control - является не только интерфейсом. Это и целая система модулей, объединенных единым интерфейсом и едиными конструктивными параметрами, предназначенная для сбора, накопления, преобразования и обработки измерительных и управляющих сигналов в системе автоматизации эксперимента.

Система КАМАК позволяет реализовать основные принципы построения АСНИ: модульность, программную управляемость компонентов и магистральную организацию системы. Стандарт КАМАК [26] определяет основные положения, которыми следует руководствоваться при разработке, производстве и применении отдельных компонентов и всей системы в целом. Он устанавливает требования к механическим характеристикам, источникам питания и электрическим сигналам, которые обеспечивают совместимость между различными элементами системы. Пользователям предоставлена возможность самим дорабатывать и расширять систему модулей. Система свободна от патентных ограничений, открыта для расширения и совершенствования с публикацией всех новых разработок.

Механические характеристики

Стандарт определяет размеры основных элементов системы - крейта и модуля с целью их конструктивной совместимости.

Крейт (каркас, корзина) (рис. 31*а*) представляет собой конструктив со стандартными внутренними размерами окна: ширина 430 мм, высота 200.9 мм и глубина 360 мм. Глубина крейта может быть увеличена до 525 мм для установки в тыльной части крейта блока питания. Крейт должен содержать не более 25 станций для установки вставных модулей с шагом 17.2 мм. Каждая станция имеет верхнюю и нижнюю направляющие, по которым вдвигаются вставные модули. Все станции крейта пронумерованы слева направо от 1 до 25. Левые 24 станции называются нормальными, последняя правая 25-я станция называется управляющей (контроллер крейта). Разъемы крейта соединяются определенным образом многопроводным каналом связи - магистралью крейта. Распайка разъемов всех нормальных станций одинакова и отличается от распайки разъема управляющей станции.



а) б)

Рис. 31

Вставной модуль (рис. 31*б*) представляет собой блок высотой 221.5, глубиной 305 и шириной 17.2 х К мм, где К=1, 2, 3, ... Модуль состоит из следующих основных частей:

- передней панели, на которой могут располагаться органы ручного управления и связи с внешними устройствами; внизу на передней панели модуля расположен домкратный винт;
- верхнего и нижнего ребер, которые скользят по направляющим установочной станции крейта;
- монтажной платы с 86-контактной вилкой разъема. Расположение платы, крепежных деталей и элементов разъема оговорено стандартом.

Если модуль занимает более одной станции (K > 1), то он может иметь до K-разъемов для связи с магистралью крейта.

Все модули КАМАК делятся на функциональные и управляющие. Функциональные модули реализуют определенные функции, необходимые для экспериментатора, в соответствии с командами и сигналами, поступающими по магистрали. Управляющие модули (контроллеры) управляют работой функциональных модулей путем генерирования команд на магистраль. Управляющий модуль всегда занимает 25-ю управляющую станцию и хотя бы одну из нормальных станций (обычно 24-ю). Состав и расположение функциональных модулей в крейте определяется только требованиями экспериментатора.

Энергетические характеристики

Стандарт регламентирует значения напряжений источников питания в крейте, максимальные токовые нагрузки на шинах питания в крейте и через контакты его разъемов, максимальные мощности, рассеиваемые в одной станции и в крейте в целом.

Шина питания магистрали крейта включает линии для подключения обязательных источников, дополнительных и резервные линии.

Полная мощность, рассеиваемая в крейте, не должна превышать 200 Вт, что соответствует 8 Вт на одну станцию. При использовании принудительной вентиляции мощность, рассеиваемая в одной станции, может быть увеличена до 25 Вт, однако общая мощность в крейте не должна превышать 200 Вт.

Характеристики сигналов

При работе функциональных модулей в системе они обмениваются сигналами с внешним, по отношению к крейту, оборудованием экспериментальной установки, а также между собой и контроллером по магистрали крейта. Стандартом предусмотрены информационная совместимость элементов системы по внешним и внутренним сигналам крейта. Для обеспечения совместимости функциональных модулей по внешним сигналам в системе КАМАК стандартизованы уровни входных сигналов, принимаемых модулями от экспериментальной установки, и уровни выходных сигналов, передаваемых во внешние устройства. Стандартизация касается как логических, так и аналоговых сигналов.

Уровни логических сигналов «1» и «0», передаваемые по несогласованным линиям связи, должны соответствовать уровням микросхем ТТЛ, по согласованным линиям - ЭСЛ. Характеристическое сопротивление линии для согласованных сигналов - 50 Ом.

Полярность аналоговых сигналов на входах и выходах модулей должна быть положительной, диапазон изменения 0,5 В. Для биполярных сигналов изменение амплитуды рекомендуется в пределах от —5 до +5 В.

При необходимости включения в состав оборудования экспериментальной установки устройств, требующих других уровней электрических сигналов, допускается использование сигналов с характеристиками, соответствующими сигналам подключаемого оборудования.

Уровни логических сигналов, передаваемых по магистрали крейта, выбраны совместимыми с уровнями ТТЛ ИС. Значения сигналов логической 1 и логического 0 являются инверсными относительно уровней напряжения, предоставляющих эти сигналы.

Магистраль крейта

Обмен информацией между контроллером и функциональными модулями происходит по магистрали крейта. Магистраль крейта (рис. 32) представляет собой пассивный многопроводный канал связи, соединяющий определенным образом контакты магистральных разъемов отдельных станций.

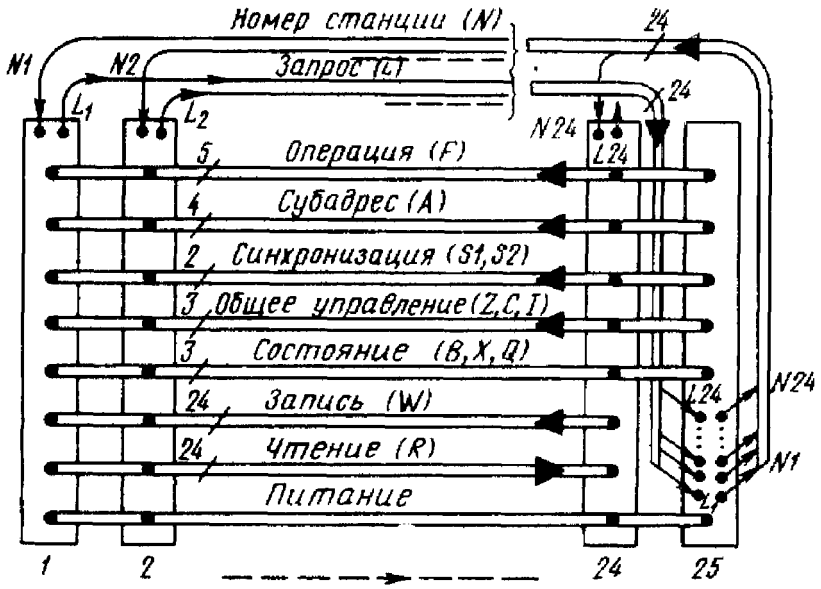


Рис. 32

По типу соединений линии связи магистрали крейта можно разделить на три вида:

- сквозные линии, соединяющие одноименные контакты всех разъемов крейта;

- линии, соединяющие одноименные контакты только нормальных станций (1 - 24);

- индивидуальные линии, соединяющие определенный контакт одной из нормальных станций с определенным контактом управляющей станции.По функциональному назначению все линии связи магистрали крейта можно разделить на 6 шин.

Шина данных состоит из 24 линий *W1 - W24* (*запись*) и 24 линий *R1 - R24 (чтение)*, объединяющих одноименные контакты только нормальных станций. По линиям *W* контроллер может пересылать в функциональный модуль информацию, которая предназначена для управления работой модуля или внешнего оборудования. По линиям *R* функциональный модуль может передавать контроллеру как экспериментальные данные, так и служебную информацию. Обмен данными может выполняться словами до 24 двоичных разрядов.

Шина адреса состоит из 24 индивидуальных линий N_i *(номер станции)* и 4 сквозных линий *A1, A2, A4, A8* (*субадрес*). Сигналы на линиях N_i служат для адресации нормальных станций в крейте. Такой способ адресации модулей называют *географической* адресацией, поскольку адрес конкретного модуля определяется его расположением в крейте. Сигналы на линиях субадреса позволяют выбрать один из шестнадцати элементов внутри модуля - от *A(0)* до *A(15).*

Шина управления состоит из 5 сквозных линий *F1, F2, F4, F8, F16* (*операция*) и 3 линий *Z, C, I* (*общее управление*). Сигналы на линиях *F* по-

зволяют выбрать одну из 32 команд от *F(0)* до *F(31)*, которая должна быть выполнена адресованным модулем. Коды операций, используемых в модуле, должны полностью декодироваться в модуле. Сигналы общего управления *Z* (*zero*), *C* (*Clear*), *I* (*inhibit*) позволяют подавать на магистраль безадресные сигналы управления, предназначенные для всех модулей крейта. Сигнал *пуск* (Z), служащий для начального запуска системы, позволяет установить все регистры данных, управляющие регистры и отдельные триггеры в определенное начальное состояние. Сигнал *сброс (C)* служит для установки в исходное состояние выбранных разработчиком регистров и отдельных триггеров. Разница в действии сигналов *Z* и *C* заключается в том, что сигнал *Z* должен обязательно использоваться разработчиком модулей, а сигнал *C* может применяться по усмотрению разработчиков. Сигнал *I* (*запрет*) может запрещать любые действия в модуле, запрет которых с помощью этого сигнала предусмотрен разработчиком. Сигнал *I = 1* может вырабатываться контроллером крейта, функциональными модулями или вводиться от внешнего источника. Сигнал *запрет* обязательно вырабатывается контроллером при генерации сигнала *пуск*.

