МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
ЮЖНО-КАЗАХСТАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. М.О.АУЕЗОВА
КАФЕДРА АВТОМАТИЗАЦИИ, КОМПЬЮТЕРИЗАЦИИ И ИНФОРМАТИЗАЦИИ

Инков А.М.

АВТОМАТИЗАЦИЯ ТИПОВЫХ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

Конспект лекции по теме:
«**АСУТП-часть 1**».

для студентов специальностей
36.01.40 и 36.02.40.

ШЫМКЕНТ 2005

УДК 66-52:66(075)
ББК 6П7.1

Инков А.М. Автоматизация процессов и производств/ Конспект лекции по теме «**АСУТП-часть 1»**. Для студентов специальностей 36.01.40 и 36.02.40.
Шымкент, Южно-Казахстанский Государственный Университет (ЮКГУ). 2005 г. – 62 стр.

В конспекте приведен материал по теме «**АСУТП- часть 1»** », приведены примеры использования

Рецензенты: д.т.н., профессор, проректор по развитию и УМО ЮКГУ Ескендиров Ш.З.

к.т.н., проректор по научной работе Международного гуманитарно-технического Университета к.т.н., Апсеметов А.Т.

Рассмотрены и рекомендованы к печати научно - методическим центром ЮКГУ, протокол № __ от __ __ 200__ г.

© Южно-Казахстанский Государственный Университет (ЮКГУ), 2005 г.

© Инков А.М., 2005 г.

СОДЕРЖАНИЕ

Стр .

1 Ведение 5
1.1 Определение АСУТП 5
1.1 Задачи АСУТП: 5
1.2 Основные характеристики эффективности АСУТП 5
1.3 Особенности АСУТП: 5
1.4 Определение управления 6
1.5 Функции управления 6
2 Архитектура АСУТП 8
2.1 Определение архитектуры 8
2.2 Составляющие архитектуры 8
2.3 Архитектура АСУТП с точки зрения обработки входных данных во времени 8
2.4 Организация вычислений в АСУТП 8
2.5 Архитектура АСУТП с точки зрения тестируемости 9
2.6 Эволюция архитектур АСУТП 10
2.7 Примеры архитектур 10
2.8 Влияние особенностей архитектуры АСУТП 11
3. Типы вычислительных элементов АСУТП 12
3.1 Препроцессоры 12
3.2 Векторные и матричные процессоры 12
3.3 Цифровые сигнальные процессоры - DSP 13
3.4 Контроллеры 13
3.5 Процессоры данных 14
3.6 Выводы: 15
3.7 Реализация различных типов вычислительных элементов 15
4.Коммуникации в АСУТП 16
4.1 Шины управления 16
4.2 Шины данных 18
4.3 Шины тестирования и настройки 19
4.4 Шины ввода-вывода 20
4.5 Сетевые соединения 21
4.6 Шины персональных компьютеров 22
5. Топология сетей в АСУТП 23
5.1 Линейная топология 23
5.2 Топология кольца 23
5.3 Коммутируемые топологии 24
5.4 Топология решетки 24
6. Типы архитектур АСУТП 25
6.1 Архитектура прямого соответствия 25
6.2 Архитектура с разделяемой шиной 25
6.3 Кольцевая архитектура 26
6.4 Коммутируемая архитектура 26
6.5 Решетчатая архитектура 27
6.6 Комбинированная архитектура 27

7 Архитектура программного обеспечения АСУТП 28
8 Уровни стандартизации программного обеспечения 29
9 Организация процесса разработки программного обеспечения 31
9.1 Структурное программирование 31
9.2 Объектно-ориентированное программирование 31
10 Общий алгоритм функционирования 33
11 Процесс выбора архитектуры автоматизированной системы управления и контроля 35
12 Протоколы доступа к физическим каналам связи 37
12.1 Показатели эффективности протокола 37
12.2 Протоколы опроса. 38
12.3 Выбор протокола доступа к каналу связи 41
13 Организация ввода/вывода в цифровых АСУТП 42
13.1 Разновидности ввода/вывода 42
13.3 Буферизация ввода-вывода 43
13.4 Опережающий ввод-вывод (составные буфера) 44
13.5 Ввод-вывод по прерыванию 44
13.6 Объединение буферов в пул ввода-вывода 46
14 Общие понятия и задачи системного программного обеспечения 47
15 Программные методы контроля 49
16 Свойства формальных моделей в АСУТП 51
17 Рекомендуемая литература 61

1 Ведение

1.1 Определение АСУТП

Система, осуществляющая сбор данных о каком-либо физическом или ином объекте, их обработку по заданному или синтезируемому алгоритму с целью выделения информативных параметров об объекте, и принимает решение о формировании сигналов управления, может считаться «системой управления».

Если сигналы управления не формируются, а вместо этого осуществляется какое-либо представление полученной информации, то такая система является «Системой контроля».

В случае, если какая-то часть операций выполняется без участия человека, то эти системы становятся автоматизированными, а конце концов автоматическими.

В общем случае в состав АСУТП входит несколько составляющих: датчики, преобразующие физические поля объекта контроля и управления в электро-магнитные сигналы, устройство обработки, осуществляющее выделение информации с данных датчиков, решающее устройство, формирующее сигналы контроля и управления, исполнительное устройство и устройство индикации и линии связи, объединяющие всё это в единую систему.

1.1 Задачи АСУТП:

1) Сбор информации об объекте.
2) Обработка информации об объекте по алгоритму, определяющемуся состоянием объекта и АСУТП.
3) Выработка сигналов контроля и управления.
4) Составные части АСУТП:
5) Источники данных об объекте - датчики.
6) Линии передачи информации.
7) Средства обработки информации, принятия решений и формирования сигналов контроля, индикации, управления.
8) Исполнительные устройства, устройства индикации и контроля.

Основным принципом построения АСУТП является принцип иерархической организации: все системы и подсистемы находятся в определенной подчиненности в соответствии с предназначением системы и подсистемы.

1.2 Основные характеристики эффективности АСУТП

1) Надежность – достигается резервированием аппаратуры, самотестируемостью, способностью динамической реконфигурируемости, дублированием.
2) Качество управления – повышение достоверности информативных параметров объекта за счет алгоритмов обработки данных, алгоритмы управления.
3) Помехозащищенность – по данным, по системам передачи данных, защита от сбоев.
4) Устойчивость – за счет алгоритмов управления.
5) Интерфейс – внутренние и внешние аппаратно-программные интерфейсы, пользовательский интерфейс.

1.3 Особенности АСУТП:

1) Большое количество элементов, ее составляющих.

2) Наличие общей задачи и единой цели функционирования системы.
3) Возможность иерархического деления системы на функциональные блоки.
4) Иерархическая структура связей подсистемы.
5) Иерархия критериев качества (метрик) функционирования системы.
6) Сложность поведения системы, определяющаяся случайным характером внешних воздействий и большим количеством составляющих.
7) Централизация и автоматизация управления.
8) Устойчивость к внешним и внутренним помехам и сбоям, самоорганизация, восстановление при отказе.

Регулярные методы синтеза и анализа АСУТП не позволяют плодотворно описывать и синтезировать комплексные АСУТП. В настоящее время основным инструментом разработки и верификации подобных разнородных по элементной базе и алгоритмам функционирования систем являются системы прототипирования.

1.4 Определение управления

Управление – деятельность по осуществлению на базе регулирования процесса переработки информации о состоянии объекта в сигналы, обеспечивающие либо сохранение состояния объекта, либо приведение объекта в требуемое программой состояние.

Процесс управления изучен, если возможно воспроизвести все преобразования, происходящие в системе, под воздействием управляющих сигналов, а также воспроизвести последовательность выработки этих сигналов.

Изученный процесс управления называется алгоритмом управления. Автоматизация сводится к оптимизации процесса управления.

Процессы управления изучаются, в основном, двумя науками:

Информатика – наука о способах представления и структурирования данных модели. Использует методы дискретной математики и логики как аппарат анализа и синтеза.

Кибернетика – наука об управлении на основе численных математических моделей и методов. Использует математический анализ непрерывных характеристик объектов.

1.5 Функции управления

1) Прогнозирование.
2) Планирование.
3) Учет.
4) Контроль.
5) Анализ.

Пример АСУТП. Простейшей АСУТП может считаться релейный регулятор температуры – датчик температуры, пороговое устройство, нагревательный элемент, представляемый в простейшем случае функциональной схемой (Рис.1.):

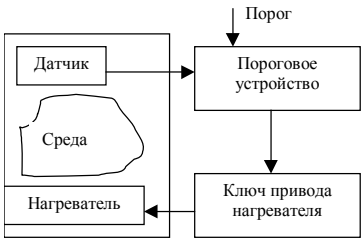


Рис. 1

Алгоритм работы такой системы и представляющая ее модель являются простыми для данного уровня их представления.

Реальные промышленные системы, хотя и могут быть представлены на верхнем уровне описания подобной схемой, значительно сложнее из-за особенностей реализации. В частности, датчик является распределенной системой отдельных устройств измерения температуры, объединяемых в единое целое линиями связи, цифровой, аналоговой, распределённой, со своими протоколами передачи данных. Пороговое устройство может быть многоканальным по входу, порог является вектором, т.е, требуется обеспечить температурное распределение по объекту. Привод нагревательными элементами также является многоканальной распределенной системой. Математическая модель описания алгоритма управления такой системой становится не столь тривиальной, как в первоначальном примере, а с учетом влияния особенностей реализации аппаратуры ее адекватность и полезность вообще вызывает сомнения.

2 Архитектура АСУТП

2.1 Определение архитектуры

Архитектура - совокупность факторов, делающих комбинацию радиотехнических устройств и модулей единой системой для решения задач, определяемых ее функциональным предназначением.

Понятие архитектура описывает форму, структуру и взаимосвязь между элементами системы и ее окружением: функциональным описанием верхнего уровня иерархии (общая функциональная схема), топологией системы, протоколами коммуникаций, интерфейсами, процедурами системного контроля и управления, конфигурацией вычислительной системы.

2.2 Составляющие архитектуры

Вычислительная система.

Характеризуется типами применяемых процессорных элементов и узлов для выполнения различных этапов обработки данных, пропускной способностью вычислительных блоков, временем реакции на входные данные (latency time), типами, размером и организацией устройств памяти, примененным программным обеспечением, моделью обработки данных, алгоритмами обработки.

Коммуникации.

Характеризуются особенностями потоков данных в системе, протоколами обмена данными, пропускной способностью передачи данных и временем реакции, используемыми шинами, интерфейсами внутри системы, внешними интерфейсами.

Топология (конфигурация).

Характеризуется стилями объединения главных компонентов системы, количеством и положением устройств обработки, конструктивным исполнением, использованием встроенных средств отладки, самотестирования, аварийного восстановления.

Дополнительные компоненты.

Включают в себя характеристики программного обеспечения, параметры конструктивов АСУТП.

2.3 Архитектура АСУТП с точки зрения обработки входных данных во времени

1) Время-зависимая обработка – предварительная фильтрация, декодирование, кодирование, цифровое формирование квадратур.
2) Объектно-ориентированная обработка – выделение из предварительно обработанных данных с датчиков информативных параметров, характеризующих состояние объекта (многоальтернативное обнаружение, оценка параметров).
3) Функционально-зависимая обработка – обработка выделенной информации по алгоритмам, определяющим функциональное назначение системы представление данных, алгоритмы принятия решений, функции управления.

2.4 Организация вычислений в АСУТП

Комплексные АСУТП, в состав которых входят АСУТП, принципиально состоят из большого количества вычислительных, коммуникационных, интерфейсных элементов. Данные в таких системах одновременно обрабатываются

в различных их составляющих, поэтому можно рассматривать такие системы с точки зрения организации параллельных вычислений.

Архитектуры АСУТП могут различаться:

1) Гранулярность обработки – хорошо разделенная (fine-grain), слабо разделенная (coarse-grain) - распределение объема вычислений по вычислительным элементам с точки зрения оформления процедур, составляющих алгоритма обработки. Fine-grain – блок алгоритма помещается в отдельный вычислительный элемент, производится достаточно большое количество операций над определенным блоком данных. Coarse-grain – малое количество операций над интенсивно поступающими данными.
2) Зависимость отдельных процессоров в отношении получения инструкций и данных – Single Instruction Multiple Data, Multiple Instruction Multiple Data.
3) Сильно связанные и слабо связанные архитектуры вычислительных элементов – характеризуют степень взаимосвязанности процессоров в отношении получения доступа к ресурсам вв/выв, памяти, межпроцессорные коммуникации. Мультипроцессор, мультикомпьютер.
4) Гомогенность и гетерогенность вычислительной среды – степень унификации применяемых вычислительных элементов.
5) Топология межпроцессорного взаимодействия – организация коммуникаций между вычислительными элементами в системе.
6) Эти параметры архитектуры АСУТП определяющим образом влияют на потребительские характеристики конечной системы, такие как стоимость, надежность, тестируемость, масштабируемость, отказоустойчивость…

2.5 Архитектура АСУТП с точки зрения тестируемости

Для обеспечения интегральной тестируемости АСУТП необходим иерархический подход к организации проверки функционирования элементов системы на различных уровнях – компонент, модулей, блоков.

1) Компоненты. До 95% всех компонент в системах управления критическими процессами должны обладать встроенной функцией и аппаратурой тестирования. Стандарт IEEE1149.1 определяет спецификации “встроенного самотестирования” – Built-In-Self-Test, “порта доступа к тесту” – Test Access Port, и способы объединения TAP вместе – Boundary Scan Architecture.
2) Модули. Модули должны содержать специальные контроллеры (Built In Test), обслуживающие TAP порты отдельных элементов. Функции контроллеров – обеспечение доступа к тестированию элементов по определенным производителем алгоритмам контроля работоспособности и обеспечение интерфейса с системой тестирования и настройки верхнего уровня.
3) Блоки. Блоки объединяются между собой специальной шиной Test & Maintenance Bus – шиной настройки и тестирования. TM-bus служит для обеспечения комплексной проверки системы и может использоваться для осуществления динамической переконфигурации аппаратуры и/или алгоритмов работы. Определяется стандартом IEEE1149.5.

Коммерческим стандартом, соответствующим IEEE1149.1 является JTAG-интерфейс. В системах специального назначения может содержаться дополнительные системы тестирования и настройки.

2.6 Эволюция архитектур АСУТП

На различных этапах развития радиоэлектронных и вычислительных технологий существовали различные подходы к организации АСУТП.

1) Независимая архитектура – все компоненты – независимые, специализированные, “уникальные”. Отличаются сложностью обслуживания, немасштабируемостью, высокой стоимостью.
2) Централизованная архитектура – повышена унификация блоков, модулей, интерфейсов, система интегрируется на блочном принципе объединения с централизованным управлением по низкоскоростной шине управления.
3) Интегрированная архитектура – дальнейшая унификация модулей и блоков, снижение номенклатуры модулей, интерфейсов, компонент программного обеспечения. Снижение зависимости между аппаратурой и решаемой ею функциональными задачами.
4) Распределенная архитектура – разнесение вычислительных ресурсов в пространстве с сохранением сильной связанности вычислений. Требует высокопроизводительной коммуникационной аппаратуры повышенной надежности.
5) Архитектура “Открытая системы” - использование полностью стандартной коммерчески доступной аппаратуры, интерфейсов, программного обеспечения, отличаются повышенной масштабируемостью, высокой ремонтопригодностью, способностью к модернизации.

2.7 Примеры архитектур

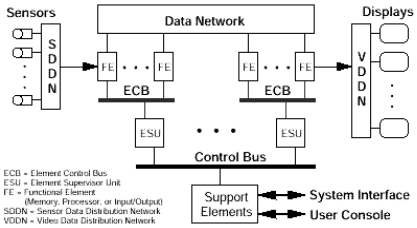


Рис. 2 Модульная, открытая, отказоустойчивая и высоко коммутируемая архитектура. Применялась в 80-х годах в ВВС США.

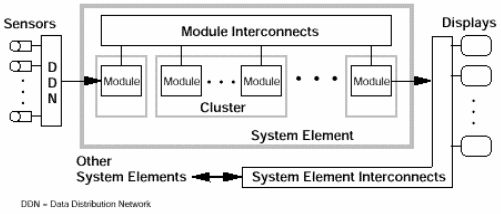


Рис. 3 Улучшенная авиационная архитектура. Advanced Avionics Architecture – A^3. Использует Common Integrated Processor фирмы Hughes. Устанавливался на F-22, LHA, Navy A-X.

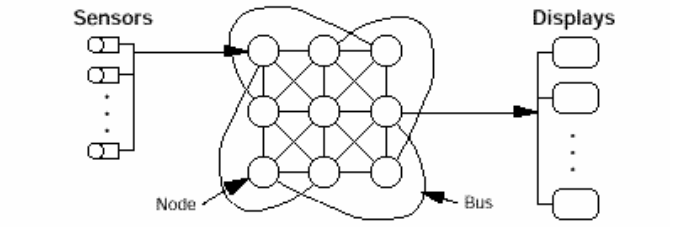


Рис. 4 Улучшенный бортовой сигнальный процессор

AOSP. Все узлы являются идентичными вычислительными элементами. В каждом узле – ядро сетевой операционной системы, выполняющие функции маршрутизации данных, запуск задач, обработка ошибок. Система действует под управлением глобальной операционной системы

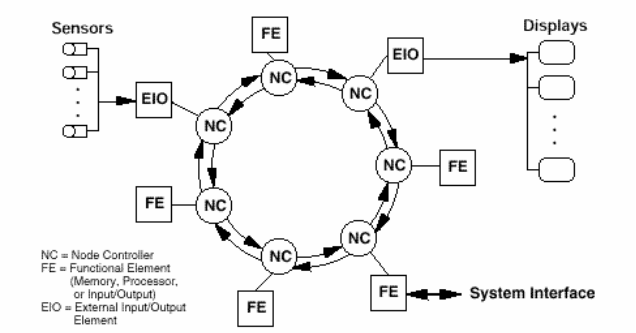


Рис. 5 Survivable Adaptable Fiber Optic Embedded Network – SAFENET

Восстановимая адаптивная фиброоптическая встроенная сеть – сочетает достаточную надежность с высокой производительностью и умеренной ценой, масштабируется, содержит унифицированные элементы

2.8 Влияние особенностей архитектуры АСУТП

В конечном итоге, именно архитектура определяет все потребительские качества АСУТП: надежность, отказоустойчивость, ремонтопригодность, тестируемость, стоимость и т.д. С другой стороны, выбранная архитектура определяет удельную стоимость разработки аппаратного, программного обеспечения и время разработки. Системы с независимой и иерархической архитектурой требуют больших затрат на разработку аппаратуры, в то время как архитектуры типа гиперкуб или SAFENET требуют больших временных и стоимостных затрат на симуляцию (моделирование) работы конкретного алгоритма обработки и управления на конкретной топологии вычислительной среды.

3. Типы вычислительных элементов АСУТП

В вычислительных системах АСУТП могут применяться самые различные типы процессоров, оптимизированные на выполнение различных функций – высокопроизводительные элементарные вычисления (сортировка массивов, умножение на скаляр или вектор, накопление), обработка по сложным алгоритмам в реальном времени, обработка данных не в реальном времени, выполнение типовых математических операций и т.д. По типу вычислений, т.е. функционально, вычислительные элементы подразделяются на несколько основных типов.

3.1 Препроцессоры

Ориентированы на выполнение простых математических и логических операций над высокоскоростными потоками данных с датчиков. Аппаратно настроены на выполнение одной или нескольких операций – аппаратный вычислитель, в некоторых случаях программно переконфигурируемый.

1) Sort Enchanced Preprocessor Element – препроцессорный элемент с расширенными возможностями сортировки – отличался аппаратной реализацией оконного режима адресации данных во входных и выходных буферах ввода.вывода. Абсолютно не переориентируем на выполнение других операций.
2) Configurable-Hardware Algorithm Mapped Processor – процессор с конфигурируемой в соответствии с алгоритмом архитектурой – аппаратный вычислитель, управляемый программно. Имеет большие возможности по применению в различных системах благодаря более гибкой архитектуре.

3.2 Векторные и матричные процессоры

В качестве операнда рассматривается вектор или матрица. Матричные процессоры могут быть образованы как комбинация векторных. Простейший векторный процессор – поддерживающий операции с комплексной арифметикой.

1. Dual Signal Processor Element – применялся в CIP для комплексной обработки данных с SEPE.
2. Macro Function Signal Processor – сигнальный процессор макро функций (RAYTHEON) – комплексная обработка данных на языке макрофункций.

- Интеллектуальные порты памяти, позволяющие независимо осуществлять выборку 2 операндов из памяти с записью результата (3 операции вв/выв).
- Произвольная независимая адресация 4-х мерных матриц для выполнения обращения, транспонирования матриц, вычисления собственных чисел и т.д.

3. SHARP LH9124/9320 – чипсет, содержащий сигнальный процессор и адресный генератор.

- Данные – 48 разрядный операнд, представляющий собой комплексное число с плавающей точкой или целочисленное.
- Тактовая частота – 40 МГц, организация – конфигурируемый конвейер.
- 4 порта данных – 2 входных, 1 выходной, 1 специального назначения, 26 инструкций программирования(фильтр, спектральный анализ, адаптивный фильтр, распознавание).
- 6 умножителей, 8 сумматоров, 4 логических блока.

- 4096 комплексное БПФ выполняется за 313 мкс, с использованием 3 DSP – за 102 мкс.
- Адресный генератор 9320 независимо программируется для обеспечения непрерывной загрузки конвейера.

3.3 Цифровые сигнальные процессоры - DSP

DSP появились как компромисс между векторными процессорами и процессорами общего назначения. Ориентированы на выполнение операций типа “умножение и накопление” над входными данными, поступающими в реальном времени. Порядок реального времени для систем обработки на DSP процессорах – 0.1 – 1 мкс.

- Texas Instruments (TMS320C30, 40, 80, 62xx и 67xx, 64xx) – один из самых крупных производителей сигнальных процессоров. C30 – первый процессор с плавающей точкой, C40 – первый использует 6 параллельных коммуникационных портов (~20Мбайт/с каждый) для межпроцессорной коммуникации, C80 – первый сигнальный мультипроцессор – состоит из 1 RISC и 4-х 32 разрядных DSP, ориентирован на обработку видео в реальном времени. Процессоры целочисленные 62xx и с плавающей арифметикой 67xx первыми достигли производительности более 1 GOPS и первыми в мире выполнены с использованием технологии VLIW – Very Large Instruction Word – суть которой заключается в группировании в один “регистр” до 8 команд процессора для одновременного выполнения. Ассемблер этих процессоров впервые выполняет оптимизацию кода при трансляции программы, имеется транслятор для переноса программного обеспечения с C3x-C4x на C6xxx.
- Analog Devices – начиная с ADSP21020 выпускает процессоры с плавающей арифметикой, как правило, с отличающимися потребительскими характеристиками, чем TI. К примеру, ADSP2106x помимо коммуникационных портов имеет средства поддержки межпроцессорного взаимодействия с использованием механизма разделяемой памяти, ADSP2116x, имея меньшую (~в 1.5 раза) производительность, чем TMS320C6xxx, также поддерживает кластерную технологию межпроцессорного взаимодействия и имеет 4 параллельных порта.
- Motorola – выпускает большое количество целочисленных сигнальных процессоров, в конце 1980-х - начале 90-х годов выпускала сигнальный процессор с плавающей точкой 96002, от производства которого и дальнейшей разработки направления отказалась примерно в 1995 году. В настоящее время вместе с Lucent Technologies разработало 4 процессорное ядро сигнального процессора и выпускает на его основе
- Существует несколько других производителей сигнальных процессоров, вроде AT&T, Siemens, NEC, Zilog и т.п., но они ориентированы на внутри корпоративное или специализированное применение.

3.4 Контроллеры

Контроллерами принято называть процессоры, архитектура и система команд которых оптимизирована для решения задач управления вводом/выводом.

К контроллерам относятся RISC процессоры Intel 860,960 и процессоры общего применения Motorola 68xxx.

Первые используются в качестве интеллектуальных контроллеров узлов ATM-сетей, сетевых принтеров, для сопряжения подсистем с локальной шиной PCI с VME-шиной.

Процессоры семейства Motorola применяются в 70% продукции в стандарте VME, телекоммуникационном оборудовании, мед технике, промышленных системах управления.

В настоящее время массовое распространение получили контроллеры на основе RISC технологии. К ним можно отнести процессоры фирм ARM, Hitachi, PIC, Atmel, и мультимедиа-контроллеры TI.

MSP430F1xx. – мультимедийный сигнальный процессор. MSP430F149 – содержит весь спектр необходимых периферийных устройств для реализации высокопроизводительного контроллера аналого-цифровых приборов с интерфейсами сопряжения с другими устройствами.

Основные характеристики – тактовая частота – 32кГц-6МГц, 16бит, 2кБ – RAM, 60кБ+256Б – FLASH, 2 USART, 8 канальный 12-разр. АЦП 200 кГц, JTAG, аппаратное умножение 16x16, 0,1-400мкА, 1,8-3,6 В, 6 мкс – wakeup, Watchdog таймер, 2 таймера с 3 компараторами каждый и ШИМ выходом, прецизионный аналоговый компаратор.

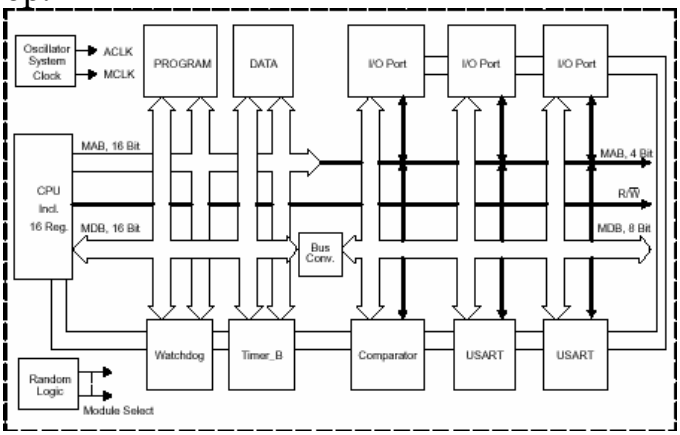


Рис. 6

3.5 Процессоры данных

Процессоры данных оптимизируются на одновременное решение всего спектра задач, связанных с обработкой данных: сортировка, сложные математические вычисления, обработка графики, работа баз данных, многозадачные многопользовательские защищенные системы, сервера различных типов.

Важнейшие параметры, определяющие возможность применения того или иного процессора в конечной системе, приведены в Табл.1 (примерно 1995 год)

Табл. 1

Процессор	Тактовая частота, МГц.	SPECint92	Потребление, Вт.	Технология, мкм.
Pentium Intel	2000	2000	40	0.26
Pentium Pro Intel	133	200	20	0.6
SuperSparc 2 Sun	90	135	14	0.8
PA7100 HP	100	80	23	0.8
PowerPC-604 IBM-Motorola	100	160	до 1 Вт. с использованием энергосберегающего режима	0.5
R4400	150	90	15	0.6

MIPS				
Alpha21064 DEC	275	170	33	0.5

3.6 Выводы:

1) Более высокая тактовая частота необязательно соответствует более высокой производительности – Pentium-100 и SuperSparc-2.
2) Показатель производительности SPECint92 не в полной мере отражает конечные свойства процессора для решения задач конкретного применения, к примеру PA7100, имеющий этот параметр всего 80 (сравните с Pentium), обладает 32 32-разрядными регистрами общего назначения, 28 64-разрядными регистрами для операндов с плавающей точкой и допускает подсоединение к шине межпроцессорного обмена Runway Bus с пропускной способностью 768 МБ/с.
3) Важнейшим показателем, характеризующим возможность “полевого” применения процессора во встроенных системах – потребляемая мощность. По этому показателю применение большинства из современных высокопроизводительных процессоров данных затруднено.

3.7 Реализация различных типов вычислительных элементов

В связи с успехами микроэлектроники в настоящее время во многом стираются различия в технологии изготовления вычислительных элементов различных типов.

Так, препроцессоры и векторные процессоры выполняются по технологиям ASIC и FPGA, ядра сигнальных процессоров совместно с ядрами контроллеров также встраиваются в заказные СБИС и поставляются в качестве библиотечных функций к САПР проектирования ASIC и FPGA. Последние серии FPGA позволяют разместить на одном кристалле и управляющий контроллер, и специализированный коммуникационный процессор, и ядро сигнального процессора. Таким образом, стирается грань между технологиями реализации различных по функциональному назначению вычислительных элементов. Тем не менее, различия в принципе работы и реализации этих устройств сохраняются.

4.Коммуникации в АСУТП

К коммуникациям относятся все аппаратно-программные средства, служащие для организации соединений между составляющими систему обработки и управления конструктивно выделенными частями – модулями, блоками, подсистемами. Коммуникации могут быть локальными (шины) или сетевыми (сети).

Шины могут быть подразделены по количеству линий передачи данных – последовательные, параллельные; по принципу коммутации – прямые соединения, разделение доступа, изменяемая конфигурация; по протоколам обмена – синхронный обмен, асинхронный с подтверждением, доступ по опросу, доступ при уведомлении и т.п.

Основным примитивом в коммуникациях между источником и приемником данных является порт - аппаратный или логический элемент, служащий для адресации передаваемых данных.

Характеристики существующих шин приведены на рис.

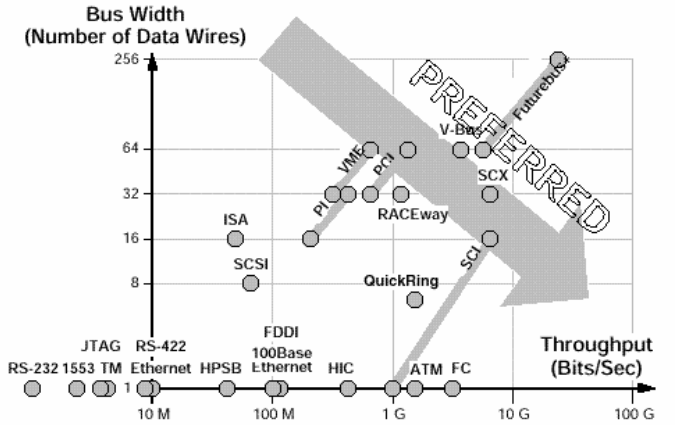


Рис. 7

В зависимости от выполняемых функций коммуникации подразделяются на следующие типы:

1) шины управления;
2) шины данных;
3) шины тестирования и настройки;
4) шины ввода-вывода;
5) сетевые соединения.

4.1 Шины управления

Шины управления ориентированы на организацию соединений нескольких модулей для обмена сигналами управления работой и временной синхронизации. Примеры современных шин – Табл.2.

Табл. 2

Шина	Пропускная способность, разрядность	Примечание
Future Bus+ IEEE 896.x ISO/IEC 10857:1994	3200 MB/s, 256 bit 100 MB/s, 32 bit	Шина для внутренних коммуникаций в высокопроизводительных системах, замена VME.
PI Bus Parallel Intermodule Bus	50 MB/s, 32 bit	Шина управления для применения в авиационных

		системах ВМС США
VME Versa Module Europe IEEE 1014-1987	40 MB/s, 32 bit	Промышленные системы обработки данных и управления.
VME-64 IEEE P1014.D	80 MB/s, 64 bit	Последняя ревизия стандарта VME.
cPCI	132-256 МБ/с, 32-64 бит 528МБ/с – 66МГц	Замена VME в компактных промышленных системах
PC-104+	132-256 МБ/с, 32-64 бит	PCI. Расширение PC-104

FB+

Шина с разделяемым доступом, асинхронным подтверждением адресов и данных, позволяет объединить в одном шасси 32 модуля, базовая разрядность – 64 с возможностью снижения до 32 (с дублированием) и расширения по данным до 128 или 256. Использует технологию Backplane Transceiver Logic – активная шина, уменьшение переходных процессов, уменьшение монтажных емкостей, повышение порогов срабатывания, может работать в режимах синхронного и асинхронного обменов на частоте до 100 МГц.

Программируется:

1) логика арбитража – централизованный, распределенный;
2) режим обмена – пакетный, одиночный;
3) доступ к ресурсам – приоритезированный, равноправный.

Реализован механизм когерентного кэширования для мультипроцессорных систем с разделяемой памятью, “горячая” замена блоков, интерфейс Control Status Register (IEEE 1212).

Профили FB+:

A. общего назначения;
B. ввод-вывод;
C. кабель;
D. настольная система;
E. рабочая станция;
M. военная
T. телекоммуникации.

Недостатки:

1) многоразрядность;
2) высокая удельная цена на узел;

PI Bus

Разделяемая шина, рассчитанная на применение в системах реального времени, отказоустойчивая. Разрядность – 16 или 32, 32 узла, синхронная, активная, согласованная (по импедансу), поддерживает передачу сообщений. Арбитраж доступа – мастер или подчиненный, обработка ошибок – обнаружение или коррекция. Приоритет доступа определяется приоритетом задачи и номером устройства. Тактовая частота – 12,5 МГц, используется BTL. Имеется “Шина служебных сигналов” с аппаратной поддержкой следующих функций:

1) сброс;
2) оповещение о сбое питания;
3) режим доступа пользователя;
4) точка останова системы;

5) отладочный режим шины;
6) холодная перезагрузка;
7) разрешение записи в общую память;
8) запуск узла N;
9) идентификация узла.