Шина состояния, состоящая из 3 сквозных линий Q, *X, B* (*состояние*) и 24 индивидуальных линий L_i *(запрос),* служит для передачи статусной информации по магистрали. Сигнал *X* (*команда принята*) является обязательным ответом модуля на любую адресованную ему команду *F(0) - F(3l).* Если команда принята модулем и может быть выполнена им, модуль сообщает об этом сигналом *X = 1.* Ответ *X = 0* означает наличие неисправности в системе: модуль не установлен в адресуемой станции, отсутствует питание, внешнее оборудование к модулю не подключено или не включено его питание, и, наконец, контроллер подает команду, использование которой в модуле не предусмотрено. Сигнал на линию *Q* (*ответ*) может вырабатываться модулем в ответ на любую адресуемую команду для указания состояния выбранного элемента модуля. Сигнал *B (занято*) вырабатывается контроллером крейта и сопровождает любые действия на магистрали, инициируемые контроллером. Значение *B = 1* сообщает всем функциональным модулям о том, что магистраль занята. Сигналы *L-запросов* используются функциональными модулями для сообщения контроллеру о необходимости выполнения определенных действий по их обслуживанию. Модуль вырабатывает сигнал L = 1 асинхронно по отношению к операциям на магистрали, когда он готов к обмену информацией с контроллером крейта.

Шина синхронизации состоит из двух входных линий *S1* (*строб1*) и *S2* (*строб2*). Сигналы *S1* и *S2* предназначены для синхронизации выполнения адресуемых и безадресных операций в модулях или контроллере крейта. *Строб1* разрешает выполнять прием данных с линий *W* адресуемому модулю и контроллеру с линий *R,* Q, *X* или начинать действия, которые не вызывают изменения состояния линий магистрали крейта. *Строб2* исполь-

зуется для начала действий, которые могут вызывать изменение состояния линий магистрали.

Шина питания включает 12 сквозных линий для подключения обязательных и дополнительных источников питания, а также 2 резервные линии для последующего развития системы КАМАК.

На разъемах магистрали крейта кроме контактов, которые задействованы соответствующими линиями связи, имеются свободные контакты *P1-P5* на разъемах нормальных станций и *P1-P7* на разъеме управляющей станции, которые предназначены для свободного использования. Контакты *P1* и *P2* всех нормальных станций должны быть связаны двумя свободными сквозными линиями.

Структура функциональных модулей КАМАК

Широкое применение аппаратуры КАМАК в научных исследованиях и промышленности привело к необходимости разработки функциональных модулей как общего применения, так и специализированных. В настоящее время в мире разработано более 2 000 наименований модулей КАМАК. Независимо от назначения в каждом функциональном модуле можно выделить следующие элементы (рис. 33):

- функциональную часть, определяющую назначение модуля и его технические характеристики;
- интерфейсную часть, обеспечивающую возможности работы модуля в системе;
- органы ручного управления и связи с объектом, расположенные на лицевой панели модуля.

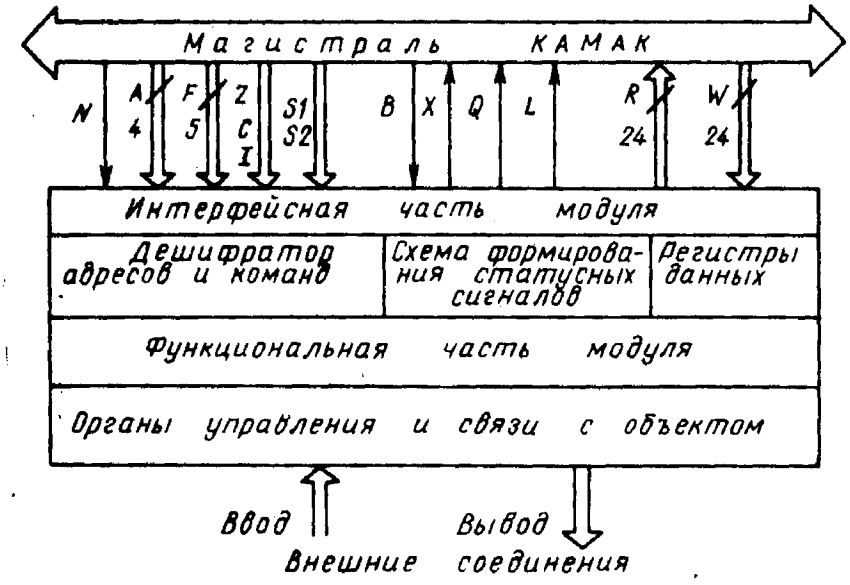


Рис. 33

Все существующее многообразие типов модулей целиком определяется различным назначением их функциональных частей. Интерфейсная

часть модуля должна включать: дешифратор адресов и команд *N, A, F*, схемы формирования статусных сигналов *X, Q*, *L* и регистры данных.

Схема дешифратора адресов и команд позволяет декодировать: сигналы линий субадреса *A8, A4, A2, A1,* операций *F16, F8, F4, F2, F1* и сформировать требуемую команду для любого элемента модуля в виде единого сигнала, поступающего на его исполнительные устройства.

Схема формирования статусных сигналов позволяет модулю генерировать:

- сигнал на линию *X* о возможности выполнения адресованной ему команды;
- сигнал на линию *L* о необходимости обслуживания модуля;
- сигнал на линию *Q*, показывающий состояние части модуля, выбранного командой.

В ответ на любую адресуемую команду, которую модуль может выполнить, он должен сообщить сигналом *X = 1* (команда принята) на магистраль крейта.

Обмен данными между функциональным модулем и контроллером крейта может происходить как по инициативе контроллера, так и по запросу модуля. В том случае, когда инициатором обмена является функциональный модуль, он при готовности к обмену данными сообщает об этом контроллеру крейта сигналом *L = 1*. В другом случае контроллер проверяет состояние модуля, считывая значение сигнала с линии Q, и осуществляет обмен данными с модулем, если последний находится в состоянии готовности (*Q = 1*).

Функциональные модули, устанавливаемые в крейте и представляющие собой измерительные и управляющие элементы, связанные с экспериментальной установкой, взаимодействуют через магистраль крейта с КК (только с ним), а через него с ЭВМ. Такой принцип, когда система КАМАК представляет собой внешнее устройство ЭВМ, позволяет создавать гибкие и легко перестраиваемые комплексы. При изменении в таком комплексе типа ЭВМ производится замена только КК. Промышленностью выпускаются различные КК для различных типов ЭВМ.

Интерфейс RS-232

RS-232 - рекомендованный стандарт *(RS - Recommended Standard)* Ассоциации электрической промышленности, определяющий последовательный коммуникационный интерфейс между устройствами приема и передачи данных. Число 232 - исходный серийный номер данного стандарта.

Последовательный интерфейс может использоваться для большинства периферийных устройств: плоттер, удаленный принтер, мышь, модем и т.д. Европейским аналогом RS-232 являются два стандарта, разработанные

CCITT (Международный консультативный комитет по телеграфии и телефонии), - V24 (механические характеристики) и V28 (электрические характеристики). Хотя первоначально RS-232 был предназначен для связи центральной машины с терминалами, его простота и богатые возможности обеспечили ему более широкое применение.

Основным преимуществом последовательной передачи является возможность пересылки данных на большие расстояния. Конкретные значения максимальных расстояний между блоками зависят от скорости обмена и могут составлять десятки метров. Скорость последовательной передачи данных оценивается в бодах (бит полезной информации в секунду). Обычно передача данных осуществляется на нескольких дискретных скоростях: 2400, 4800, 9600 и 19200 бод. Чем выше скорость обмена, тем выше требования к используемому кабелю.

Немаловажно и то, что в простейшем случае для приема и передачи используются только три сигнала: TxD (*Transmit Data* - передача данных), RxD (*Receive Data* - прием данных) и GND (*Graund* - Земля). При обмене данными могут использоваться различные протоколы - правила обмена - от простейшего, упомянутого выше и работающего только с тремя сигналами интерфейса, до более сложных, использующих, например, пару квитирующих (*handshake*) сигналов RTS-CTS. Различные программы могут применять различные протоколы обмена.

Все оборудование, соединяемое по RS-232 протоколу, разделяют на DCE (*Data Communication Equipment* - оборудование передачи данных) и DTE (*Data Terminal Equipment* - терминальное оборудование), хотя различие между ними состоит только в направлении используемых сигналов. Так, если сигнал для DTE является входным, то для DCE этот же сигнал будет выходным, и наоборот. Терминальное оборудование обычно оснащено разъемом со штырьками, а связное - разъемом с отверстиями.

Ниже приводится назначение линий 25- и 9-контактного разъема для интерфейса RS-232C и описание их функций.

Таблица 3

Номер контакта (9-Pin)	Сигнал	Направление	Полное название
1	FG	---	Основная (или защитная земля)
2 (3)	TD(TXD)	К DCE	Передаваемые данные
3 (2)	RD(RXD)	К DTE	Принимаемые данные
4 (7)	RTS	К DCE	Запрос передачи
5 (8)	CTS	К DTE	Сброс передачи
6 (6)	DSR	К DTE	Готовность модема
7 (5)	SG	---	Сигнальная земля
8 (1)	DCD	К DTE	Обнаружение несущей данных
9	---	К DTE	(Положительное контрольное напряжение)
10	---	К DTE	(Отрицательное контрольное напряжение)
11	QM	К DTE	Режим выравнивания
12	SDCD	К DTE	Обнаружение несущей вторичных данных

Номер контакта (9-Pin)	Сигнал	Направление	Полное название
13	SCTS	К DTE	Вторичный сброс передачи
14	STD	К DCE	Вторичные передаваемые данные
15	TC	К DTE	Синхронизация передатчика
16	SRD	К DTE	Вторичные принимаемые данные
17	RC	К DTE	Синхронизация приемника
18	DCR	К DCE	Разделенная синхронизация приемника
19	SRTS	К DCE	Вторичный запрос передачи
20 (4)	DTR	К DCE	Готовность терминала
21	SQ	К DTE	Качество сигнала
22 (9)	RI	К DTE	Индикатор звонка
23	---	К DCE	(Селектор скорости данных)
24	TC	К DCE	Внешняя синхронизация передатчика
25	---	К DCE	(Занятость)

Примечания:

Линии (контакты) 11, 18 и 25 обычно считаются незаземленными.

Линии 9 и 10 используются для контроля отрицательного (MARK) и положительного (SPASE) уровней напряжения.

Иногда отдельные фирмы используют запасные линии RS-232C для контроля или специальных функций, относящихся к конкретной аппаратуре (по неиспользуемым линиям подают даже питание или аналоговые сигналы).

Сигналы интерфейса RS-232C подразделяются на следующие классы.

Последовательные данные (например, TXD, RXD). Интерфейс RS-232C обеспечивает два независимых последовательных канала данных: первичный (главный) и вторичный (вспомогательный). Оба канала могут работать в дуплексном режиме.

Управляющие сигналы квитирования (например, RTS, CTS). Сигналы квитирования - это средство, с помощью которого обмен сигналами позволяет DTE начать диалог с DCE до фактических передачи или приема данных по последовательной линии связи.

Сигналы синхронизации (например, TC, RC). В синхронном режиме (в отличие от более распространенного асинхронного) между устройствами необходимо передавать сигналы синхронизации, которые упрощают контроль целостности сигнала в целях его декодирования.

На практике вспомогательный канал RS-232C применяется редко, и в асинхронном режиме из 25 линий обычно используются только 9.

Последовательный порт работает асинхронно - данные передаются без тактового сигнала. В этом случае незначительное различие скоростей приема и передачи не влияет на качество обмена.

В тех случаях, когда RS-232 не удовлетворяет по скорости и дальности передачи, его можно заменить на подобные (по методу обмена) RS-485

и RS-422. В табл. 4 представлены сравнительные характеристики этих интерфейсов.

Таблица 4

	RS-232	RS-422	RS-485
Соединения	Одиночный провод	Одиночный провод/много соединений допустимо	Много соединений допустимо
Количество устройств	1 передатчик 1 приемник	5 передатчиков 10 приемников на 1 передатчик	32 передатчика 32 приемника
Вид протокола	дуплексный	дуплексный	полудуплексный
Макс. длина провода	~15.25 м при 19.2Kbps	~1220 м при 100Kbps	~1220 м при 100Kbps
Макс. скорость передачи	19.2Kbps для 15 м	10Mbps для 15 м	10Mbps для 15 м
Двоичная 1	-5В мин. -15В макс.	2В мин. (В>А) 6В макс. (В>А)	1.5В мин. (В>А) 5В макс. (В>А)
Двоичный 0	5В мин. 15В макс.	2В мин. (А>В) 6В макс. (А>В)	1.5В мин. (А>В) 5В макс. (А>В)
Мин. входное напряжение	+/- 3В	0.2В диф.	0.2В диф.
Выходной ток	500мА	150мА	250мА

Применение дуплексного протокола позволяет осуществлять и прием, и передачу информации одновременно, т.е. оба устройства могут быть и приемником, и передатчиком одновременно. Небалансный сигнал передается по несбалансированной (несимметричной) линии, представляющей собой сигнальную землю и одиночный сигнальный провод, уровень напряжения на котором используется, чтобы передать или получить двоичные 1 или 0. Напротив, балансный сигнал передается по сбалансированной (симметричной) линии, которая представлена сигнальной землей и парой проводов, разница напряжений между которыми используется для передачи/приема бинарной информации (все вместе составляет экранированную витую пару). Не углубляясь в подробности, можно сказать, что сбалансированный сигнал передается быстрее и дальше, чем несбалансированный. Отметим, что контроллеры RS-232, применяемые в персональных компьютерах, имеют выходной ток, равный 10 мА максимум.

Универсальная последовательная шина USB

Универсальная последовательная шина USB (*Universal Serial Bus)* - это кабельная шина, обеспечивающая высокоскоростной обмен информацией между обрабатывающей системой и различными периферийными устройствами. Шина USB разрабатывалась с целью создать реальную возможность пользователям работать в режиме Plug&Play с периферийными

устройствами. Это означает, что должно быть предусмотрено подключение устройства к работающему компьютеру, автоматическое распознавание его немедленно после подключения и последующая установка соответствующих драйверов. Скорость шины должна быть достаточной для подавляющего большинства периферийных устройств.

Дополнительно решается проблема нехватки ресурсов на внутренних шинах IBM PC совместимого компьютера - контроллер USB занимает только одно прерывание независимо от количества подключенных к шине устройств.

Первая версия стандарта открытой архитектуры шины USB [26] была выпущена Intel в 1996 г.

В соответствии со спецификацией шины USB периферийные устройства делятся на два основных типа:

- низкоскоростные устройства (как правило, это переключатели, мыши, джойстики и клавиатуры) передают данные со скоростью 1.5 Мбит/с.;
- высокоскоростные - это устройства, передающие данные со скоростью 12 Мбит/с (например, видеоустройства, диски и сетевые адаптеры).

Возможность использования только двух скоростей обмена данными ограничивает применяемость шины, но существенно уменьшает количество линий интерфейса и упрощает аппаратную реализацию.

Каждое периферийное устройство может запитываться либо от собственного источника питания, либо от шины. USB-устройства, в зависимости от способа получения питания, можно распределить по 5 категориям:

1) концентраторы с питанием от шины;
2) концентраторы с собственным питанием;
3) маломощные периферийные устройства с питанием от шины;
4) мощные периферийные устройства с питанием от шины;
5) периферийные устройства с собственным источником питания.

Напряжение постоянного тока подается от главного концентратора вниз по иерархии. Максимальный ток, потребляемый каждым устройством от шины, составляет 500 мА при напряжении 5 В. Выходные порты концентраторов с собственным питанием могут запитываться от внутреннего источника концентратора с максимальным током нагрузки 500 мА при напряжении 5В. Концентраторы без собственного источника получают питание только от шины, поэтому общий ток нагрузки при подключении к такому концентратору нескольких периферийных устройств может превысить допустимые 500 мА. Для предотвращения отказов источников постоянного тока главный и все промежуточные концентраторы с собственным питанием должны иметь схемы защиты своих источников от перегрузки по току.

Сигналы USB передаются по 4-проводному кабелю, схематично показанному в табл. 5.

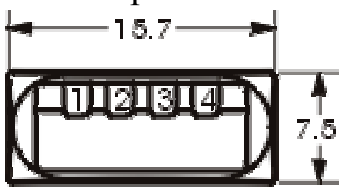
Таблица 5

Номер контакта	Назначение	Цвет провода
1	V BUS	Красный
2	D-	Белый
3	D+	Зеленый
4	GND	Черный
Оплетка	Экран	Оплетка

Здесь GND - цепь "корпуса" для питания периферийных устройств, VBus - +5V также для цепей питания. Шина D+ предназначена для передачи данных по шине, а шина D- для приема данных.

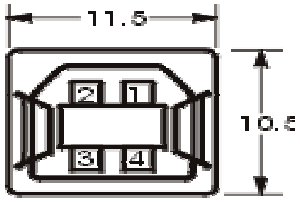
Кабель для поддержки полной скорости шины (full-speed) выполняется как витая пара, защищается экраном и может также использоваться для работы в режиме минимальной скорости (low-speed). Кабель для работы только на минимальной скорости (например, для подключения мыши) может быть и неэкранированным.

Разъемы, используемые для подключения периферийных устройств, показаны на рис. 34.



Series "A"

Предназначены только для подключения к источнику, т.е. к компьютеру или HUB'у



Series "B"

Предназначены только для подключения к периферийному устройству

Рис. 34

Как видно из рисунка, невозможно подключить устройство неправильно, так как разъем серии "A" можно подключить только к активному устройству на USB - HUB'у или компьютеру, а серии "B" - только к собственно периферийному устройству. Конструкция разъемов для USB рассчитана на многократное сочленение/расчленение. Поэтому интерфейс особенно удобен для подключения часто подключаемых (отключаемых) приборов.