Имеется встроенная поддержка отказоустойчивости программного обеспечения, защита памяти, 100% выявление ошибок обработки сигналов шины, программируемая система контроля, автоматическое отключение модулей при аварии. Реализован принцип аппаратного подтверждения передач для реализации протокола коммутации по правилу “передал и забыл” с аппаратной проверкой ошибок. Вследствие своих возможностей применяется в системах высокоскоростной обработки с сильно связанными вычислениями.

VME

Самая распространенная шина среди систем промышленной автоматизации и обработки данных. Позволяет объединить до 21 узла, используется в 2 исполнениях 3U и 6U. Пассивная разделяемая согласованная шина с тремя состояниями, новая ревизия поддерживает горячую замену модулей. Арбитраж доступа – программируемый с использованием специальной шины арбитража, имеются шина приоритетных прерываний, шина служебных сигналов. Конструктивно шасси имеет 2 разъемных штырьковых соединительных разъема P1 и P2, причем P1 является непосредственно 16 разрядной шиной VME, а P2 служит для расширения разрядности шины до 32 и/или размещения шин расширения:

1) VMX – шина расширения памяти;
2) VMS – последовательная шина;
3) VSB – высокоскоростная шина расширения ввода-вывода;
4) VXI – шина расширения инструментальных измерительных систем.

Compact PCI

Pereferial Component Interconnect – PCI, cPCI - +Евромеханика (разъёмы 302 контакта – 5 рядов). Синхронная шина, поддерживает передачу неограниченного блока данных (BURST), управление обменом со стороны периферийных устройств. Шина Адреса/Данных мультиплексирована, прерывания, пространства адресов ввода-вывода, памяти, конфигурационное пространство модулей. IEEE 1149.1. Арбитраж обменов по таймеру.

4.2 Шины данных

Шины данных дополняют возможности шины управления в системе по передаче данных между процессорами, памятью. Топологией шин данных, как правило, является двухточечное соединение.

Параметры двух различных по принципам организации шин данных приведены в табл. 3.

Табл. 3

Шина	Пропускная способность, разрядность	Примечание
SCI Scalable Coherent Interface IEEE 1596-1992	1GB/s, 16 bit 128 MB/s, 1bit	Используется для построения гетерогенных параллельных процессоров
RACEway	160 MB/s, 32 bit	Расширение VME

Mercury Computer Systems, Inc		

SCI

Сеть данных с однонаправленными двухточечными соединениями, используется двухфазная (по положительному и отрицательному фронтам) синхронизация передачи. Возможна топология кольца, набора колец и коммутируемая.

Адресное пространство – 64 бит, 16 из которых – адрес узла. Адресуемая единица – байт, фундаментальная единица – символ – 16 бит + 1бит флаг. Передачи ведутся пакетами по 40 символов, состоящими из 8 байт заголовка, 64 байт данных и циклически уменьшающегося кода (счетчик мостов). Данных может также быть 0, 16 или 256 байт (См. рис. 8).

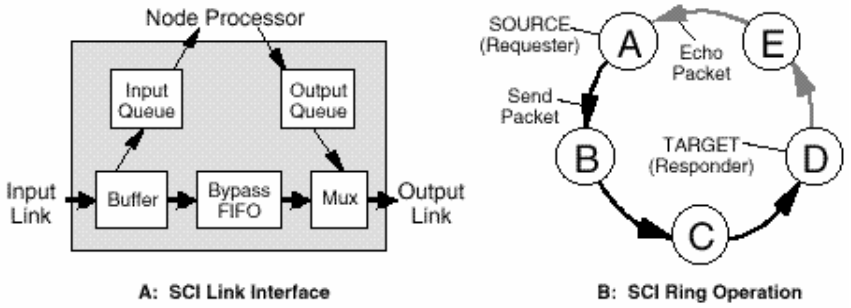


Рис. 8

RACEway

Используется в качестве шины данных в высокопроизводительных системах обработки на основе шин VME.

Основным элементом шины является микросхема коммутатор RACE, имеющая 6 32 разрядных портов ввода-вывода и 8 разрядную шину управления работой микросхемы. Реализуется переключаемая сеть с вытеснением. Инициатор передачи данных начинает передачу с установления пути соединения через, возможно, несколько коммутаторов. Затем передается блок данных из до 256 64 разрядных слов. В случае, если данная передача мешает более высокоприоритетному обмену, то она прерывается – “вытесняется”, и происходит более приоритетный обмен, после которого перезапускается прерванная. Имеется 3 уровня приоритета.

4.3 Шины тестирования и настройки

Предназначены для обеспечения монтирования системы, тестирования аппаратуры и алгоритма, переконфигурирования и т.д.(табл. 4).

Табл. 4

Шина	Пропускная способность	Примечание
HPSB IEEE P1394	6 MB/s – bus 50 MB/s –cable	Используется совместно с FB+
TM Bus Test & Maintenance Bus	0.8 MB/s	Используется совместно с PI Bus
MTM Bus Module Test & Maintenance Bus IEEE 1149.5	1.2 MB/s	Используется для соединения JTAG контроллеров
JTAG IEEE 1149.1	3 MB/s	Локальная шина объединения TAP-портов

		элементов

MTM Bus

Служит для объединения контроллеров JTAG для построения системы тестирования и настройки.

“Slave” в данном случае – это как раз JTAG контроллер.

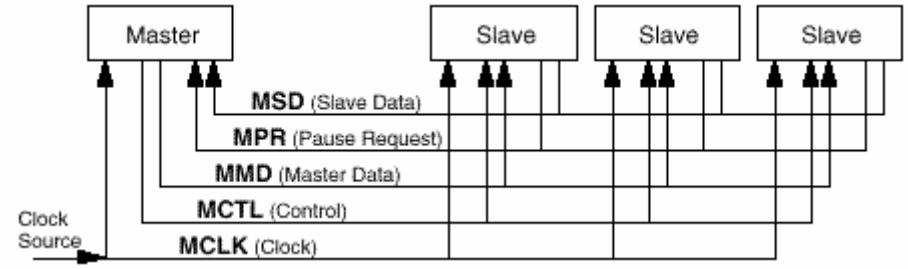


Рис. 9

JTAG

Объединяет TAP порты отдельных элементов модулей в одну цепочку. По JTAG интерфейсу возможна загрузка ПЗУ, ПЛИС, процессоров программами, подключение эмуляторов, реализация многопроцессорных многооконных отладочных сред разработки программного обеспечения и т.д.

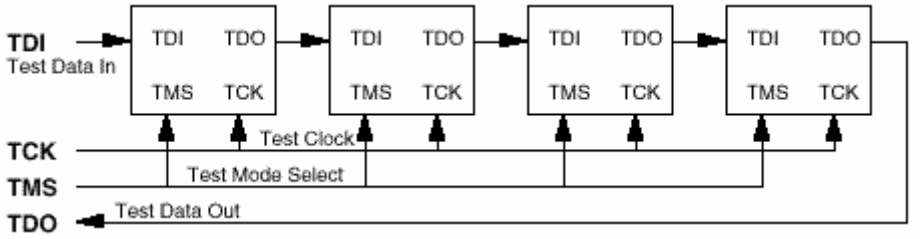


Рис. 10

4.4 Шины ввода-вывода

Шины ввода-вывода служат для подключения к системе датчиков ввода-вывода и оптимизированы для передачи больших блоков данных или постоянных потоков данных с минимальной проверкой ошибок передачи-приема (см. табл. 5).

Табл. 5

Шина	Пропускная способность, разрядность	Примечание
Fiber Channel	12.5 – 100 MB/s, 1bit до 530 MB/s	Датчики видео ввода-вывода
SCSI Small Computer System Interface	10 MB/s – 8 bit 20 MB/s – 16bit 40 MB/s – 32 bit	Подключение периферийных устройств к компьютерным системам
RS-422	1 MB/s	Замена RS-232
MIL Std 1553B	0.1 MB/s	
RS-232	0.02Mb/s	

FC

Шина последовательная двунаправленная, предназначенная для предсказуемой передачи данных большими блоками. Передача ведется по выделенным логическим каналам и без создания канала – дейтограммы. Элементом коммуникаций является пакет, состоящий из 24 байт заголовка и до 2048 байт данных.

SCSI

Наиболее распространенная шина для подсоединения периферийных устройств к компьютерным системам, служит для подключения до 7 устройств к одному контроллеру по разделяемому каналу связи. Используется распределенный арбитраж доступа к шине. Существует 4 ревизии стандарта – SCSI (тактовая частота 4 МГц, разрядность 8 бит), FastSCSI (тактовая частота 10МГц, разрядность 8 бит), SCSI-2 (16 разрядная) и UltraSCSI (32 разрядная)

Mil Std 1553

Digital Time Division Command/Response Multiplex Data Bus получила широкое распространение в военной и авиационной технике для объединения блоков и датчиков в иерархическую систему с централизованным управлением. Шина согласованная (75 Ом), устройства подключаются через трансформаторную развязку, для передачи используется двухфазный код Манчестер-2. Возможно подключение к одному контроллеру до 30 узлов, каждое 16 разрядное слово данных преобразуется в 20 разрядный последовательный код. По шине передаются команды, данные и статусные слова.

RS-232, 423, 422, 485

RS-232 служит для выделенного подключения устройств на удалении до 15 м со скоростью обмена до 20 килобит/сек.

RS-423 является развитием RS-232 и увеличивает количество приемников данных до 10 с повышением скорости обмена до 100 килобит/сек.

RS-422 строится на основе дифференциальных приемопередатчиков и позволяет увеличить пропускную способность до 10 Мбит/сек.

RS-485 похож на 422, но использует трехстабильные передатчики, что позволяет объединять по одной разделяемой шине до 32 приемников и 32 передатчиков.

4.5 Сетевые соединения

Сетевые соединения служат для объединения распределенных систем обработки и управления (см. табл. 6).

Табл. 6

Шина	Пропускная способность	Примечание
ATM Asynchronous Transfer Mode	300 MB/s	Телекоммуникации, локальные сети
FDDI Fiber Distributed Data Interface	12 MB/s	SAFENET
100Base Ethernet IEEE P802.14	12 MB/s	Замена 10Base Ethernet
10Base Ethernet IEEE 802.3	1.2 MB/s	Локальные сети
Myrinet	40 MB/s (8 разрядная шина)	Высокопроизводительные сети

ATM

Основная технология реализации современных локальных и глобальных сетей. ATM является набором логических протоколов для организации передачи больших

объемов данных с частотным разделением диапазона по разделяемому носителю – двухточечным соединениям, коммутируемым сетям.

Элемент коммуникации – пакет фиксированной длины, состоящий из 5 байт заголовка и 48 байт данных.

Заголовок адресует виртуальный путь, виртуальный канал связи и/или функции управления сетью. Поддерживается статическое и динамическое образование виртуальных каналов связи, а также обмен дейтограммами. ATM может базироваться на различных физических интерфейсах – Fiber Channel, FDDI.

4.6 Шины персональных компьютеров

Предназначены для использования в дешевых персональных компьютерах и, как следствие, объединяют в себе свойства шин управления, данных, ввода-вывода (см. табл. 7).

Табл. 7

Шина	Производительность	Примечание
PCI, compactPCI Pereferial Component Interconnect Intel	100 MB/s, 32 bit	PCI – число слотов –4 CPCI – до 8
VLB VESA Local Bus	100 MB/s, 32 bit	Устаревающий стандарт
V-bus	500 MB/s, 64 bit	Построение многопроцессорных недорогих систем
EISA	16 MB/s, 32 bit	Малое распространение
ISA	8 MB/s, 16 bit	Широко распространена

5. Топология сетей в АСУТП

Конфигурация системы обработки данных описывает, как вычислительные и коммуникационные элементы объединяются между собой. Поскольку это описание включает в себя много факторов, таких как подключение датчиков, конструктивы исполнения аппаратуры, имеет смысл описывать ее как топологию сетевого объединения как первичного фактора. Основными типами топологий (см. рис.) являются: линейная, кольцевая, коммутируемая и решетчатая. В реальных системах могут применяться комбинированные топологии, сочетающие в себе несколько основных типов.

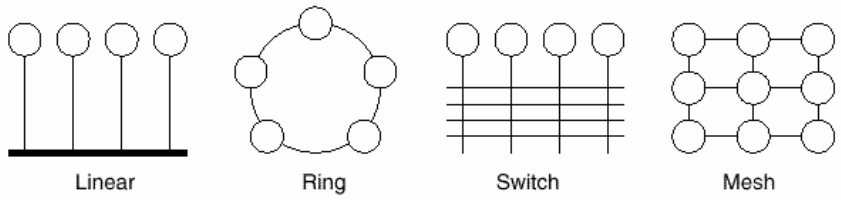


Figure 2.2.3-1: Basic Topology Types

Рис. 11

5.1 Линейная топология

Линейная топология применяется во всех системах на основе разделяемых шин – VME, Mil Std 1553B, Ethernet. К недостаткам этой топологии можно отнести влияние отказов узлов на функционирование системы и наличие физических ограничений на количество объединяемых узлов.

Для преодоления этих недостатков используют линейные иерархические топологии.

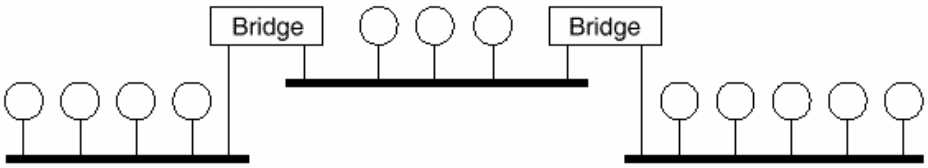


Figure 2.2.3.1-1: Hierarchical Topology

Рис. 12

5.2 Топология кольца

Кольцевую топологию отличает более высокая производительность и масштабируемость. Кроме того, этот тип топологий отличает большая временная детерминированность работы. К недостаткам можно отнести малую отказоустойчивость. Для повышения надежности используются модифицированные кольцевые топологии (см. рис.).

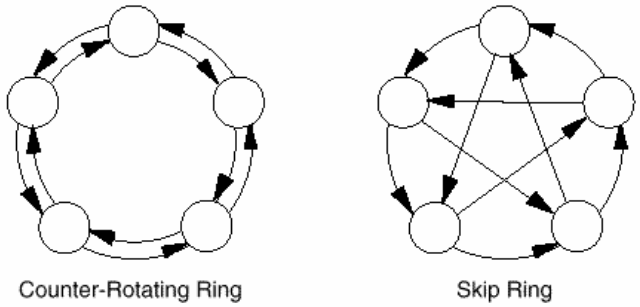


Figure 2.2.3.2-1: Dual Ring Topologies

Рис. 13

5.3 Коммутируемые топологии

Коммутируемые сети и шины достигают максимальной производительности обмена в связи с аппаратной поддержкой маршрутизации пересылок данных. Существуют топологии с полной коммутацией на основе матричных мультиплексоров и топологии звезды, где существует центральный коммутатор, а абоненты расположены в “лучах звезды”. Коммутируемые топологии отличает повышенная удельная стоимость, поэтому применяют модифицированные решения (см. рис.)

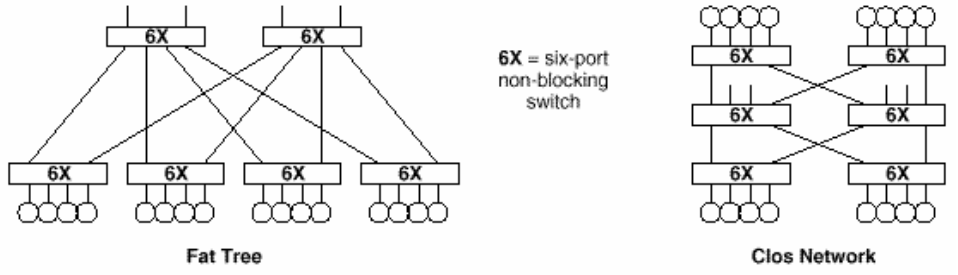


Figure 2.2.3.3-1: Example Switch Topologies

Рис. 14

5.4 Топология решетки

Топологии решеток сочетают в себе высокую масштабируемость, повышенную отказоустойчивость, хорошую детерминированность обмена с меньшей удельной стоимостью. Существует большое количество вариаций решетчатых топологий (см. рис.).