Архитектура USB поддерживает построение такой иерархической топологии, в которой "нижележащие" концентраторы и устройства подключаются к "вышележащим" по типу "звезда", при этом вдоль каждого пути может быть не более 5 концентраторов. Концентраторы - это интеллектуальные устройства, которые работают не только как ретрансляторы и маршрутизаторы сигналов, но и обеспечивают некоторые основные функции управления. *Главный концентратор* (в хост-контроллере USB) представляет собой конечный пункт для всех USB-портов системы. Обмен данны-

ми между главным концентратором и USB-устройством происходит в двух направлениях: *вниз* - информация передается от главного концентратора по сигнальным линиям через промежуточные концентраторы, пока не достигнет требуемого периферийного устройства, и *вверх.*

На рис. 35 показан пример правильного соединения периферийных устройств в условную USB-сеть. Так как обмен данными по USB идет только между компьютером и периферийным устройством (между устройствами обмена нет), то устройства с большими объемами приема и/или передачи данных должны подключаться либо к самому компьютеру, либо к ближайшему свободному узлу. В данном случае наивысший трафик у колонок (~1.3 Mb/s), так как драйвер колонок отправляет оцифрованный звук сразу в колонки, где с помощью АЦП он преобразуется в аналоговый сигнал и подается на динамики. Затем идут модем и сканер, подключенные к HUB'у в мониторе, и завершают цепь клавиатура, джойстик и мышь, трафик у которых близок к нулю.

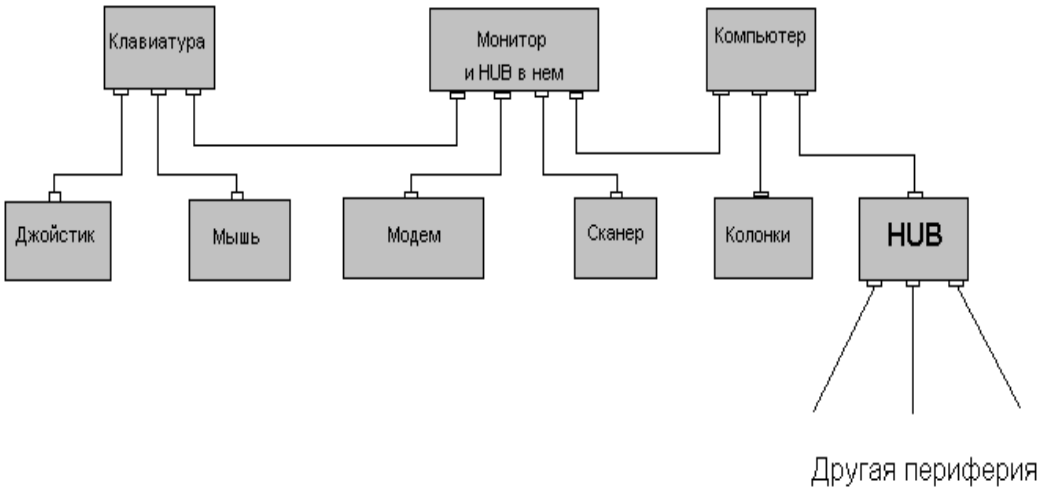


Рис. 35

Системное ПО, поддерживающее USB в хост-системе, должно осуществлять проверку того, что каждая указываемая в операции по передаче данных конечная точка соответствует определенному *физическому* устройству на шине. Для этого при каждом добавлении к шине (или удалении) устройств и/или концентраторов оно должно распознавать физическую иерархию шины и обновлять соответствие между логическими и физическими устройствами. Все операции по передаче данных по шине USB инициируются хостом. Периферийные USB-устройства сами начать обмен данными не могут. Они могут только реагировать на команды хоста. Как только прикладная программа хоста затребует передачу информации, системные программы хоста инициируют ее независимо от направления пересылки данных: от хоста к периферийному устройству или наоборот.

Программное обеспечение АСНИ

Программному обеспечению АСНИ отводится чрезвычайно важная роль, так как именно оно в конечном итоге настраивает на конкретную задачу универсальную систему и определяет, что, как и когда должно делать оборудование. Программное обеспечение является самой гибкой частью АСНИ, что позволяет менять алгоритм обработки и тактику управления в ходе эксперимента.

Главное отличие программного обеспечения АСНИ от традиционных программ научных расчетов состоит в том, что оборудование системы функционирует в конкретном (реальном) времени и тесно связано с объектами эксперимента. Поэтому в составе программного обеспечения АСНИ можно выделить две основные части: вычислительную (алгоритмическую) и управляющую.

Вычислительная часть реализует алгоритм обработки экспериментальных данных и может изменяться в зависимости от результатов хода эксперимента.

В функции управляющей части входят:

- организация совместной работы узлов ЭВМ АСНИ;
- управление передачами данных;
- управление сменой программ (в том числе модулей операционной системы) в процессоре;
- обеспечение диалога с экспериментатором;
- контроль за работой системы.

Вычислительные модули разрабатываются для каждой конкретной задачи пользователем АСНИ. Управляющие модули являются наиболее универсальными (и наиболее трудными в разработке математического обеспечения), разрабатываются и поставляются, как правило, изготовителями оборудования систем.

Вновь создаваемая часть программного обеспечения должна удовлетворять ряду требований, в том числе выполнять заданные условия по выбору реализуемых функций, быстродействию, объему используемой оперативной памяти, надежности, мобильности. Программа должна уметь поддерживать диалог с пользователем на языке, близком к естественному языку предметной области исследований, и представлять информацию в виде, удобном для интерпретации (таблицы, графики, изображения).

Наиболее целесообразно программные средства, входящие в состав АСНИ, условно делить на *системное* и *проблемное* ПО, в зависимости от функций, выполняемых в системе. Однако не всегда можно отнести ту или иную программу к одной из этих категорий. Можно предложить классификацию ПО и по другим признакам:

- по этапу исследования, на котором применяются программы: до эксперимента, при подготовке к эксперименту, в ходе эксперимента, после завершения эксперимента;

- по степени типизации различных АСНИ: общего назначения, проблемно-ориентированные, узкоспециализированные, уникальные;

- по форме представления: набор независимых программ, библиотека программ, пакет программ, диалоговая система, экспертная система;

- по квалификации пользователей, необходимой для применения программ: общедоступные, для специалистов-исследователей, для программирующих исследователей, для администраторов системы.

Системное программное обеспечение

Системное программное обеспечение АСНИ (рис. 36) должно быть достаточным для обеспечения эффективного взаимодействия специалистов, вычислительной системы и исследуемого процесса на всех уровнях. Оно должно обеспечить управление интерфейсом и аппаратурой сопряжения с объектом в реальном масштабе времени, распознавание неисправностей в системе и, по возможности, их ликвидацию, выполнение задач с заданной дисциплиной обслуживания, распределение ресурсов ЭВМ между пользователями, поддержку работы многомашинных и многотерминальных структур АСНИ и т.д. Для реализации этих функций ПО АСНИ должно содержать программы, реализующие режим работы ЭВМ в реальном масштабе времени и в режиме разделения времени.

Рис. 36

Операционная система (ОС ЭВМ) является неотъемлемой составной частью ЭВМ, обеспечивая управление всеми аппаратными компонентами, и позволяет отделить остальные классы программ от непосредственного взаимодействия с аппаратурой [28]. Выбор ОС является важным этапом, оказывающим влияние на эффективность работы всех компонентов программного обеспечения.

В качестве примеров ОС можно привести *MS-DOS* [29]*, CP/M, Unix, OS/2, Novell, Linux,* (для ПК); *OS/360, MVS, VM, OC/EC* (ЭВМ общего на-

значения); *Cray, VPP, Unix, Borough, Эльбрус* (суперЭВМ). При этом, если для ПК, мини-ЭВМ и в ЭВМ общего назначения четко прослеживается тенденция к унификации ОС (платформы *MS-DOS, Windows, Unix),* то суперЭВМ в значительной мере исповедуют *оригинальные* ОС. Интересный исторический обзор по проблематике ОС можно найти в [3].

Как показывает практика, среди имеющегося множества ОС, поставляемых производителями ЭВМ, всегда может быть выбрана та разновидность (версия), которая в наибольшей степени отвечает требованиям конкретной АСНИ, возможным режимам и характеру применения используемых ЭВМ. Для машин, используемых на инструментальном и объектном уровнях, стандартные ОС должны быть дополнены программными средствами, обеспечивающими составление и отладку прикладных программ, связанных с управлением интерфейсными элементами системы (например, блоками КАМАК или МЭК 625.1).

Инструментальное ПО предназначено для создания оригинальных программных средств в любой проблемной области, включая системное ПО. Поэтому в его состав входят *компиляторы* и *интерпретаторы* с языков программирования различного уровня, *библиотеки стандартных программ*, средства редактирования, отладки, тестирования и загрузки, а также *системы программирования* (СП). Под *языком программирования* (ЯП) будем понимать *алфавит, систему записи* и *набор правил,* определяющих синтаксис правильной программы. *Текст* алгоритма задачи, описанный средствами ЯП, называется *исходным* модулем, который может находиться как на бумажном, так и на машинном носителе, пригодном к вводу в ЭВМ. Средством, предназначенным для перевода *исходного* модуля в последовательность команд ЭВМ, является *специальная* программа - *транслятор*. Имеется два основных типа таких программ - *компиляторы* и *интерпретаторы*. *Компилятор транслирует* весь текст *исходного* модуля в машинный код, называемый *объектным* модулем, за один непрерывный процесс. *Объектный* модуль выполняться не может, так как может содержать неразрешенные ссылки на другие модули или программы, а также перемещаемый код. Поэтому перед выполнением программы ее *объектный* модуль должен быть обработан специальной программой - *редактором связей* и *загрузчиком* (определяющим для *загрузочного* модуля абсолютные адреса в оперативной памяти); после этого программа уже может выполняться. Таким образом, общая схема преобразования *исходной* программы в выполняемый *загрузочный* модуль имеет вид:

(ИМ, ОМ, ЗМ - соответственно исходный, объектный, загрузочный модуль).

В общем случае разбиение процесса подготовки *исходного* модуля к выполнению делится по крайней мере на два этапа: *компиляция* и *редактирование,* что позволяет весьма эффективно организовывать решение задач различных классов и типов, давая возможность на стадии редактирования использовать для *основной* программы подпрограммы из богатых *библиотек стандартных программ* (БСП), реализующих различные функции *как общего,* так и *специального* назначения.

В отличие от *компилятора, интерпретатор* выполняет *исходный* модуль программы в режиме "*Оператор за оператором",* по ходу работы превращая каждый оператор ЯВУ в машинные команды. В действительности большинство *интерпретаторов* проводят некоторую *предварительную* работу над всем текстом *исходного* модуля (упрощение представления ключевых слов, присвоение адресов идентификаторам переменных и т.д.). Вместе с тем, оба типа *транслирующих* программ - *компиляторы* и *интерпретаторы* - имеют существенные различия: присутствие в ОП *первых* после получения *объектного* модуля из *исходного* необязательно, тогда как присутствие *вторых* обязательно в период выполнения *исходной* программы.

В принципе, реализация транслятора любого ЯВУ может быть как *компилирующего,* так и *интерпретирующего* типа. Но в силу сложившихся традиций каждый ЯВУ отдает предпочтение одному из указанных типов реализации. Так, *Fortran, Pascal, C* являются, как правило, языками *компилирующего* типа, тогда как *Basic* одинаково использует оба указанных типа. Основным преимуществом ЯВУ *компилирующего* типа является скорость выполнения *загрузочного* модуля скомпилированной *исходной* программы, тогда как *интерпретатор* должен генерировать соответствующую последовательность машинных команд в момент выполнения оператора ЯВУ, что замедляет время ее выполнения. В целом ряде случаев ЯВУ *интерпретирующего* типа более предпочтительны; они хорошо отвечают программам *диалогового* типа, отладка программ производится без выхода из *интерпретатора,* что существенно упрощает эту процедуру и сокращает временные издержки на нее, ибо каждый раз не требуется перетрансляции скорректированного *исходного* модуля.

Для сокращения затрат на разработку программного обеспечения применяют *инструментальные системы программирования*, которые включают стандартные готовые модули управляющих программ, пакетов и программные средства генерации пакетов. Введенные в состав инструментальных систем сервисные средства позволяют значительно уменьшить объем трудозатрат на программирование и сосредоточивают внимание разработчиков пакетов на формировании логических и информационных связей в модели предметной области. При этом поддерживается диалоговый процесс с разработчиком на квази-естественном языке, проводится синтаксический и семантический анализ вводимой информации, выдаются различные справочные сведения.

В отдельную группу можно выделить программы, управляющие *интерфейсом* между ЭВМ и объектом исследования. С их помощью осуществляется заданный план измерений и управления объектом, преобразование, а иногда и первичная обработка информации. Для получивших широкое распространение международных стандартов интерфейсов КАМАК и МЭК 625.1 применяются типовые комплексы управляющих программ-мониторов интерфейса.

Для организации хранения, накопления и поиска информации в процессе исследования в состав ПО включают *систему управления базами данных (СУБД)*, которая представляет собой программную систему, реализующую различные операции с хранимыми данными и поддерживающую определенную физическую организацию хранения данных на внешних носителях.

Контроль работоспособности компонентов технического и программного обеспечения осуществляется с помощью *диагностических программ,* которые позволяют удостовериться в работоспособности системы. Эти программы выполняются перед началом эксперимента или его крупных этапов, а также при возникновении сомнений в нормальном функционировании системы.

Для контроля качества информации, поступающей с измерительных элементов АСНИ, используются программы, составляющие *метрологическую подсистему*. С их помощью обеспечивается точность и достоверность полученных данных, периодически осуществляется метрологический контроль характеристик всей системы.

Подсистема *диалога* предназначена для поддержки процесса общения исследователя с АСНИ в ходе решения задачи. Эти программы обеспечивают определенный уровень комфорта при работе исследователя, делают систему более привлекательной для экспериментаторов, не имеющих большого опыта общения с вычислительной техникой.

Для обеспечения удобства анализа полученных данных их отображают в виде графиков, гистограмм, поверхностей в различных системах координат на экране дисплея или графопостроителе. Для реализации этих функций в ПО АСНИ должны быть включены программы, составляющие *подсистему графики*.

Процесс начального освоения АСНИ поддерживается программами *подсистемы обучения* (АОС), которая включает типовые автоматизированные курсы для освоения стандартных технических и программных средств системы, а также курсы для обучения проведению конкретных исследований с использованием АСНИ в целом.

Проблемное программное обеспечение

К проблемному ПО относятся отдельные программы и комплексы программ, реализующие те функции, для выполнения которых и создается АСНИ: сбор и обработка информации с объекта исследования, управление экспериментом на объекте, моделирование объекта и его частей, представление результатов в наглядной форме и др. В процессе исследования данные, полученные при выполнении одной программы, могут потребоваться для выполнения другой. Поэтому проблемные программы должны иметь информационные связи друг с другом, которые удобно реализовать на основе единой СУБД.

В проблемном ПО можно выделить шесть групп программ (рис. 37) с функциональным назначением для сбора информации, ее первичной и основной обработки, планирования эксперимента, управления экспериментом и моделирования процесса исследования.

Рис. 37

Управление экспериментом, сбор информации и ее первичная обработка обычно выполняются *в режиме реального времени*, в котором на время выполнения операций накладываются жесткие ограничения. Разработка программ, реализующих эти функции, должна выполняться с использованием машинно-ориентированных языков. Эти программы в значительной степени зависят от типа ЭВМ и технических средств АСНИ.

При сборе информации в ходе эксперимента программы должны осуществлять опрос датчиков с требуемой периодичностью, которая может быть разной для разных групп датчиков. Программная реализация этой функции зависит от применяемого в АСНИ интерфейса, от числа источников информации и от требований к быстродействию системы. Информация, полученная за определенные промежутки времени, составляет исходный кадр измерений. Так как за этот промежуток некоторые датчики могут опрашиваться несколько раз, при формировании исходного кадра производится усреднение таких измерений.

По результатам измерений физических величин в реальном масштабе времени часто требуется вычислить ряд производных величин, не поддающихся непосредственному измерению. Оперативно получаемые значения расчетных величин позволяют исследователю контролировать ход эксперимента, в некоторых случаях принимать решения, влияющие на получаемые результаты. Расчетные величины вместе с непосредственно измеренными данными составляют полные кадры измерений, запоминаемые на устройствах длительного хранения информации.

При *первичной обработке* информации выполняются операции подготовки данных к дальнейшему, более полному анализу. В процессе первичной обработки повышается качество данных за счет выявления и локализации аномальных измерений, выделение полезных составляющих сигнала на фоне шума, преобразование данных с целью исключения систематических ошибок, учет нелинейных характеристик измерительных трактов. Производится расчет основных статистических характеристик с целью анализа свойств отдельных источников информации.

Типовой набор программ, реализующих первичную статистическую обработку, должен обеспечивать расчет оценок первых четырех моментов распределения случайной величины и их доверительных интервалов, коэффициентов корреляции. Для временных рядов рассчитываются оценки корреляционных функций и их доверительных интервалов, оценки спектральной плотности. Алгоритмы фильтрации сигналов должны обеспечивать подавление помех в высокочастотной и низкочастотной областях спектра, в заданной полосе частот, осуществлять оптимальную и согласованную фильтрацию.

В ходе *основной обработки* выполняются более сложные операции с данными, приводящие к достижению целей исследования. При этом проводится построение модели объекта исследования, ее анализ и интерпрета-

ция, проверяются гипотезы о свойствах объекта, принимаются решения о целях дальнейшего исследования.

Для получения математических моделей статических объектов в программном обеспечении АСНИ необходимо иметь программы аппроксимации зависимостей полиномами, ортогональными функциями, сплайнами, программы непараметрической аппроксимации зависимостей.

Большое значение имеют программы *редукции размерности* задач, позволяющие отобрать наиболее информативные факторы и отбросить незначимые компоненты модели.

При проведении исследований часто требуется создать на ЭВМ модель объекта, имитирующую его поведение. Для этого в ПО необходимо включить программы, которые моделируют типовые передаточные функции статических и динамических объектов, генерируют типовые сигналы, имеющие место в реальном объекте (например, гармонические, импульсные, случайные процессы с разными свойствами).