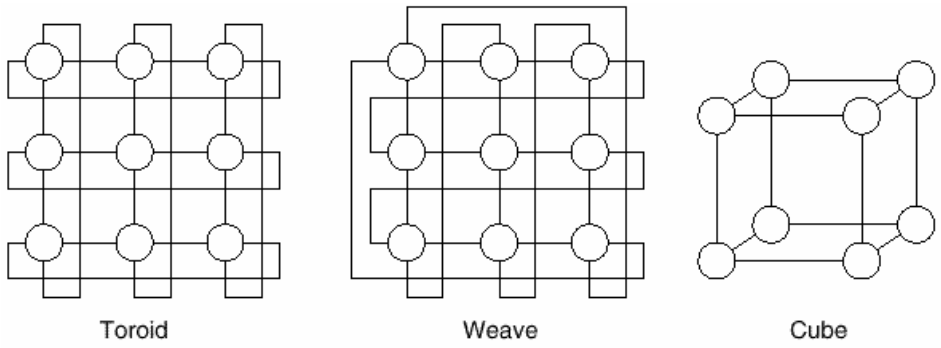


Figure 2.2.3.4-1: Mesh Variant Topologies

Рис. 15

6. Типы архитектур АСУТП

С развитием радиоэлектроники изменялись подходы к построению систем цифровой обработки данных.

По мере повышения производительности вычислительных устройств появилась возможность использования унифицированных модулей и блоков, которые программировались на выполнение пользовательских прикладных алгоритмов, что привело и приводит к появлению новых архитектур.

Минимальная архитектура

Минимальная архитектура используется в простейших сосредоточенных системах обработки и управления.

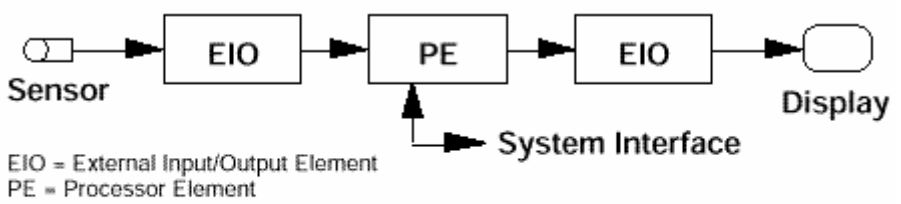


Figure 2.3.1-1: Minimal Architecture

Рис. 16

6.1 Архитектура прямого соответствия

Системы с этой архитектурой имеют однозначное соответствие между проблемным алгоритмом работы и функциональной схемой построения системы. Используется специализированные коммуникационные интерфейсы, слабо унифицированная аппаратура. Результирующая система не пригодна для масштабирования или модернизации.

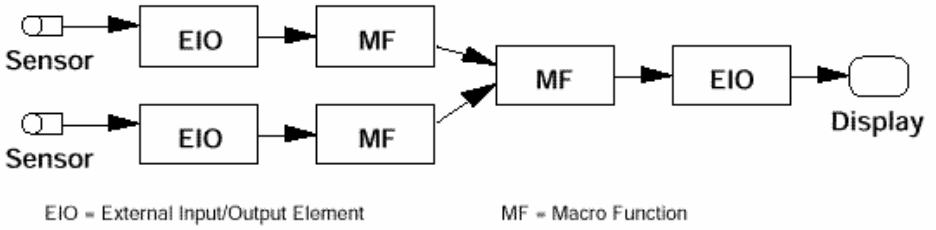


Figure 2.3.2-1: Direct Mapped Architecture

Рис. 17

К достоинствам такой системы можно отнести оптимальность использования аппаратуры (вследствие ее специализированности) и логическую понятность ее функционирования и разработки.

6.2 Архитектура с разделяемой шиной

В системе используется разделяемая шина, объединяющая достаточно унифицированную аппаратуру. Исчезает связь между топологией системы и проблемным алгоритмом ее работы. Подобные системы хорошо модернизируются , масштабируются в некоторых пределах.

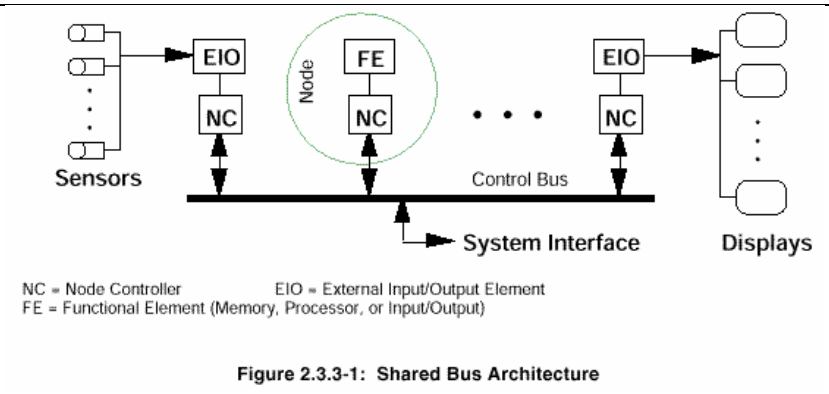


Figure 2.3.3-1: Shared Bus Architecture

Рис. 18

6.3 Кольцевая архитектура

Системы с кольцевой архитектурой имеют лучшие параметры производительности, чем с разделяемой шиной, и позволяют реализовать распределенную вычислительную среду обработки данных и управления.

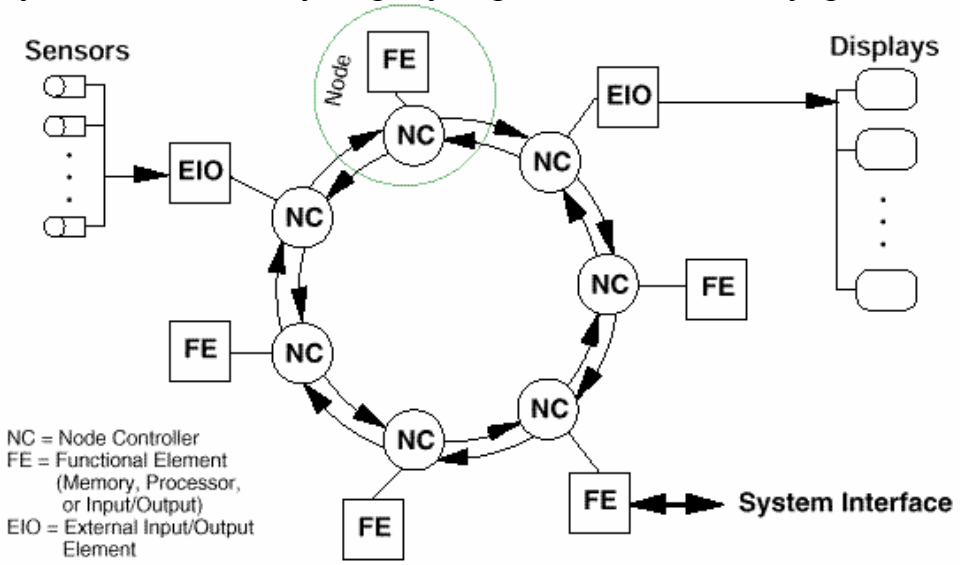


Figure 2.3.4-1: Ring Architecture

Рис. 19

6.4 Коммутируемая архитектура

Использует для обеспечения коммуникаций какую-либо коммутируемую сеть или шину. Отличается повышенной стоимостью и слабой масштабируемостью.

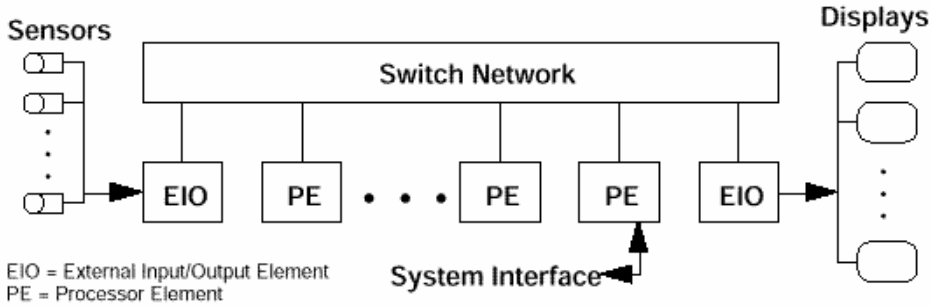


Figure 2.3.5-1: Switch Architecture

Рис. 20

6.5 Решетчатая архитектура

Обладает всеми достоинствами системы с решетчатой топологией. К недостаткам можно отнести сложность разработки подобных систем и сложность системного программного обеспечения, необходимого для ее работы.

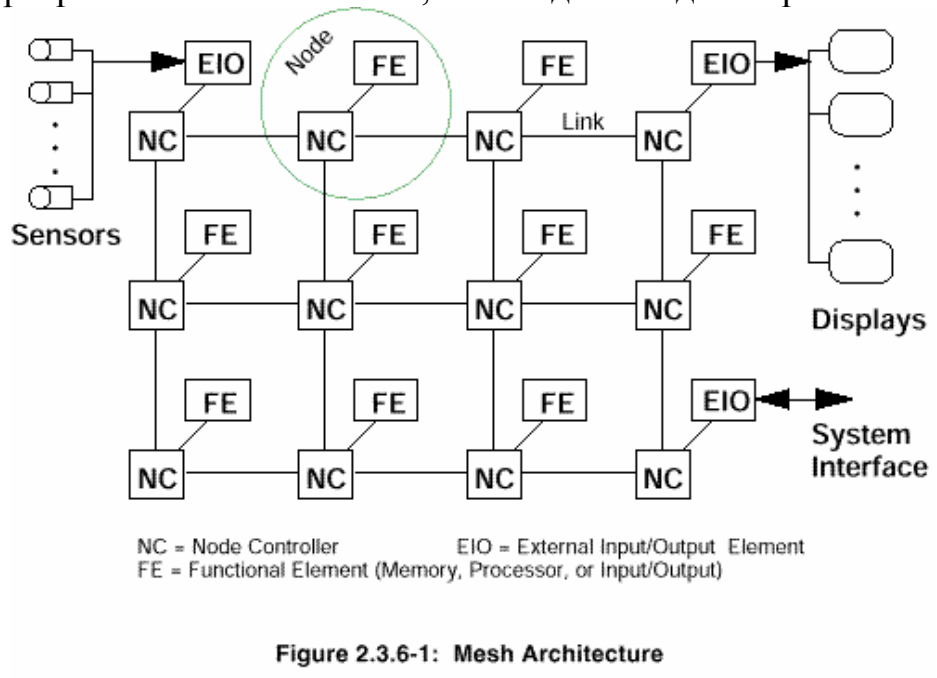


Figure 2.3.6-1: Mesh Architecture

Рис. 21

6.6 Комбинированная архитектура

Реализация в системе комбинированной архитектуры позволяет достичь более эффективных параметров для отдельных ее составляющих.

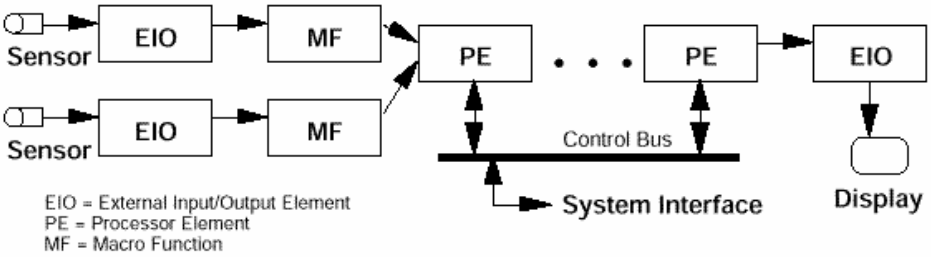


Figure 2.3.7-1: Mixed Architecture: Direct Mapped and Shared Bus

Рис. 22

7 Архитектура программного обеспечения АСУТП

Архитектура программного обеспечения (ПО) является прямым результатом принятых на стадии системного проектирования и разработки аппаратуры решений. ПО сочетает в себе конструктивные и архитектурные элементы.

Архитектурные элементы ПО отражают взаимосвязь аппаратуры и программного обеспечения, имеющие долговременную стабильность и отражающие функционирование алгоритмов обработки.

Конструктивные элементы ПО связаны с конкретной реализацией отдельных составляющих ПО, определяемой спецификой конкретных (в данной системе) аппаратных решений. Примером конструктивных элементов может являться конкретная для применяемой операционной среды реализация драйверов устройств ввода/вывода.

При исполнении АСУТП в стандарте “Открытая система” разработанное программное обеспечение также должно нести черты, присущие стандарту: масштабируемость, конфигурируемость. В части программного обеспечения это означает, что все виды интерфейсов – физические, электрические, системные, функциональные, временные, контрольные – должны быть программно управляемыми или настраиваемыми. Введение в систему дополнительного оборудования или замена существующего на другое должно приводить только к необходимости переконфигурирования программного обеспечения, а не к его модернизации.

Задачи, решаемые программным обеспечением

1. Задачи операционной системы.
2. Задачи пользовательских приложений.
3. Задачи управления системой:

3.1. инициализация аппаратуры;
3.2. загрузка аппаратуры выполняемыми программами;
3.3. запуск программного обеспечения;
3.4. встроенная диагностика;
3.5. переконфигурация;
3.6. режим аварийного функционирования;
3.7. сообщение об ошибках;
3.8. обработка исключительных ситуаций;

Архитектура программного обеспечения упорядочивает поток данных через систему, определяет порядок и структуры хранения данных, способ модификации этих структур для решения задач обработки, управления и контроля.

8 Уровни стандартизации программного обеспечения

По мере того, как росла производительность вычислительных устройств и снижалась их удельная стоимость в составе систем автоматизации и обработки данных, возрастала трудоемкость (в силу повышения сложности и комплексности решаемых задач) и стоимость программного обеспечения. Одной из причин этого являлась разнородность применяемых аппаратных платформ и отсутствие единого стандарта на программное обеспечение.

За все время существования вычислительной техники предпринималось несколько попыток стандартизации программного обеспечения.

ISA

Первым из этих подходов была попытка стандартизации системы команд компьютеров – ***Instruction Set Architecture*** – ISA. Вычислительное устройство, по этой идеологии, являлось конфигурируемым процессором, изменяющим свою топологию в зависимости от используемой аппаратно независимой системы команд. Программист работал только с программной моделью компьютера – регистры, блоки памяти, адреса ввода/вывода, система команд.

Таким образом, путем переконфигурирования аппаратуры достигалась переносимость программного обеспечения. С появлением микропроцессорных устройств данная идеология не позволяла эффективно использовать процессор в связи с его аппаратной простотой. Тем не менее следы этого подхода остаются в существующих интерпретируемых и динамически компилируемых языках программирования.

API (OSI)

Следующий подход в стандартизации программного обеспечения возник с появлением развитых операционных систем семейства UNIX и подобных им по функциональности и связан с применением в качестве базового языка системного программирования языка C. Стандартизации подвергался ***Application Program Interface*** (API) или ***Operating System Interface*** (OSI по терминологии IBM corp.)- вызовы процедур операционной системы со стороны пользовательских приложений.

Таким образом, программа, написанная на языке высокого уровня могла вызывать стандартные функции системных библиотек, отвечающие за взаимодействие с аппаратурой и операционной системой.

Данный подход закреплен в стандартах группы ANSI POSIX-1003 (Portable Operating System Interface). Эта группа стандартов определяет набор системных вызовов, доступных пользовательскому приложению, перечень пользовательских утилит командной строки для решения типовых задач управления приложениями, требования по обеспечению работы в реальном времени (POSIX 1003.1b) и требования по обеспечению много потокового (multithread) программирования (POSIX 1003.1.c).

Начиная с версии Windows NT 4.0 фирма Microsoft проявила желание обеспечивать совместимость с вышеупомянутым стандартом, хотя и с обычным для этой фирмы своеобразием.

SPI

С появлением операционной системы нового поколения – OSF/1 (разработана на основе Mach) – функционально масштабируемой по сети, со встроенной отказоустойчивостью, на основе динамически масштабируемого ядра, для

обеспечения аппаратной независимости компонент системного программного обеспечения появился стандарт System Program Interface (SPI). SPI устанавливает стандарты на интерфейс взаимодействия компонент ядра операционной системы.

Языки программирования

Большинство языков программирования высокого уровня являются стандартизованными тем или иным комитетом ANSI и другой подобной организацией (ANSI C, ANSI C++, ANSI Fortran, ANSI HPFortran, ANSI HPC). Однако, в связи со специализацией языков программирования на решение конечного ряда задач, трудно применить какой либо один язык программирования для разработки программного обеспечения системы обработки данных, управления и контроля.