Высокая эффективность эксперимента обеспечивается при его оптимальном планировании. Для этого в ПО включают программы генерации ряда типовых планов, численного синтеза априорных планов по разным критериям оптимальности, последовательного планирования. Сопоставление различных вариантов планов осуществляется с помощью программ анализа характеристик планов.

Для реализации выбранного плана эксперимента или заданного закона управления объектом необходимо иметь в распоряжении комплекс программ, реализующих локальное регулирование заданного типа, контроль ограничений по управляющим воздействиям, выработку воздействий при аварийных ситуациях. Управление объектом в реальном масштабе времени требует использования в программах специальных приемов для согласования процессов управления и измерения с процессами, протекающими в изучаемом объекте.

Нет необходимости, чтобы в ПО какой-либо АСНИ входили и поддерживались постоянно в работоспособном состоянии все перечисленные программы. Наиболее целесообразным является вариант генерируемого ПО, при котором на каждом этапе исследования сопровождается лишь строго ограниченное подмножество программ. При возникновении необходимости к нему присоединяются новые типовые или вновь разработанные специализированные программы. При этом из ПО исключаются компоненты, необходимость в которых отпала на данном этапе исследований.

Языки программирования в АСНИ

При разработке прикладного ПО для АСНИ необходимо помнить, что оно должно решать следующие задачи в режиме реального времени: управление экспериментальной установкой, сбор и первичную обработку информации с объекта исследования. Таким образом, программы должны иметь возможность доступа к регистрам ввода/вывода внешних устройств, к регистрам прерываний. Кроме этого, необходимо обеспечить быструю реакцию ПО на происходящие в системе изменения. ПО АСНИ должно обеспечить хранение собранных данных. Данные из различных предметных областей по структуре могут сильно отличаться друг от друга. Поэтому желательно иметь гибкие языковые средства, дающие возможность работать с данными сложной структуры. Наконец, ПО АСНИ должно обеспечивать математико-статистическую обработку информации.

В связи с тем, что ни один язык программирования не может удовлетворить всем предъявляемым требованиям, возникает интерес к языкам, обеспечивающим достаточную надежность и эффективность программ и пригодным для программирования как математических расчетов, так и на аппаратном уровне и позволяющим работать со сложными структурами данных. К таким языкам наиболее близки PASCAL, C, BASIC.

Языки PASCAL и C чаще всего используются программистами для разработки системных и прикладных программ. Оба эти языка позволяют работать с данными сложной структуры; оба имеют развитые средства для выделения отдельных частей программ в процедуры. Трансляторы с этих языков работают в режиме компиляции, что позволяет создавать эффективные машинные программы. Важным средством для построения больших программных систем является модульность, т.е. возможность независимой разработки отдельных частей программ и последующего их связывания в единую систему. Все эти особенности способствовали тому, что именно на PASCAL и C разрабатывается большинство крупных программных систем.

Между указанными языками, несмотря на общее сходство, имеются существенные различия.

PASCAL является классическим языком программирования. Программы, написанные на нем, понятны любому программисту-профессионалу, они транслируются в эффективные машинные коды.

Язык C, в отличие от PASCAL, с момента своего появления (1972 г.) был ориентирован на разработку системных программ. В этом языке имеются более гибкие средства для эффективного использования особенностей аппаратуры, чем в PASCAL. Благодаря этому порождаемые машинные программы, как правило, более компактны и работают быстрее, чем программы, полученные PASCAL-трансляторами. С другой стороны, синтаксис языка C менее нагляден, чем у PASCAL; возможностей для внесе-

ния ошибок больше; чтение текстов программ требует определенного навыка. Язык С применяется главным образом для создания программ, в которых скорость работы и объем памяти являются критическими параметрами. Это увеличивает его привлекательность при разработке и эксплуатации АСНИ.

Язык BASIC является широко распространенным благодаря своей простоте в освоении и использовании. В BASIC встроены удобные функции для работы с экраном дисплея, клавиатурой, внешними накопителями, коммуникационными каналами. BASIC обладает хорошими возможностями графического представления информации, аппаратными функциями. Это позволяет относиться к языку BASIC как к "продолжению" аппаратуры ЭВМ.

Таким образом, для разработки больших программных систем в АСНИ в операционной системе типа UNIX можно рекомендовать использовать язык С. При разработке отдельных программ, особенно если они обеспечивают работу конкретной аппаратуры и не требуется переноса на другие машины, можно использовать язык PASCAL.

Пакеты программ автоматизации эксперимента и обработки данных

Наряду с традиционными средствами программирования на языках высокого уровня, таких, как PASCAL и С, активно развиваются специализированные программные оболочки, ориентированные на создание программного обеспечения автоматизированных систем.

Интегрированная система программирования MathCAD

В настоящее время для моделирования и численного анализа различных устройств широко используется система MathCAD [30, 31].

MathCAD является интегрированной системой программирования, ориентированной на проведение математических, инженерно-технических, статистических и экономических расчетов. MathCAD содержит текстовый редактор, вычислитель и графический процессор. Текстовый редактор служит для ввода и редактирования текстов. Вычислитель системы MathCAD [32] работает в режиме интерпретатора и обеспечивает вычисления по сложным математическим формулам, имеет обширный набор встроенных математических функций, обеспечивает вычисления рядов, сумм и произведений, определенных интегралов и производных. Графический процессор служит для создания графиков. Возможно построение двумерных графиков в декартовой и полярной системах координат, контурных графиков, 3D-поверхностей, 3D-диаграмм, полей векторов. Запись команд в системе MathCAD осуществляется на языке, очень близком к

стандартному языку математических расчетов, что резко упрощает постановку и решение задач. Таким образом, главные аспекты решения математических задач смещаются с их программирования на алгоритмическое и математическое описание.

Пример программы, написанной в среде MathCAD, представлен на рис. 38.

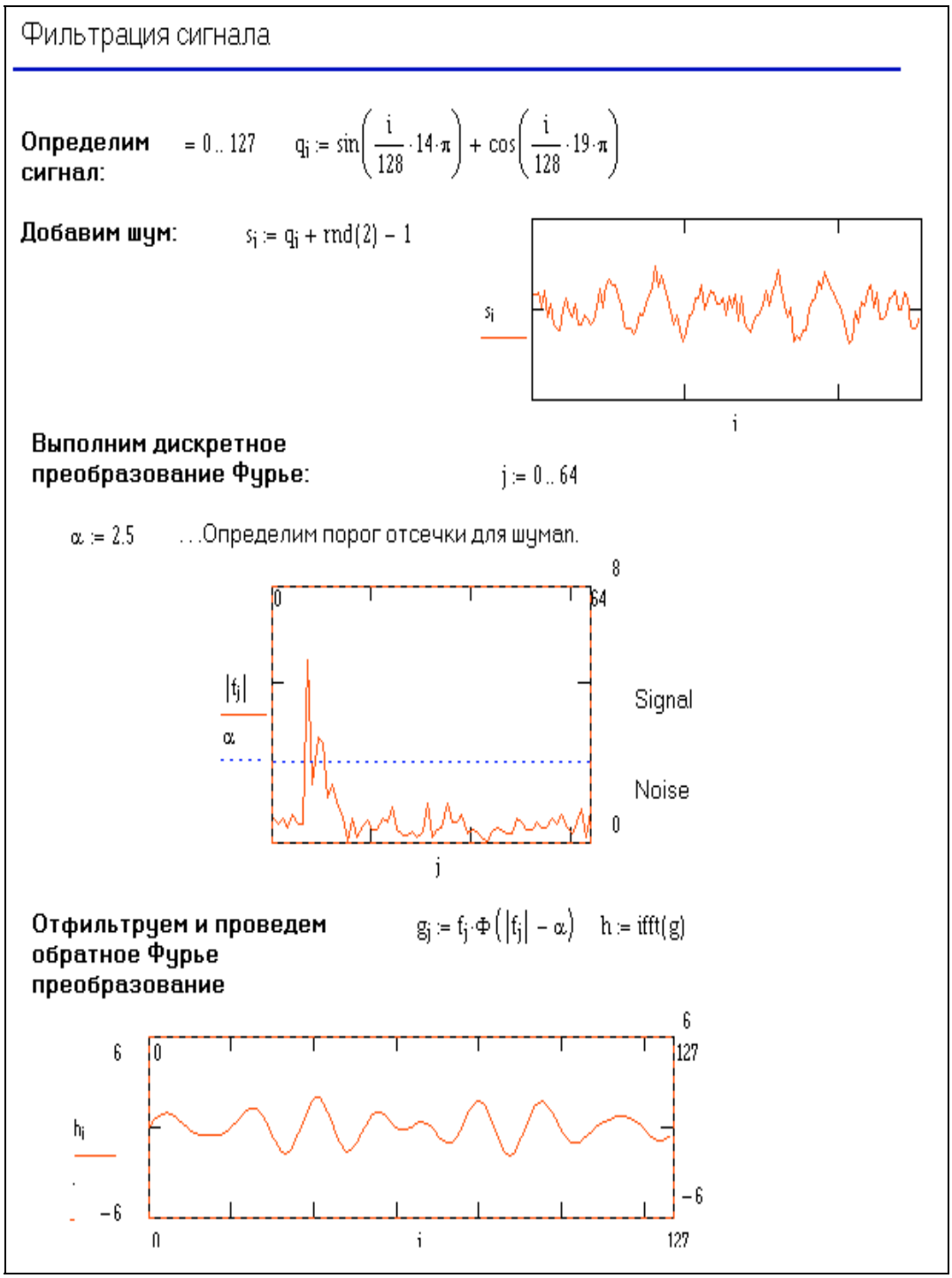


Рис. 38

MATLAB

Система MATLAB *(MATrix LABoratory)* давно и успешно разрабатывается фирмой MathWorks.

Нынешний MATLAB - это высокоэффективный язык инженерных и научных вычислений. Он поддерживает математические вычисления, визуализацию научной графики и программирование с использованием легко осваиваемого операционного окружения, когда задачи и их решения могут быть представлены в нотации, близкой к математической. Наиболее известные области применения системы MATLAB:

- математика и вычисления;
- вычислительный эксперимент, имитационное моделирование, макетирование;
- анализ данных, исследование и визуализация результатов;
- научная и инженерная графика;
- разработка приложений, включая графический интерфейс пользователя.

MATLAB - это интерактивная система, основным объектом которой является массив, для которого не требуется указывать размерность явно. Это позволяет решать многие вычислительные задачи, связанные с векторно-матричными формулировками, существенно сокращая время, которое понадобилось бы для программирования на скалярных языках типа C или Fortran.

Система MATLAB - это одновременно и операционная среда, и язык программирования. Одна из наиболее сильных сторон системы состоит в том, что на языке MATLAB могут быть написаны программы для многократного использования. Пользователь может сам написать специализированные функции и программы, которые оформляются в виде M-файлов. По мере увеличения количества созданных программ они группируются в специальные папки. Это приводит к концепции пакетов прикладных программ (ППП), которые представляют собой коллекции M-файлов для решения определенной задачи или проблемы.

Например, система The Student Edition of MATLAB 5 включает ядро системы MATLAB 5, а также пакеты прикладных программ по символьной математике Symbolic Mathematics Toolbox, по системам управления Control Systems Toolbox, по обработке сигналов Signal Processing Toolbox. Все ППП функционируют в операционной среде системы MATLAB.

Операционная среда системы MATLAB - это множество интерфейсов, которые поддерживают связь этой системы с внешним миром. Это диалог с пользователем через командную строку или графический интерфейс, просмотр рабочей области и путей доступа, редактор и отладчик M-файлов, работа с файлами и оболочкой DOS, экспорт и импорт данных, динамическое взаимодействие с внешними системами.

Реализуются эти интерфейсы через командное окно, инструментальную панель, системы просмотра рабочей области и путей доступа, редактор/отладчик М-файлов, специальные меню и т.п.

Основная программная единица - М-файлы. Самым простым типом М-файла являются сценарии: у них нет входных и выходных аргументов. Они полезны для выполнения последовательности MATLAB-команд вычисления, которые должны были бы многократно вводиться из командной строки.

Одним из компонентов MATLAB является Simulink – интерактивная среда для моделирования и анализа широкого класса динамических систем. Simulink предоставляет пользователю графический интерфейс для конструирования моделей из стандартных блоков при помощи технологии “drag-and-drop”. В Simulink входит большая библиотека блоков, позволяющая легко создавать модели. Для этого достаточно перенести компоненты из библиотеки в новую модель и соединить их с помощью мыши. Библиотека блоков может быть дополнена специализированными библиотеками. Например, библиотека DSP Blockset используется для разработки алгоритмов цифровой обработки сигналов, библиотека Fixed-Point Blockset расширяет возможности Simulink для моделирования систем управления и цифровой фильтрации.

Более подробно система MATLAB описана в [31, 32, 33]

LabVIEW

Как правило, программный пакет покрывает только один аспект поставленной задачи, но не решает все проблемы: сбор данных, их анализ, представление и управление. LabVIEW предоставляет вам все необходимые средства, объединенные одной методологией. LabVIEW дает возможность пользователям более быстро программировать аппаратуру, системы получения и накопления данных.

LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*) - комплексная система для автоматизированного проектирования лабораторного эксперимента, основанная на графическом языке программирования G. LabVIEW полностью интегрирован для работы с оборудованием по протоколам GPIB, VXI, PXI, RS-232, RS-485, получения и накопления данных от встраиваемых контроллеров. LabVIEW предлагает [34] более 600 драйверов для приборов от более чем 50 мировых производителей, таким образом исключается необходимость низкоуровнего программирования приборов. LabVIEW также имеет встроенные библиотеки для использования стандартного программного обеспечения по сетевым протоколам TCP/IP и ActiveX.

Программа, разработанная в системе LabVIEW, называется Виртуальным Инструментом (*VI* или ВИ), поскольку она моделирует некоторый реальный или воображаемый прибор.

LabVIEW имеет большой набор готовых библиотек программ ВИ [35] для математических, логических, текстовых преобразований данных, операций ввода/вывода, управления таймером компьютера, интерфейсными устройствами и т.д. Вы можете воспользоваться цифровой обработкой сигналов (DSP), цифровой фильтрацией, статистикой и численным анализом.

Библиотеки управления интерфейсными устройствами [36] предназначены для управления обменом данными в автоматизированных системах.

Библиотека управления многофункциональными измерительно-управляющими платами DAQ. Фирмой National Instruments разработан ряд встраиваемых в компьютер и внешних многофункциональных устройств, таких, как многоканальные АЦП, ЦАП, счетчики, и др. Весь этот спектр оборудования управляется единой библиотекой NI-DAQ. LabVIEW имеет встроенную систему взаимодействия с NI-DAQ, следовательно, с помощью LabVIEW можно быстро и просто разрабатывать программы автоматизированных систем, построенных на базе встраиваемых устройств. Библиотека NI-DAQ имеет функции управления аналоговым вводом/выводом, цифровым вводом/выводом, частотным вводом/выводом, калибровкой, преобразователями сигналов и др.

Библиотека управления приборами по последовательным портам RS-232 и RS-485. В состав библиотек LabVIEW входит библиотека управления устройствами по последовательным портам RS-232, RS-485. Она позволяет настраивать коммуникационные порты на нужный режим и проводить обмен данными между компьютером и периферийными устройствами.

Библиотека управления приборами по магистрали GPIB. Библиотека GPIB в LabVIEW содержит в себе полный набор функций, необходимых для работы с приборным интерфейсом. Весь спектр GPIB устройств и приборов, имеющих приборный интерфейс, управляется единой библиотекой NI-GPIB.

В построении всякого ВИ фигурируют три компонента: *передняя панель, блок-диаграмма и иконка/коннектор.*

Передняя панель ВИ (рис. 39) представляет собой аналог панели некоторого реального или воображаемого прибора с возможностью выбора разнообразных органов управления программой (интерактивный интерфейс пользователя). Регуляторы (control) служат для ввода данных в программу, а индикаторы (indicator) - для отображения результатов работы. Образцы регуляторов и индикаторов (тумблеры, кнопки, лампочки, графики, числовые и стрелочные индикаторы и т.п.) можно найти в специальной палитре Controls Palette (рис. 41*б*).

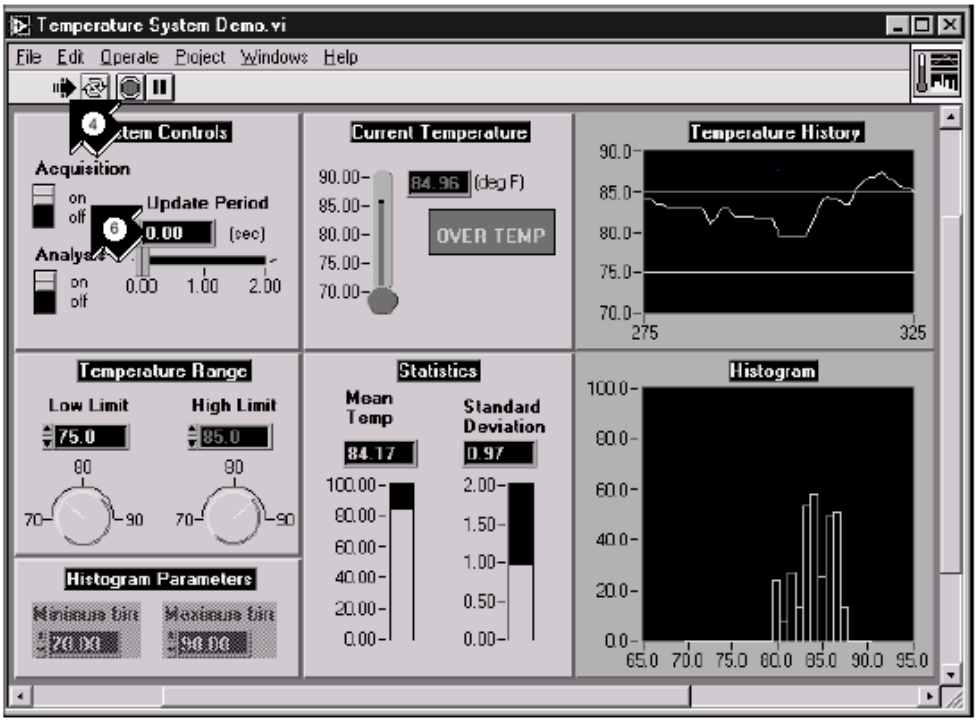


Рис. 39

Блок-диаграмма ВИ (см. рис. 40) соответствует передней панели ВИ и создается одновременно, но в другом окне. Блок-диаграмма собственно и является программой в LabVIEW. Компоненты блок-диаграммы соответствуют элементам традиционного языка программирования, например циклам, операторам условного перехода, арифметическим функциям и т.п., но они представленны в виде графических объектов. Все многообразие таких объектов можно найти в специальной палитре Functions Palette (рис. 41*в*).