С этой точки зрения можно предъявить несколько требований к языку программирования:

1) поддерживать разрабатываемую программу в удобочитаемом виде;
2) позволять сконцентрироваться на логике алгоритмов и вместе с этим прорабатывать детали особенностей реализации;
3) обладать заложенными возможностями автоматизации документирования.

В ВВС США долгое время был разрешен только язык ADA для разрабатываемого по линии этого ведомства программного обеспечения.

В настоящее время происходит некоторая революция в области языков программирования для встраиваемых приложений (программное обеспечение промышленных автоматизированных систем и контроллеров) – активно внедряется язык JAVA в качестве интерпретируемого и/или динамически компилируемого языка для написания программ. В качестве окружения для выполняемых программ на JAVA используется система виртуальных машин JavaOS или аппаратная поддержка JAVA, закладываемая в контроллер или процессор.

9 Организация процесса разработки программного обеспечения

Основным требованием к процессу разработки программного обеспечения для систем обработки, контроля и управления является необходимость интеграции его с процессом разработки аппаратуры для этой системы. Программное обеспечение является лишь одной, хотя и немаловажной компонентой в результирующей системе.

9.1 Структурное программирование

- это разработка и тестирование по мере разработки программного обеспечения как иерархической структуры модулей последовательно сверху вниз с расширением решаемых функций.

Этапы структурного программирования:

1. Общие задачи, решаемые программным обеспечением системы, делятся на приоритезированную последовательность подзадач, более малых, понятных, конкретных, управляемых и обеспечивающих заложенную тестируемость.
2. Функции подвергаются декомпозиции в иерархическую древовидную структуру программных компонент.
3. Код программы разрабатывается одновременно с разработкой программного обеспечения и его тестированием.

9.2 Объектно-ориентированное программирование

С развитием методологии программирования, выразившейся в появлении языка С и ,сначала, как его расширения, а, затем, как самостоятельного языка С++ перед разработчиком программного обеспечения появилась поддерживаемая компилятором возможность объединения структур и типов данных, характеризующих конкретный физический объект или процесс с методами их обработки в единую абстракцию под названием “объект”. Объект представляет собой совокупность типов и структур данных и методов работы с ними.

Объектно-ориентированный подход к разработке программного обеспечения включает в себя несколько фаз проектирования:

1) Объектно-ориентированный анализ – анализ алгоритма работы системы обработки данных, управления и контроля для выявления закономерностей представления и изменения типов используемых данных и функций их модификации с целью формализации типовых структур данных и базовых операций над ними.
2) Объектно-ориентированное конструирование – синтез программного алгоритма на основе представления объектов и методов их преобразований для достижения конечного результата.
3) Объектно-ориентированное кодирование – кодирование (написание программ) разработанного алгоритма с использованием объектно-ориентированного языка программирования – С++, ObjectC, Java и т.п.

Массовое применение объектно-ориентированного подхода к программированию в последнее время вызвано тем, что он хорошо поддается автоматизации. Для конкретных специализации программного обеспечения созданы АРМы, реализующие CASE технологию для решения первых двух фаз проектирования, и автоматизирующих кодогенерацию по разработанному алгоритму (к примеру пакеты моделирования Hyperception, LabView фирмы National Instruments, исследовательский проект гетерогенного распределенного

прототипирования Ptolemy университета Беркли). Вместе с этим, существующие характеристики производительности вычислительных устройств не позволяют повсеместно применять этот подход из-за необходимости ,в некоторых случаях, обеспечения работы в реальном времени.

10 Общий алгоритм функционирования

Основой общего алгоритма функционирования является набор задач, решаемых системой контроля и управления. Конкретизация отдельных задач позволяет рассматривать общий алгоритм функционирования как совокупность частных алгоритмов, выполняемых в определенной временной последовательности и приоритезированной функциональной взаимосвязи, реализующих единую задачу надежной обработки информации с требуемой точностью и заданной дискретностью выработки сигналов контроля и управления.

Общий алгоритм может быть представлен структурной схемой рис.

- Алгоритм комплексной обработки входных данных – оптимизация обработки входных данных на основе устранения их избыточности для повышения точности и достоверности оценки входных параметров. Реализуется как время зависимая и функционально зависимая обработка.
- Алгоритмы задач управления – набор алгоритмов задач управления, определяющийся задачами, решаемыми системой и отражающий состав датчиков входных данных, интерфейсов управления и контроля. Эти алгоритмы определяют основные характеристики системы в целом.
- Алгоритмы защиты функциональных алгоритмов – состоят из алгоритмов обнаружения ошибок и алгоритмов анализа и исправления ошибок.
- Алгоритмы диспетчеризации и прерывания вычислений – управление распределением вычислительных ресурсов системы для одновременного или квази одновременного выполнения всех решаемых задач.
- Алгоритмы тестовой проверки и контроля работоспособности – позволяют проверить работоспособность аппаратуры путем тестирования в известных условиях.
- Алгоритмы имитации исходных данных и выполняемых задач – тренировка системы и обслуживающего персонала.
- Алгоритмы выдачи информации на исполнительные устройства и устройства отображения – задачи ввода команд управления, вывода команд на исполнительные устройства, визуализация, документирование и т.п.

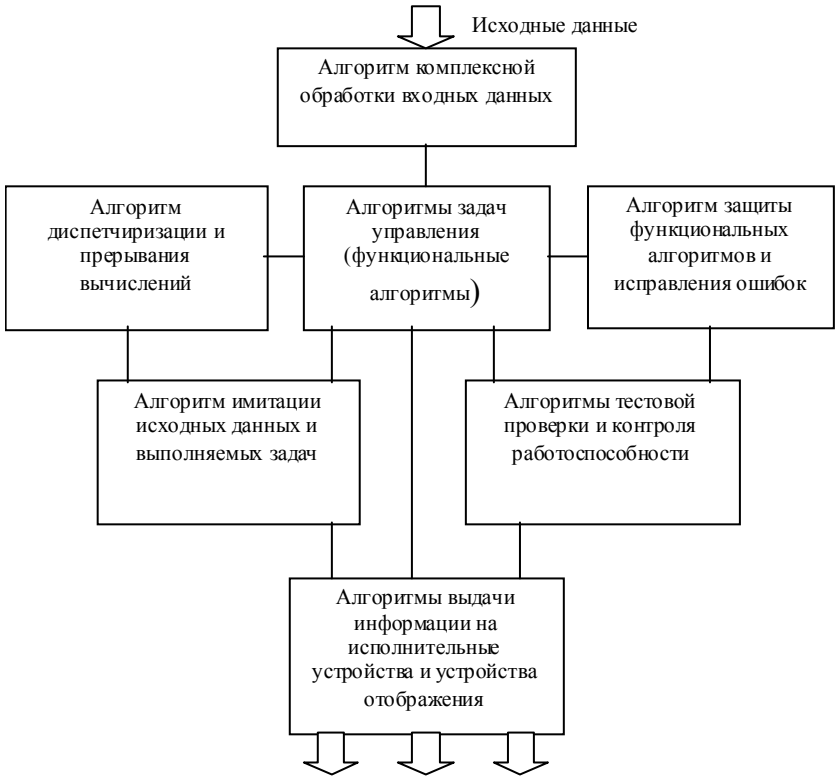


Рис. 23

В зависимости от конкретного применения удельная доля отдельных частных алгоритмов в составе общего алгоритма функционирования может изменяться вплоть до полного их исчезновения. В частности, в простейших системах управления и контроля могут отсутствовать алгоритмы диспетчеризации вычислений, а в высоконадежных системах управления летательными аппаратами объем алгоритмов защиты функциональных алгоритмов и исправления ошибок может превышать объем самих функциональных алгоритмов.

11 Процесс выбора архитектуры автоматизированной системы управления и контроля

Архитектура системы обработки, контроля и управления выбирается путем анализа возможных архитектурных решений с учетом требований к конкретной системе.

На основании проблемного алгоритма, конкретизированного в виде общего алгоритма функционирования, необходимо выбрать из доступных архитектур несколько наиболее подходящих. Каждая из отобранных архитектур используется в качестве шаблона для перенесения проблемного алгоритма на набор аппаратных и программных модулей, ей соответствующих, используемый как строительные блоки для данной архитектуры. Затем эти решения моделируются, определяются и сравниваются показатели производительности и эффективности для отбора наиболее удачного решения (Рис.). Сам процесс выбора архитектуры является разработкой и моделированием новых архитектурных решений, обновляющий библиотеки типов архитектур и модулей.

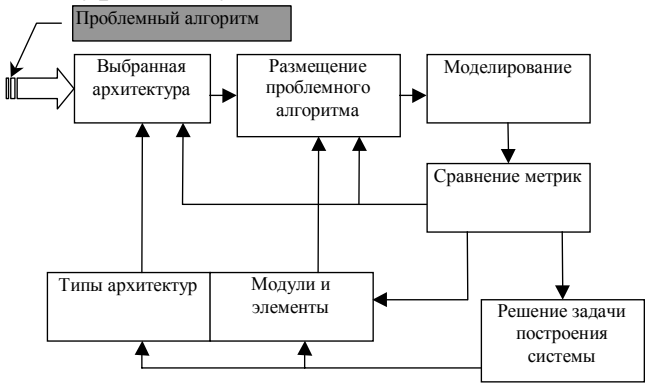


Рис. 24

Решение задачи пригодности анализируемой архитектуры может быть принято на основании:

1) соответствия спецификации системы (техническому заданию) – удовлетворение каким-то качественным требованиям к системе по производительности, масштабируемости, живучести и т.п.;
2) тактико-технических показателей – эффективность выбранного варианта построения системы на основании технико-экономического анализа.
3) Тактико-технические показатели (метрики)
4) Функциональные показатели – метрики, касающиеся количественного выражения функциональных качеств системы, определяющих решаемые ею задачи (точности, пропускные способности, время реакции системы, и т.д.).
5) Показатели совместимости с окружающей средой – температурный диапазон работы, влажность, радиация, ЭМС, вибрации, и т.д.
6) Интерфейсы – внешние интерфейсы сопряжения с окружающими системами, интерфейсы с пользователем, стандартность этих интерфейсов, используемые внутренние аппаратные и программные интерфейсы.
7) Показатели по обеспечению информационной безопасности – требования по уровню секретности, средства идентификации пользователей,

администрирование, криптографическая защита, время восстановления при взломе, протоколирование работы, fool tolerance.

8) Планово-календарные показатели – время разработки системы, время на организацию выпуска продукции, время на моделирование системы.
9) Ценовые показатели – себестоимость разработки, себестоимость опытного образца, себестоимость аппаратуры, себестоимость разработки программного обеспечения, доступность готовых коммерчески доступных модулей и элементов.
10) Показатели вычислительной производительности – количество операций в MFLOPS, количество процессоров в системе обработки, их тип, тактовые частоты, разрядность и т.д.
11) Характеристики коммуникационных элементов – скорости обмена данными, пиковая и средняя пропускные способности, типы соединений, протоколы коммуникаций, физические носители и т.п.
12) Характеристики программного обеспечения – сложность (объем) программного обеспечения (в строках исходного кода программ), языки программирования, модель программирования, стандарты разработки, используемые операционные системы.
13) Габаритные показатели – размеры, стандартные конструктивы, наличие шасси и т.д.
14) Весовые показатели – аппаратуры, дополнительного оборудования (блоки питания, система кондиционирования, арматура крепления).
15) Потребляемая мощность – ток, напряжение питания, требования по бесперебойности питания.
16) Показатели надежности – среднее время наработки на отказ, коэффициент резервирования аппаратуры.
17) Показатели тестируемости – внутренние и внешние системы тестирования, коэффициент тестируемости аппаратуры.
18) Показатели настраиваемости - изменение и/или восстановление состава системы и/или отдельных ее параметров.
19) Показатели отказоустойчивости – способность обнаруживать сбои, ошибки, отказы аппаратуры и программного обеспечения и сохранять работоспособность по мере выхода части системы из строя.
20) Показатели масштабируемости – возможность расширения состава системы и ее функций.
21) Показатели стандартизации – выполнение системы в соответствии с определенными стандартами.

Все эти метрики взаимно связаны, поэтому возможно присвоение каждой из метрик удельного веса, определяющего ее вклад в совокупный показатель качества системы (путем, допустим, применения метода экспертных оценок). Сравнивая совокупные показатели качества анализируемых систем возможно принять решение о эффективности выбора какого либо варианта.

12 Протоколы доступа к физическим каналам связи

В связи с тем, что элементы систем обработки данных часто распределены в пространстве, совокупные показатели качества таких систем во многом определяются примененными коммуникационными решениями. Основной характеристикой коммуникаций является протокол доступа к физическому каналу связи – проводу, радиоканалу, оптической линии. Именно выбранный протокол доступа определяет потребительские характеристики коммуникационной подсистемы – удельная стоимость на один адаптер, пропускная способность, время реакции интерфейса, надежность, масштабируемость, отказоустойчивость соединения.

12.1 Показатели эффективности протокола

В приложении к промышленным системам управления и контроля отдельное внимание должно уделяться показателям эффективности протокола.

Удельная цена на адаптер

Минимизация этого параметра при удовлетворении требований применения и является критерием выбора коммуникационного протокола.

Эффективность использования канала связи

Под эффективностью использования канала связи понимается соотношение объема передаваемых данных к пропускной способности канала связи (в некоторых протоколах высокого уровня для передачи 1 байта информации требуется передача еще 31 байта служебной информации). Эффективность протокола определяется размерами служебной информации в передаваемых сообщениях, и уменьшение заголовков сообщений улучшает использование канала связи.

Время доступа к каналу связи

Эффективность применяемого протокола также связана со временем, необходимым для выполнения служебных манипуляций перед началом передачи. Это время обычно определяется необходимостью арбитража доступа к каналу связи.

Детерминированность протокола

Для систем, осуществляющих управление или контроль в режиме реального времени необходимо знать максимальное время на передачу сообщения при любых возможных условиях работы.

Поддержка приоритетов узлов и сообщений

Возможность введения приоритезации в очередность передачи сообщений является важной составляющей конечной эффективности протокола.

Протокол прямого соединения

Исторически именно эти протоколы использовались для соединения между собой компьютеров класса майнфрейм (через модемы). Этот протокол поддерживает только два узла для одного физического канала связи, хотя с помощью него возможно объединение нескольких узлов с использованием нескольких каналов связи (Рис.). Для двух напрямую соединенных узлов данный протокол обеспечивает хорошую временную детерминированность, для более обширных систем характеристики значительно ухудшаются. Удельная цена узла невелика (но зависит от физического канала связи – модем, оптоволокно). Используется в соответствии со стандартом X.25 в услугах по предоставлению доступа к общественным сетям телефонными компаниями и в качестве основного протокола в System Network Architecture (SNA) IBM.

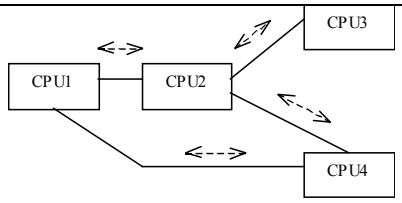


Рис. 25

12.2 Протоколы опроса.

Эта группа протоколов имеет большую распространенность в системах сбора информации с распределенных систем датчиков в связи с простотой и детерминированностью (Рис).

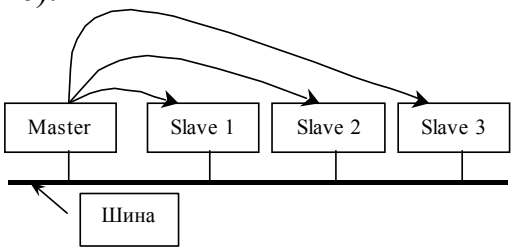


Рис. 26

Для реализации этого протокола используется разделяемая шина или канал связи. На этой шине имеется главное устройство – Master – производящее выдачу команд на подчиненные устройства и управляющее обменом по линии связи. В данный протокол легко вводятся приоритеты в обмен сообщениями, обеспечивается хорошая временная детерминированность. Вместе с этим, достаточно невелика эффективность использования канала связи из-за большой удельной доли служебных передач команд и ответов. Отказ контроллера шины или обрыв линии связи приводит к выходу из строя всей системы. Данный протокол стандартизован военным стандартом для авиационных межсистемных коммуникаций (MIL-STD-1553B). Существуют модификации этого протокола с несколькими master-узлами – Profibus.

Time Division Multiple Access – TDMA (Множественный доступ с разделением времени)

Протоколы этой группы интенсивно используются в спутниковых телекоммуникационных системах. Топология объединения узлов в сеть похожа на предыдущую, но вместо подачи команд каждому устройству используется подача одного сигнала кадровой синхронизации (Sync) всем узлам (Рис.).

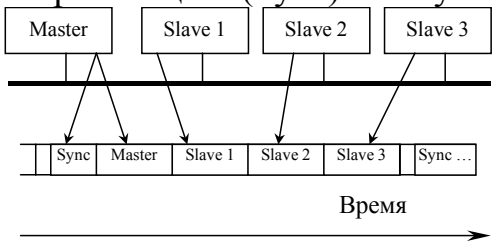


Рис. 27

По этому сигналу каждый из узлов производит установку своего таймера, и через определенное время осуществляет передачу данных. Отличие данного протокола от предыдущего заключается в большей его эффективности при сильной загрузке канала связи и в большей удельной цене узлового адаптера, т.к. требуется высокостабильный таймер. Другим недостатком можно считать необходимость

использования сообщений фиксированной длины. Используется данный протокол в аэрокосмической промышленности (NASA, Boeing).

Token Ring (Кольцо с передачей маркера)

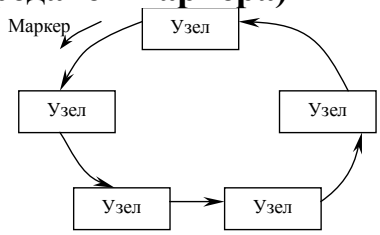


Рис. 28

Все узлы при использовании протоколов этой группы объединяются в однонаправленное кольцо с использованием прямых двухточечных соединений. По этому кольцу передается сообщение или сигнал специального вида, называемый маркером. Получение узлом маркера дает ему право осуществить передачу своего сообщения, вслед за чем маркер передается следующему узлу. В отсутствии маркера узел транслирует через себя данные с предыдущего узла (Рис). При известном количестве узлов в кольце N время максимальной задержки передачи сообщения составляет N*T, где T – время передачи от одного узла к другому. В данном протоколе возможно установление приоритетов в работу узлов путем введения в состав маркера поля, содержащего уровень текущего приоритета сообщений. В случае, если приоритет сообщения ниже этой величины, то оно не передается либо до возрастания его уровня, либо понижения приоритета в маркере. Данный тип протоколов хорошо подходит для применения оптоволоконных линий связи, поскольку использует двухточечные соединения. Для повышения надежности таких коммуникационных систем используют двойные кольца и специальные контроллеры, позволяющие переключать соединения для восстановления работоспособности после отказа одного из линков (SAFENET).

Данный протокол очень эффективен при сильной загрузке коммуникационной системы.

Token Bus (Шина с передачей маркера)

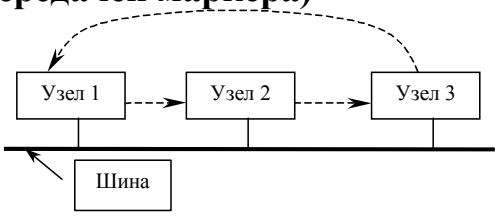


Рис. 29

Логически этот протокол однотипен с предыдущим, однако в качестве физического канала связи используется разделяемая шина, на которой организуется передача права доступа для передачи сообщения в виде маркера (Рис). Так как в протоколе Token Ring передача ведется одновременно всеми узлами, а в Token Bus только одним узлом, то максимальное время на передачу сообщения составляет для этого протокола ~N^2*T. Достоинством этого протокола можно считать временную детерминированность и более высокую надежность, т.к. выход из строя одного узла не приводит к отказу системы.

Binary countdown (Протокол “двоичного отсчета”)

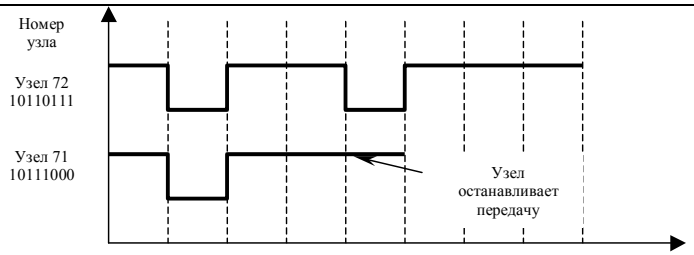


Рис. 30

Данный протокол использует в качестве физического канала связи разделяемую шину с детерминированным пассивным состоянием (логические “0” или ”1”). Передача инициируется узлами в момент освобождения канала. Арбитраж доступа осуществляется в соответствии с номером узлов автоматически (Рис.).

Пусть линия передачи является проводом с принудительно установленным уровнем логической “1”, а в качестве передатчиков используются элементы с открытым коллектором, тип логики - отрицательный. Тем самым выдача каким то из узлов лог. “0” приведет к установлению уровня логического нуля. Передача начинается с выдачи в линию (с инверсией) номера узла, ведущего передачу. Попутно этот узел отслеживает состояние линии связи. Как только наблюдается расхождение в передаваемом коде и наблюдаемом в линии, узел “выбывает из соревнования”. Данный протокол имеет заложенные в него свойства арбитража доступа, и поэтому обладает хорошими показателями по пропускной способности и детерминированности. Используется данный протокол в системах автоматизации автомобилестроения.

Carrier Sense Multiple Access with Collision Detection (CSMA/CD) (Протокол множественного доступа с проверкой наличия несущей и обнаружением коллизий)

Данные протоколы получили широкое распространение в локальных вычислительных сетях – Ethernet. В качестве линии связи используется разделяемая шина. Передача ведется асинхронно. Наличие передачи проверяется путем обнаружения несущей частоты в линии. В случае, если линия свободна, узел может начинать передачу данных, контролируя при этом сигнал на самой линии связи. При обнаружении искажений в передаваемом сигнале, контроллер принимает решение о коллизии – столкновении передаваемого пакета с пакетом с другого узла, и производит “откат” на случайно выбираемое время, после чего повторяет попытку передачи.

С одной стороны, принципы, заложенные в протоколе позволяют сочетать низкую удельную стоимость с хорошими показателями производительности при малой загрузке, однако данный протокол абсолютно не подходит к системам, работающим в реальном времени, т.к. его называют “протоколом негарантированной доставки сообщений”. Кроме этого, при повышении загрузки канала использование канала связи резко ухудшается в связи с множественными коллизиями. Системы, использующие этот протокол, хорошо масштабируются.

Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) (Протокол множественного доступа с проверкой наличия несущей и устранением коллизий)

Данный протокол является комбинацией протоколов CSMA/CD и TDMA. В состав системы вводится один или несколько контроллеров сети, которые отслеживают сетевой трафик. При повышении загрузки (обнаружении одной или нескольких коллизий) подается специальный сигнал оповещения, переводящий все

подчиненные контроллеры в режим TDMA. Этим достигаются хорошие показатели эффективности при сильной загрузке сети.

12.3 Выбор протокола доступа к каналу связи

Как видно из предыдущих пунктов, все протоколы имеют свои достоинства и недостатки, поэтому применение конкретного протокола определяется спецификой системы.

Выводы:

1) Протоколы опроса, TDMA и прямого соединения просты, но не всегда обеспечивают достаточную гибкость для сложных систем.
2) Протоколы, базирующиеся на передаче маркера, хорошо детерминированы, но могут ухудшать использование канала связи и требовать сложного программного обеспечения для улучшения своих параметров.
3) Протоколы Binary Count-Down сильно зависят в своей работе от параметров физического канала связи.
4) Протоколы CSMA/CD неприменимы для систем реального времени, работающих с сильной загрузкой канала связи.

Сравнение коммуникационных протоколов доступа к физическому каналу связи

Табл. 8

+ хорошо * нормально - плохо	Эффективность при малой загрузке	Эффективность при сильной загрузке	Детермини-рованность	Приоритет	Низкая удельная цена на узел
Прямое соединение	*	-	+	*	*
Опрос	-	*	+	-	*
TDMA	-	+	+	-	-
Token Ring	+	+	+	+	*
Token Bus	*	+	+	*	*
Binary Count-Down	+	+	*	*	+
CSMA/CD	+	-	-	*	*
CSMA/CA	+	+	+	+	+

13 Организация ввода/вывода в цифровых АСУТП

Само понятие системы управления и контроля предполагает ввод данных с датчиков, подключаемых к физическому объекту и вывод управляющих воздействий или сигналов контроля. Поэтому подсистема ввода/вывода является важной неотъемлемой частью систем обработки данных, управления и контроля.

Принципы организации ввода/вывода оказывают влияние на всю архитектуру системы, определяют особенности реализации алгоритмов обработки данных и/или алгоритмов управления.

Определения:

- Порт – минимальный адресуемый элемент ввода/вывода (логический или физический). Портом также может считаться регистр внутри процессора, ОЗУ и т.д. Операции ввода/вывода в данном случае требуют адресации конкретного порта.
- Канал ввода/вывода – совокупность аппаратно-программных средств, позволяющих осуществлять именованный (или нумерованный) ввод/вывод на высоком уровне общения с аппаратурой. При работе с каналом ввода-вывода для пользователя отпадает необходимость взаимодействия с конкретной аппаратурой, вместо этого он взаимодействует с устройством N XXX (READ, WRITE, CONTROL) на логическом уровне.
- Устройство ввода/вывода – физическое, логическое, или виртуальное устройство, на которое возможен ввод или вывод данных. Для пользователя устройство представляет собой абстрактную модель конкретной физической или логической структуры (Примером устройства ввода-вывода может являться физическое устройство – АЦП, ЦАП, дисковод, или логическое устройство – логический диск, сокет протокола TCP/IP)

Физический ввод/вывод возможен на два основных типа физических устройств:

1) Устройства памяти:

a) процессорная память (регистры, внутри кристальная память);

b) основная память (ОЗУ, ПЗУ, флеш-память);

c) вспомогательная память (дисковая, виртуальная, т.е., та, для доступа к которой необходимы более сложные манипуляции).

2) Периферийные устройства – дополнительное оборудование, предназначенное для сопряжения с другими системами, хранения данных, индикации, взаимодействия с пользователем.

Различные физические устройства обладают различными характеристиками быстродействия, поэтому требуют различных принципов работы с ними (различие в скорости ввода/вывода составляют до 10^9 раз).

Для осуществления корректного обмена данными с каким-либо устройством в общем случае необходимо выполнить 3 операции:

1. Проверка готовности устройства к обмену данными.
2. Управление обменом с внешним устройством.
3. Чтение или запись данных.

Различные способы реализации вышеперечисленных операций приводят к появлению разнообразных видов ввода/вывода.

13.1 Разновидности ввода/вывода

Прямой ввод-вывод.

Все 3 операции обеспечения ввода-вывода выполняются непосредственно процессором (см. рис).

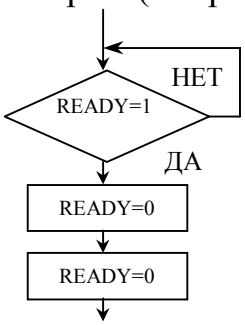


Рис. 31

Процессор ожидает данных в выходном буфере устройства, и осуществляет чтение.

Недостатки:

1. Возможно зацикливание программы в цикле ожидания готовности
2. Процессор при ожидании готовности не производит вычислений
3. Программные секции ввода-вывода должны быть распределены по программе обработки

Косвенный ввод-вывод

Косвенный ввод-вывод отличается от прямого применением дополнительных аппаратно-программных ресурсов, позволяющих разгрузить процессор от выполнения каких-либо групп операций обеспечения ввода-вывода.

Косвенный ввод – вывод достигается применением таких механизмов, как ввод-вывод по прерыванию, ПДП, доступ к данным через разделяемую память, интеллектуальные контроллеры ввода-вывода; на уровне программного обеспечения – флаги готовности данных, сопрограммы ввода-вывода, драйверы устройств.

13.3 Буферизация ввода-вывода

Опрос канала

Цель применения данного метода заключается в стремлении логически разделить операции обработки данных и операции ввода-вывода.

Функция чтения:

```
Get(BufferNumber):
{
Check(READY); Read(Channel, BufferNumber);
BufferNumber={if( BufferNumber=1) then 0 else 1}
}
```

Функция Get осуществляет ввод данных в один из двух буферов данных, возвращая номер заполненного буфера. Второй буфер при этом может находиться в обработке.

Алгоритм ввода-вывода-обработки:

```
INIT:
{
Read(Channel, 0); BuferNumber=1;
}
LOOP:
{
Get(BufferNumber); Compute(BufferNumber); GoTo LOOP;
```

}

При неравномерной требуемой производительности обработки в данном алгоритме возможно ожидание готовности данных. Тем не менее, данные вводятся и обрабатываются в разные области памяти, поэтому операции ввода логически относительно независимы от операций обработки.

13.4 Опережающий ввод-вывод (составные буфера)

Для преодоления недостатков предыдущего метода (блокировка вычислений готовностью данных) применяют алгоритм опережающего ввода-вывода, связанный с большей глубиной буферизации данных, что позволяет В СРЕДНЕМ снизить зависимость между готовностью данных и их обработкой.

Одним из вариантов применения составных буферов является схема кольцевого буфера (см. рис.)

В памяти организуется группа из N буферов, объединенных логически в однонаправленное кольцо (Buf(i), i={0:N-1}, за Buf(i) следует Buf(MOD_N(i+1))). Указатель NextIO определяет буфер, содержащий введенные данные (флаг R - READY); указатель NextGet – на пустой буфер, данные из которого уже обработаны. Возможность ввода-вывода и обработки определяется условиями:

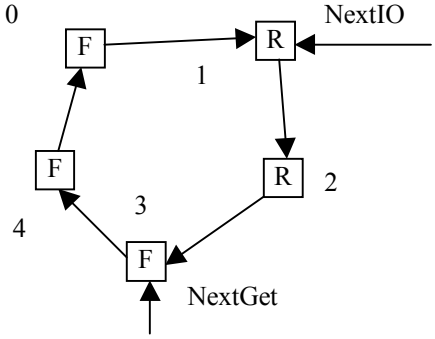


Рис. 32

$(NextGet+1)_{MOD_N}$!= $(NextIO)_{MOD_N}$;

$(NextIO+1)_{MOD_N}$!= $(NextGet)_{MOD_N}$;

т.е. ввод-вывод не может осуществляться с занятыми буферами, а обработка не может опережать ввод.

Фрагмент алгоритма ввода-вывода-обработки:

{

Get(NextGet); NextGet = $(NextGet+1)_{MOD_N}$;

Compute(NextIO);

Release(NextIO);(сбросить флаг R и установить флаг Free)

NextIO=$(NextIO+1)_{MOD_N}$;

}

Операции инкремента указателей должны сопровождаться проверкой условий (см. выше). Глубина буферизации данных должна соответствовать средней и пиковой производительности вычислителя.

Данный подход имеет лучшие характеристики, чем метод опроса канала, но в нем имеются принципиальные трудности обеспечения приоритезации ввода-вывода.

13.5 Ввод-вывод по прерыванию

При вводе выводе по прерываниям автоматически достигается опережающий ввод-вывод. Применение этого метода требует наличия дополнительной аппаратуры – контроллера прерываний, обрабатывающего сигналы готовности от

периферийных устройств, а также специальных подпрограмм обработки прерываний или драйверов устройств. Механизм обработки прерываний изложен в курсе ЦУМ и МП, поэтому рассмотрим только проблемы его применения.

1. Обработка прерывания процессором приводит к остановке программы, которая в данный момент выполняется, требует сохранения ее контекста (регистры процессора, стек), что существенно понижает его быстродействие (особенно для современных процессоров с кэшированием данных и команд). В связи с этими соображениями ожидаемая частота прерываний должна соотноситься с временем переключения контекста процессора, и часто приводит к необходимости аппаратной буферизации устройства ввода-вывода для уменьшения интенсивности прерываний.

2. Ввод-вывод по прерыванию и обработка производиться разными программами (более корректно для многозадачной вычислительной системы – процессами), поэтому необходимо обеспечить безошибочную обработку счетчиков введенных и обработанных буферов. Данная проблема имеет название проблемы “критической секции” и может быть проиллюстрирована на примере:

Покупка билетов на ЖД через автоматизированную систему:

1) Вы запрашиваете в кассе билет в …
2) Кассир запрашивает билеты в АС
3) АС предлагает несколько билетов на выбор
4) В это-же время в другой кассе происходит то же самое
5) Вы покупаете билет, при этом количество свободных билетов уменьшается
6) В другой кассе также происходит покупка билета, но число свободных билетов для декремента берется то же, что и для вас
7) Результат: куплено два билета, а количество свободных билетов уменьшается на 1, а не на два.