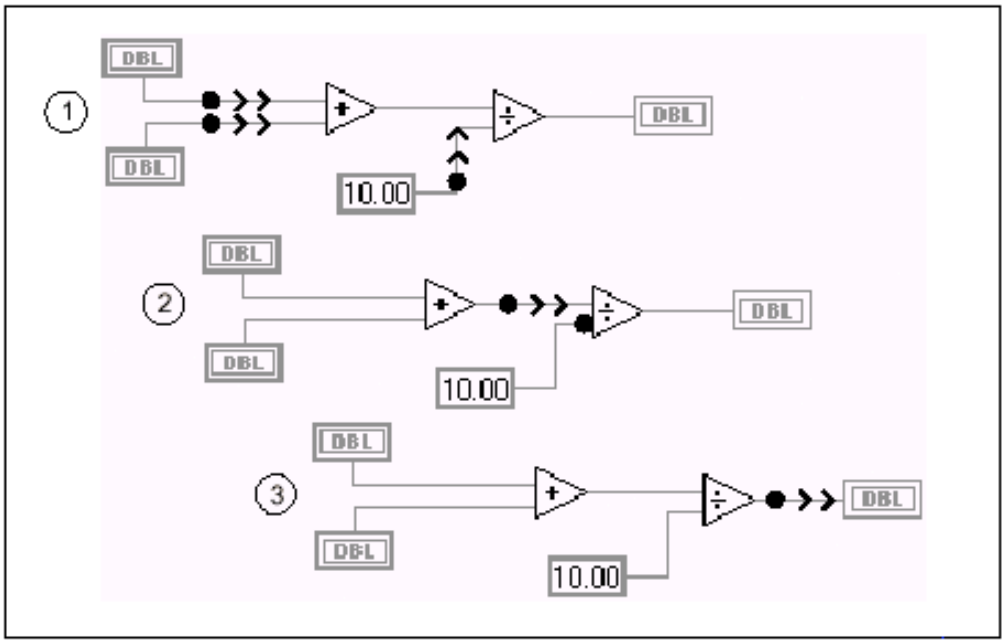


Рис. 40

Так как порядок выполнения программы в LabVIEW устанавливается течением данных между блоками, а не последовательностью строк текста, вы можете создавать диаграммы, которые имеют несколько параллельных потоков прохождения данных и несколько одновременно выполняемых операций.

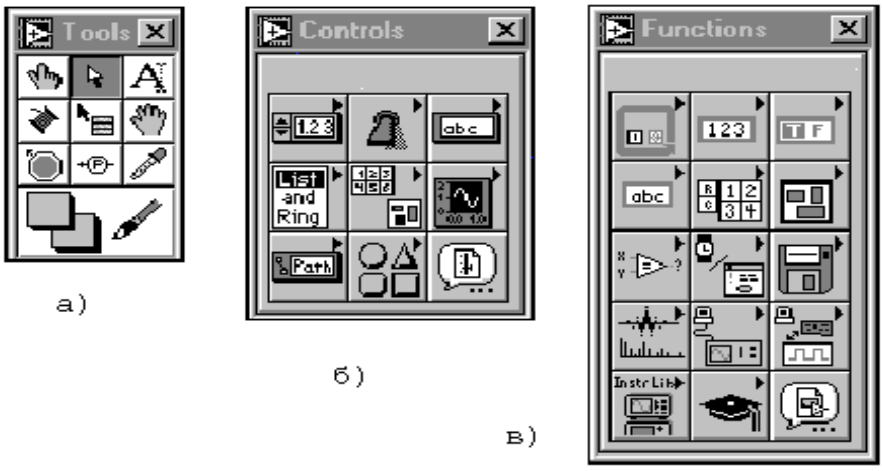


Рис. 41

Для всех операций на передних панелях и на блок-диаграммах ВИ применяются инструменты из специальной палитры Tools Palette (рис. 41*а*). С помощью этих инструментов осуществляется перетаскивание на панели разных объектов, их выделение, изменение параметров, размеров и цвета, соединения терминалов, удаление ненужных объектов и вообще редактирование и отладка всей программы.

Таким образом, LabVIEW позволяет создавать программы для работы с реальными приложениями в широком диапазоне - от простейших до достаточно сложных систем, работающих в реальном масштабе времени.

Литература

1. ГОСТ 16263-70. Метрология. Термины и определения. М., 1978.

2. Кузьмичев Д.А., Радкевич И.А., Смирнов А.Д. Автоматизация экспериментальных исследований: Учеб. пособие. М., 1983.

3. Аладьев В.З., Хунт Ю.Я., Шишаков М.Л. Основы информатики. Учеб. пособие. М., 1999.

4. Справочник по аналоговой вычислительной технике. Киев, 1975.

5. Анисимов Б.В., Голубкин В.Н., Петраков С.В. Аналоговые и гибридные ЭВМ. М., 1986.

6. Смирнов Ю., Воробьев Г. Специализированные ЭВМ. М., 1989.

7. Аладьев В.З. Архитектура и программное обеспечение СМ ЭВМ. Таллин, 1983.

8. Прохоров Н., Песелев К. Малые ЭВМ. М., 1989.

9. Аладьев В.З. Введение в архитектуру моделей ЕС ЭВМ. Таллин, 1976

10. Бабаян Б.А. и др. Многопроцессорные ЭВМ и методы их проектирования. М., 1990.

11. Каган Б.М. Электронные вычислительные машины и системы. М., 1991

12. Дорфман В.Ф., Иванов Л.В. ЭВМ и ее элементы: Развитие и оптимизация. М., 1986.

13. Компьютеры: Справочное руководство. М., 1986.

14. Технические средства АСУ: Справочник: В 2 т. Т. 2 / Под ред. Г.Б. Козлинга. Л., 1986.

15. Системы автоматизированного проектирования. Кн. 2: Технические средства и операционные системы / Д.М. Жук., В.А. Мартынюк, П.А. Сомов; Под ред. И.П. Норенкова. Минск, 1988.

16. Вейцман К. Распределенные системы мини- и микроЭВМ. М., 1982.

17. Новиков Ю.В., Карпенко Д.Г. Аппаратура локальных сетей: функции, выбор, разработка / Под общей редакцией Ю.В. Новикова. М., 1998.

18. Виноградов В.И. Информационно-вычислительные системы: распределенные модульные системы автоматизации. М., 1986.

19. Науман Г., Майлинг В., Щербина А. Стандартные интерфейсы для измерительной техники: Пер. с нем. / Под ред. А.С. Бондаревского. М., 1982.

20. Хазанов Б.И. Интерфейсы измерительных систем. М., 1979.

21. Интерфейс для программируемых приборов в системах автоматизации эксперимента / Н.И. Гореликов, А.Н. Домарацкий, С.Н. Домарацкий и др. М., 1981.

22. ГОСТ 26.003-80. Система интерфейса для измерительных устройств с байт–последовательным, бит-параллельным обменом информацией. М., 1985.

23. Мирский Г.Я. Микропроцессоры в измерительных приборах. М., 1984.

24. Автоматизированные системы научных исследований. Техническое обеспечение Виноградова Н.А. , Есюткин А.А., Филаретов Г.Ф. / Под ред. В.А. Кабанова. М., 1990.

25. ГОСТ 26.201-80. Система КАМАК. Крейт и сменные блоки. М., 1985.

26. Universal Serial Bus Specification. 1996. V. 1.0; Universal Host Controller Interface Design Guide, rev. 1.1, March 1996 (http://www. teleport. com/~usb).

27. Брябрин В.М. Программное обеспечение персональных ЭВМ. М., 1989.

28. Дьяконов В.П., Абраменкова И.В. MathCAD 7.0 в математике, физике и в Internet. М., 2000.

29. Программирование в MathCAD: Метод. указания / Сост. Н.И. Фомичев; Ярославль, 2001.

30. MathCAD 2000. Reference Manual, MathSoft, Inc., 2000 (http://www. mathsoft. com)

31. Потемкин Г.В. Система MATLAB: Справочное пособие. М., 1998.

32. Потемкин Г.В. Matlab 5 для студентов. М., 1998.

33. Коткин Г.Л., Черкасский В.С. Компьютерное моделирование физических процессов с использованием MATLAB: Учеб. пособие. Новосибирск, 2001.

34. LabVIEW Application Builder. National Instruments Corp. 1999.

35. Использование виртуальных инструментов LabVIEW / Под ред. К.С. Демирчана и В.Г. Миронова. М., 1999.

36. Ковалев С.И., Свиридов Е.В., Устинов А.В. Автоматизация лабораторного эксперимента / Под ред. Г.Ф. Филаретова. М., 1999.

ОГЛАВЛЕНИЕ