Общими ресурсами, корректность доступа к которым необходимо обеспечить для двух процессов, совместно их использующих, являются в данном случае счетчик свободных билетов, а операции по его считыванию и модификации являются критическими секциями процедур покупки билетов.

Табл. 9

Программа обработки	Примечание	Обработка прерывания	Примечание
…….	Обработка буфера	Ввод данных	Заполнение буфера
R ← - считывание количества свободных буферов	Начало критической секции, R-количество свободных буферов	R ←	Считывание количества свободных буферов
R=R+1	Освобождение буфера для ввода	R=R-1	Декремент количества свободных буферов
R → -запись нового значения свободных буферов	Конец критической секции	R→	Запись R, конец критической секции
……		Выход из обработчика	

		прерываний	

В случае, если данные секции программы обработки данных и их ввода будут выполняться в одно и то же время, количество свободных буферов будет неуклонно уменьшаться.

Способ решения данной проблемы – использование флагов синхронизации:

Табл. 10

Программа обработки	Обработка прерываний
B – флаг, означающий критическую секцию основной программы; I - флаг, означающий критическую секцию обработчика прерываний; I, B – глобальные переменные, доступные обеим процедурам	
…….	……..
B=1	If (B) then I=1
R=R+1	Else
B=0	R=R-1
If (I) R=R-1	……..
I=0	
…….	

Тем самым, возможно в некоторых случаях устранить проблему критической секции при вводе-выводе по прерываниям с использованием флагов.

13.6 Объединение буферов в пул ввода-вывода

Рассмотренные выше методы предполагают различные буфера данных для устройств ввода и для устройств вывода, что приводит В СРЕДНЕМ к неэффективному использованию памяти. Для повышения степени использования памяти и централизации управления вводом-выводом (особенно это важно при построении систем обработки данных в реальном времени, где временная детерминированность работы может быть достигнута централизацией планирования обработки и управления вводом-выводом) используется объединение буферов в единый буферный пул.

Буфер в отличие от предыдущих случаев может находиться в одном из трех состояний:

Empty

Input_Full

OutputFull

Операции работы с буферами:

GetBufIn, ReleaseBufIn,

GetBufOut, ReleaseBufOut.

Одним из критических моментов огранизации буферных пулов является устранение блокировок ввода вывода при количестве свободных буферов равным 0, т.к. операция получения свободного буфера должна задерживаться до ее успешного завершения, и, тем самым, блокирует процедуры обработки занятых буферов, которые должны его освобождать.

14 Общие понятия и задачи системного программного обеспечения

Сложные системы обработки и управления состоят из множества совместно работающих вычислительных устройств. Для синхронизации вычислений используется в той или иной форме системное программное обеспечение, главной задачей которого является создание логически правильной “виртуальной машины” пользователя, т.е. расширение возможностей физически доступных аппаратных средств с помощью функций, необходимых и более удобных для использования прикладными программами пользователя.

Данное системное программное обеспечение реализуется в форме универсальных операционных систем, специализированных операционных оболочек (расширений ОС), и интегрированных сред (комплексов различных ОС и оболочек). На различных уровнях иерархии системного программного обеспечения могут решаться различные задачи: взаимодействие с аппаратурой на физическом уровне, управление памятью, разделение вычислительных ресурсов между процессами, синхронизация процессов, механизмы защиты программ выполнения и авторизация доступа, синхронизация распределенных вычислений и т.д.

В подобных системах одной из базовых единиц выполняющихся вычислений является понятие процесса – выполняемый код программы, загруженной в память и доступное ему пространство памяти под данные.

Процессы

Процессы могут быть параллельными (независимыми друг от друга) и последовательными, когда один не может начаться раньше, чем не кончится другой.

В случае, если процессы выполняются одновременно на многопроцессорном вычислителе, то говорят, что достигнут аппаратный параллелизм, если они МОГУТ выполняться параллельно – то логический.

Большинство формул операций над матрицами обладают логическим параллелизмом, поэтому вычисления для них в многопроцессорных системах могут быть распараллелены. Логическим параллелизмом обладают также и многие алгоритмы обработки изображений.

Процессы находятся в одном из трех состояний:

- Блокирован – если для выполнения процесса нет данных для обработки;
- Активен – если он выполняется процессором;
- Готов – если данные для обработки есть, а процессор занят другим процессом.

Центральной задачей системного программного обеспечения сложных систем обработки и управления является управление процессами – создание, удаление, обеспечения взаимодействия (обмен данными), управление и планирование вычислений. Для достижения этих целей используется несколько примитивов.

Семафоры

Специального типа переменные, служащие для учета вычислительных ресурсов (количество свободных буферов, к примеру). Операции по модификации этих переменных производятся самой ОС с целью исключения критических секций пользовательских процессов, с ними взаимодействующих:

Установка семафора S:

V(S) → S=S+1

Сброс семафора S:

P(S) →

LOOP:
if (S >0) then {S=S-1}
Else
{
GoTo LOOP
} – сброс семафора блокируется, пока его значение не станет >0.

Операции V(S) и P(S) – неделимые операции, поэтому ими можно синхронизировать состязающиеся процессы более корректно, чем флагами (см. ввод-вывод по прерываниям).

Сигналы

Сигналы являются стандартными переменными, находящимися в блоках описания процессов и служат для изменения состояния процессов – перевода из блокированного состояния в активное, повторная инициализация процесса, прекращения процесса. Процессы могут воздействовать на сигналы других процессов, осуществляя воздействие на их состояние через стандартный интерфейс, не связанный с особенностями конкретного процесса

Сообщения

Сообщения посылаются от одного процесса другому с целью синхронизации времены выполнения или передачи данных на обработку. Сообщения могут иметь различные типы – без и с подтверждением приема, широковещательные, с адресом и без адреса посылателя и т.д.

Планирование

Стратегией планирования называются алгоритмы, заложенные в системное программное обеспечение, по которым принимаются решения о выборе из многих конкурирующих процессов некоторого, способного развиваться, если ему будет предоставлен процессор.

Реализующие стратегию планирования процессы составляют планировщик ОС, во многом определяющий ее пользовательские качества.

Арбитром состязания процессов – планирования – является система управления ресурсами вычислительной системы (память, процессорное время, устройства ввода-вывода)

15 Программные методы контроля

При построении АСУТП одним из важнейших показателей качества работы такой системы является надежность и достоверность. Поскольку в данном курсе рассматриваются системы автоматизированные и цифровые, то вышеуказанные показатели качества достигаются в том числе и использованием программных методов контроля правильности работы вычислительных и управляющих устройств. Для этого в программное обеспечение вводится набор дополнительных алгоритмов программно-логического и тестового контроля.

Программно-логический контроль

Использует информационную избыточность и предназначен для проверки правильности функционирования программного обеспечения.

Способ двойного просчета со сравнением результатов

Вся программа разбивается на участки, каждый из которых контролируется переменной, отражающей на данном участке результат вычислений. Повторные вычисления одних и тех же участков алгоритма с проверкой одинаковости результатов позволяют выявить одиночные сбои в вычислениях и повторить их до исчезновения ошибки

Способ контрольных соотношений

Позволяет выявить ошибки за счет случайных сбоев функционирования аппаратуры и ее отказа. Достигается введением в программу вычислений сегментов кода с вычислением очевидных выражений ($\sin^2 x + \cos^2 x = 1$). Виды арифметических операций, входящих в тривиальные вычисления должны быть те же, что и в контролируемом сегменте программы.

Способ усеченного алгоритма

Вычисления производятся по основному и упрощенному алгоритму параллельно, при этом результаты сравниваются по величине и допускам. Выявляются отказы и сбои вычислительной системы.

Способ логического анализа результатов вычислений

Выявляются случайные и систематические ошибки. Используется, когда известны законы изменения некоторых вычисляемых параметров.

Способ подстановки

По результатам вычислений решается обратная задача, затем выявляется различие в первичных и “перевычисленных” входных данных. Применим только в условиях правильности функционирования базового ПО вычислительной системы.

Контроль хода выполнения программ

Контроль правильности хода запуска программ, длительности работы программ, правильности выполнения (порядка) переходов подпрограмм и прерываний.

Применяется идеология ведущего планирования, т.е. способа организации вычислений, при котором централизуется управление вычислениями, что позволяет более легко осуществить контроль (трассировку) хода выполнения процедур.

Тестовый контроль

Предназначен для защиты алгоритмов от ошибок, локализации ошибок и их устранения.

Испытательные тесты комплексной проверки

Используются для тестирования системы при ее включении и/или периодически во время ее работы. Состоят из одной или нескольких решаемых задач, аналогичных целевым для системы с известными входными данными.

Диагностические тесты

Определяются места возникновения неисправностей, используются после проведения тестов комплексной проверки. Состоят из специально подобранных примеров, которые требуют выполнения всех операций, имеющихся в системе команд. Ориентированы на локализацию отказов и сбоев функционирования.

16 Свойства формальных моделей в АСУТП

Разрешение модели

Различные модели описания обладают своими особенностями разрешения характеристик системы по следующим категориям:

1) временным – касающихся описания функционирования системы во времени;
2) форме представления данных – касающихся детальности описания обрабатываемых данных;
3) функциональных – касающихся уровня описания функциональности системы или ее компонент;
4) структурных – отражающих взаимосвязь различных компонент в системе.

Различные модели могут обладать своим набором разрешений по указанным параметрам – рис.33

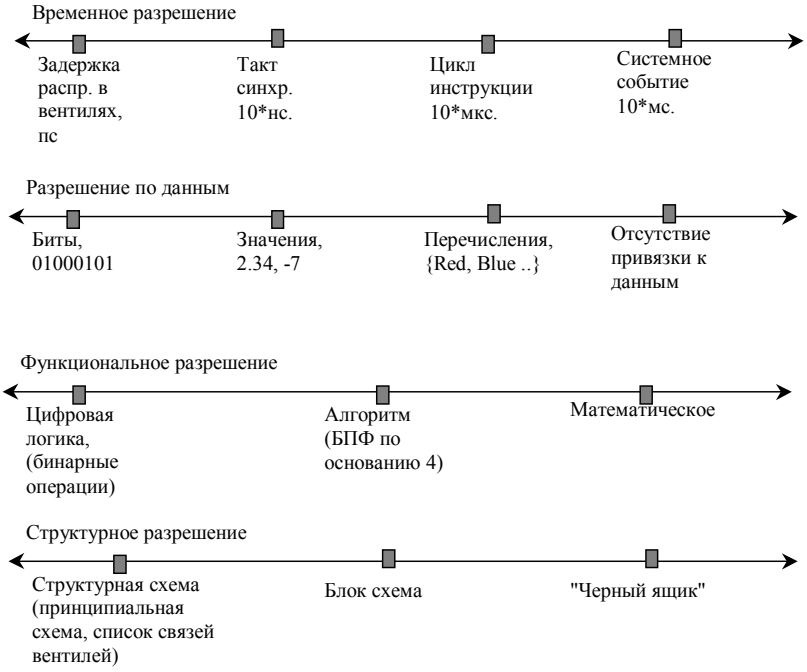


Рис. 33

Внутренние и внешние свойства модели

Модель может обладать различной описательностью для внешних и внутренних свойств представляемой системы или компонента.

Внутренним разрешением модели называется то, как она описывает временную связь событий, функций, значений и структур элементов, содержащихся в нутри рассматриваемого устройства.

Внешнее разрешение описывает детальность интерфейса устройства с внешними по отношению к нему устройствами.

Общие концепции моделирования АСУТП

АСУТП представляются общими и специализированными моделями.

Общие модели представления

Позволяют описывать систему с различным разрешением – на различных уровнях абстрагирования

Поведенческая (интерпретируемая) модель

Поведенческой моделью называют модель представления системы, отражающую функционирование системы во времени без детального описания

реализации составляющих систему элементов. Поведенческая модель может существовать для различных уровней абстрагирования в описании системы, которые определяются разрешением модели.

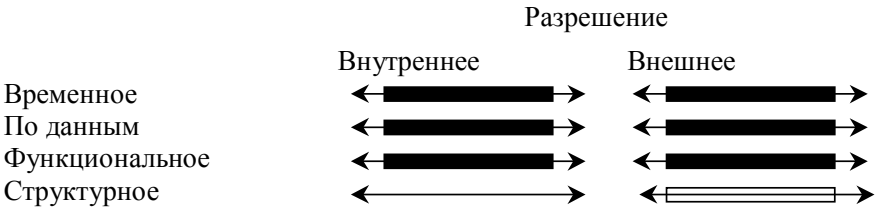


Рис. 34

Функциональная модель

Функциональная модель описывает функции отдельного элемента без спецификации конкретной реализации. Могут существовать на различных уровнях абстрагирования.

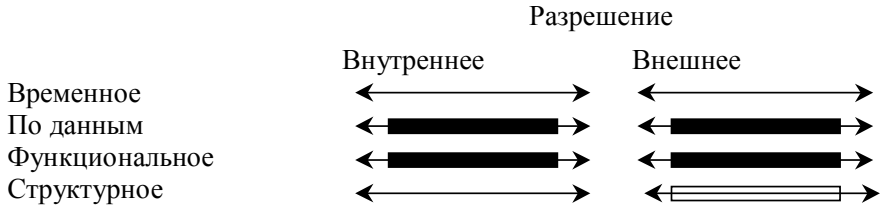


Рис. 35

Структурная модель

Структурная модель описывает иерархию конкретной физической реализации системы определением компонент системы и их топологических связей. Компоненты могут определяться структурно, функционально или на поведенческом уровне. Структурные модели могут существовать на различных уровнях абстрагирования.

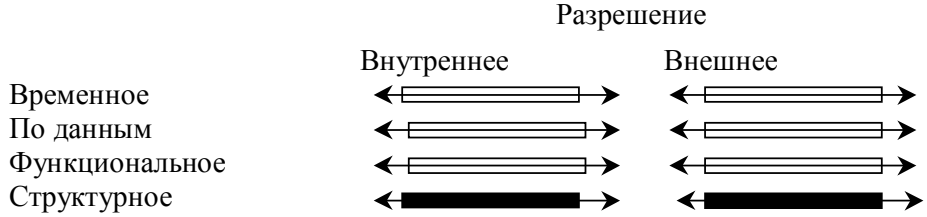


Рис. 36

Специализированные модели

Предназначены для более специальных целей описания систем и более чувствительны к уровню абстрагирования

Модель производительности

Данная модель является совокупностью метрик, отражающих временные аспекты функционирования элементов системы. Данные метрики могут вводится на различных уровнях абстрагирования.

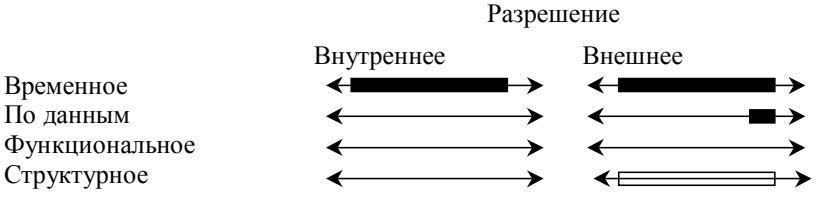


Рис. 37

Интерфейсная модель

Интерфейсная модель описывает компоненты системы по отношению к окружающим компонентам. Функциональные, структурные и временные детали интерфейса компонента вводятся для описания обмена компонента с окружением. Информация о содержании компонента отсутствует.

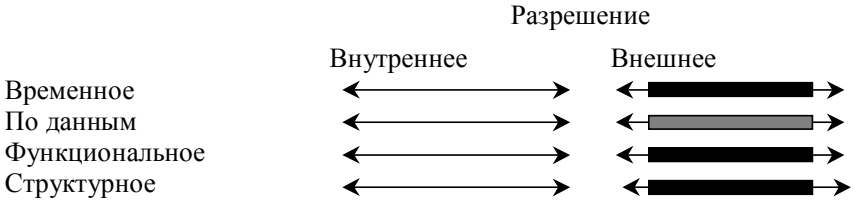


Рис. 38

Гибридная модель

Является моделью со смешанным уровнем разрешений (абстрагирования) или принципов описания.

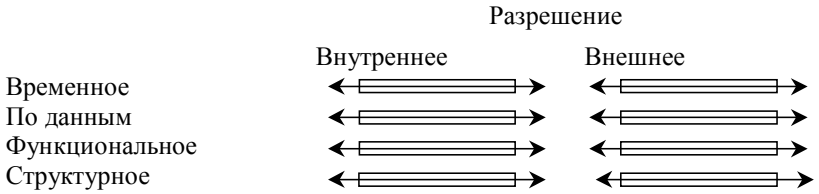


Рис. 39

Виртуальный прототип

Модель для компьютерной симуляции конечного продукта, компонента или системы. Виртуальный прототип, в отличии от физического, позволяет более абстрактно представлять и анализировать свойства конечной системы.

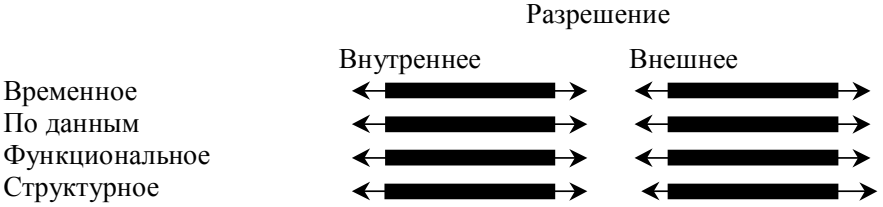


Рис. 40

Примеры



Рис. 41

Модели уровня систем

Данные модели позволяют абстрактно (без спецификации реализации) описывать цифровые электронные системы обработки, контроля и управления. Уровень абстрагирования не включает информацию о аппаратной или программной структуре реализуемой системы.

Выполняемая спецификация

Является поведенческим описанием компонент системы, отражающим функциональные и временные характеристики, наблюдаемые со стороны интерфейса элементов системы во время компьютерной симуляции. Выполняемая

спецификация может описывать электрические, поведенческие или физические аспекты системы без уточнения их конкретной реализации.

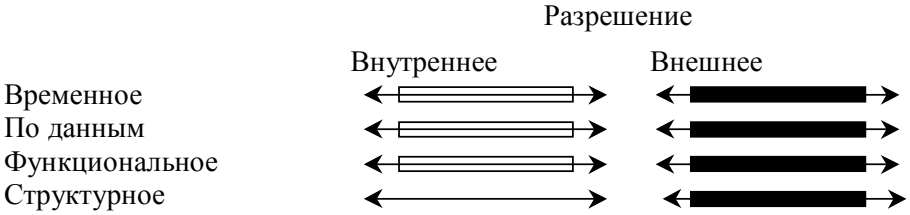


Рис. 42

Математическая модель

Описывает функциональные связи входных и выходных данных в системе через задание математических формул и соотношений без спецификации конкретной реализации необходимых вычислений.

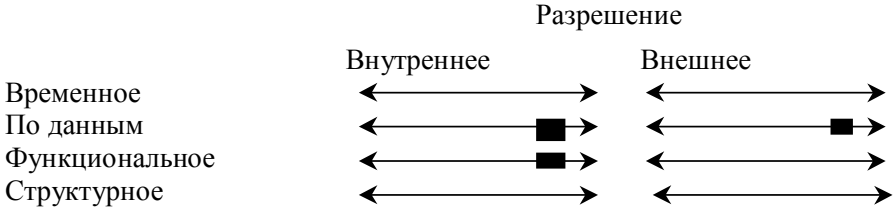


Рис. 43

Алгоритмическая модель

Алгоритмический уровень абстракции, в отличие от математического, специфицирует конкретную реализацию математического алгоритма по обработке реальных данных. Если в математической модели в качестве элемента может использоваться "Преобразование Фурье", то в алгоритмической модели – "целочисленное БПФ длиной 256 по основанию 4"

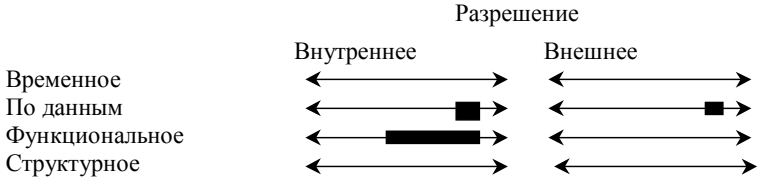


Рис. 44

Модели уровня архитектуры

Эта группа моделей предназначена для абстрактного представления аппаратуры и ПО системы.

Граф потока данных

Граф потока данных (ГПД) описывает прикладной алгоритм в отношении взаимосвязей данных его математических операций. Он является направленным графом с узлами, соответствующими математическим операциям и дугами, отражающими зависимости данных. ГПД является архитектурно независимым описанием алгоритма, что позволяет его формализацию для декомпозиции, анализа и трансформации с целью выявления возможностей его эффективной реализации.

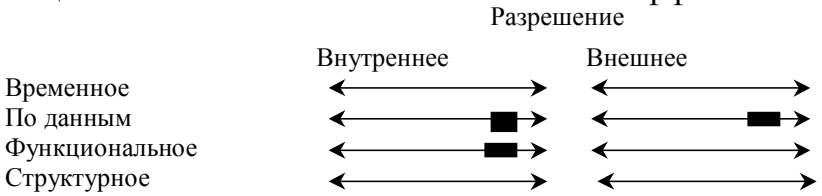


Рис. 45

Модель производительности, основанная на ярлыках

Модель производительности системы, не отражающая значений обрабатываемых данных (кроме контрольной информации). Строится преимущественно для анализа функционирования многопроцессорных вычислительных систем.

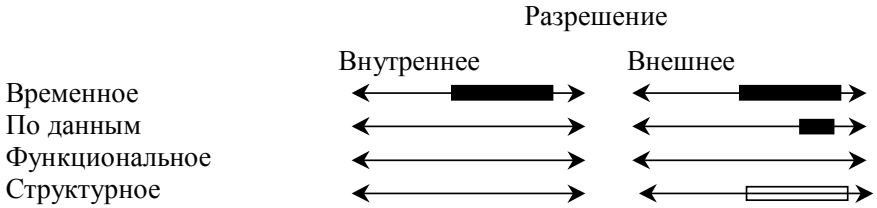


Рис. 46

Модели уровня аппаратуры

Предназначены для представления аппаратуры на различных уровнях абстрагирования.

Абстрактная поведенческая модель

Абстрактное описание компонент со стороны их интерфейса. Конкретная реализация интерфейса – разрядность, соответствие выводов, набор контрольных сигналов. Функции модели:

- установление временных и функциональных спецификаций для компонент системы с целью последующей верификации их совместного функционирования;
- численная верификация аппаратно-программной реализации на основании моделей производительности;
- возможность повторного использования моделей компонент;
- генерация тестовых векторов для более детальных моделей описания компонент и тестирования всей системы;
- представление функционирования всей системы в целом;
- документирование операций реализации системы.

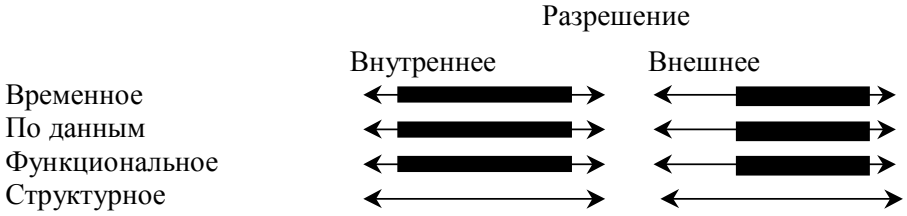


Рис. 47

Детальная поведенческая модель

Поведенческая модель компонент системы, специфицирующая их во времени и по функциональности с максимальным внешним разрешением но без описания внутренней реализации.

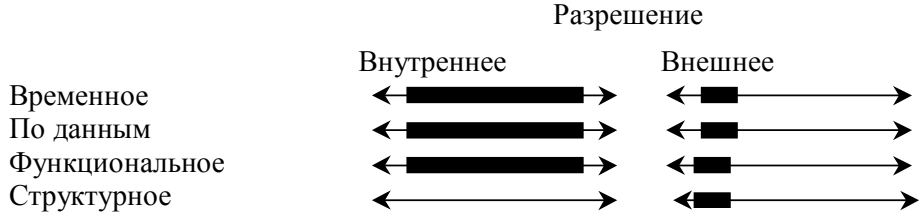


Рис. 48

Модель архитектуры набора инструкций

Представление процессора как вычислительного устройства с функциональным набором инструкций, известным набором внешних регистров и адресуемого пространства памяти. Временным разрешением модели является цикл инструкции. По данным данная модель полностью адекватна реальному устройству. В модели отсутствует структурная информация.

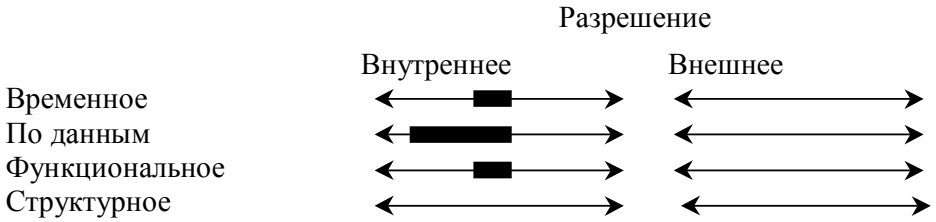


Рис. 49

Модель уровня регистровых обменов (Register Transfer Level - RTL)

Описывает систему как набор регистров и комбинационных устройств, шин, контрольных линий, реализованных обычно как машины с конечными состояниями. Соответствует уровню схемы модуля, выполненного на интегральных схемах, но не конкретных, а абстрактных, что позволяет путем (как правило) автоматизированного синтеза осуществлять планирование на конкретную реализацию.

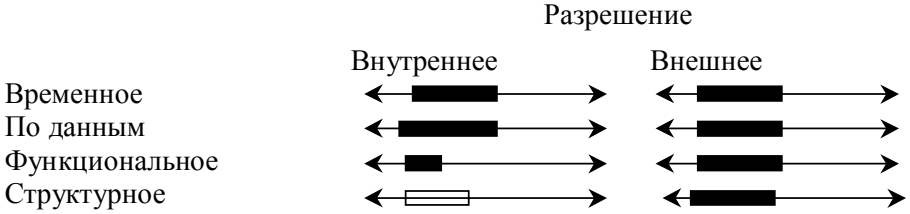


Рис. 50

Модель уровня логики

Описание поведения элементов системы как результат выполнения набора булевых функций и простейших устройств памяти – триггеров.

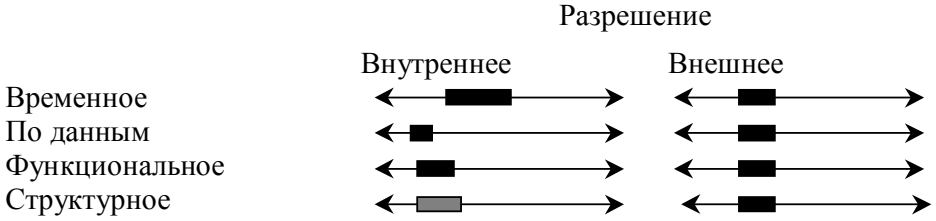


Рис. 51

Модель уровня вентилей

Описание функций, временных характеристик, структуры компонент системы на основе элементов NAND, AND, NOR, OR, XOR.

Модель уровня ключей

В качестве элемента описания используется элементарный ключ – включен-выключен.

Модель уровня цепи

Принципиальная схема цифрового устройства, представленного как аналоговая цепь с резисторами, конденсаторами и т.п.

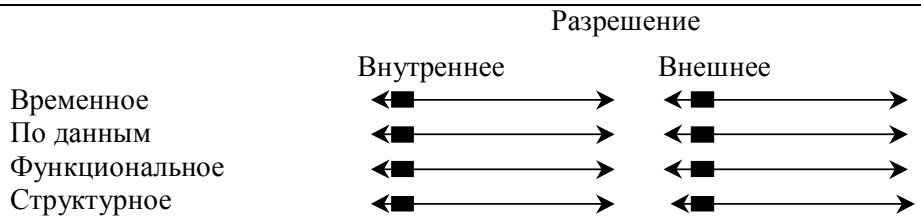


Рис. 52

Дополнительные понятия методологий моделирования АСУТП

Иерархия программного обеспечения

Примитивы задач ГПД

Прикладной алгоритм реализуется как набор задач, соответствующих подпрограммам обработки данных в графе потока данных. Как правило, ПО на этом уровне абстракции представляется в графической форме.

Псевдо код

Псевдо кодом называют упрощенный язык высокого уровня, предназначенный для более простого представления прикладного алгоритма. может являться текстовой формой представления ГПД.

Язык высокого уровня

Машинно-независимый язык программирования, соответствующий своей грамматике и синтаксису. Имеет переменные, управляющие конструкции, выражения.

Ассемблер

Мнемонический объектный код, представляемый в текстовом виде. Представляет программу как набор инструкций, выполняемых процессором.

Микрокод

Для некоторых видов процессоров, за один цикл может выполняться несколько элементарных операций. Эти операции группируются в инструкции ассемблера, а те, в свою очередь, могут быть подвергнуты декомпозиции на микроинструкции.

Объектный код

Бинарный код, который может выполняться процессором.

Уровень абстракции и иерархия

Является отображением детальности представления функции или компонента. Обратен разрешению деталей. При снижении уровня абстракции проясняется реализация деталей.

С понятием уровня абстракции связано понятие иерархии. На одном уровне абстракции система, структура, функция может быть представлена как набор компонент со своими связями, которые, в свою очередь, могут быть подвергнуты декомпозиции на более низком уровне абстрагирования.

Уровень абстракции и разрешение деталей не однозначно соответствуют точности. К примеру, абстрактная поведенческая модель детерминированного вычислительного элемента может описывать выполнение некоторой функции как 100.280 мкс, в то время как более разрешающая модель уровня регистровых обменов будет описывать эту функцию как 10028 10 нс циклов.

Иерархия представления системы, как правило, касается функциональных и структурных аспектов ее реализации. Структурное представление связано с физическим размещением компонент системы. Основной сложностью

иерархического представления систем является совмещение функционального и структурного представления.

Верификация

Процесс определения состоятельности модели, заключающийся в аналитическом или формальном доказательстве ее адекватности и соответствия предъявляемым к системе требованиям. Как правило, выполняется путем формирования входных данных, симуляции и проверке выходных данных.

Тестовый вектор

Набор значений для всех входов модели или системы и соответствующий ним набор выходных значений.

Функциональный тест

Тестирование функционирования устройства без учета его реализации, временных характеристик работы, стабильности работы.

Операционный тест

Проверка безошибочности функционирования устройства с учетом его реализации. Выявляет отказы аппаратуры, сбои и т.п.

Терминология моделирования АСУТП

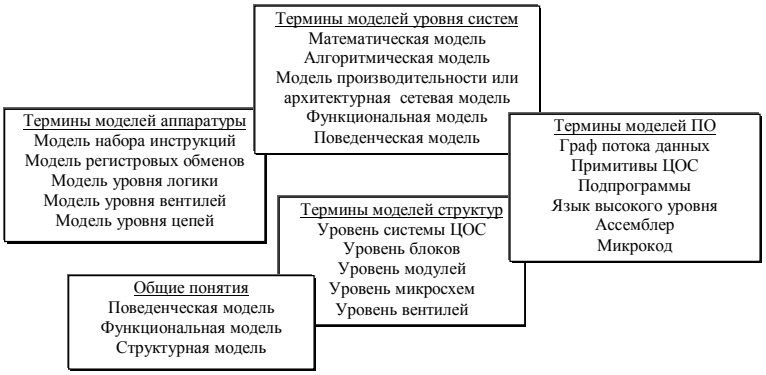


Рис. 53

Проектирование АСУТП на уровне систем

Процесс проектирования АСУТП

Процесс проектирования системы начинается с формулирования ее спецификации – требований к системе, на основании которой строится реализация (модель) верхнего уровня иерархии – абстрактная поведенческая модель, математическая модель, алгоритмическая модель, структурная и т.п. (рис. 53.).

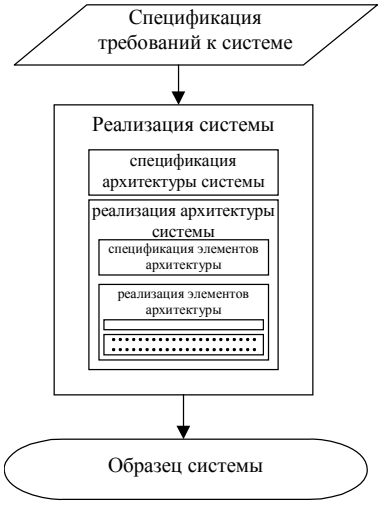


Рис. 54 Проектирование системы "сверху-вниз"

При проектировании эта модель системы подвергается последовательной декомпозиции и трансформации на составляющие с повышением детерминированности отдельных компонент с понижением уровня иерархии описания. На различных уровнях описания используются различные модели функционально-временного представления систем и их элементов и языки, их поддерживающие.

С одной стороны, выразительность языка делает его более универсальным по отношению к реализуемой модели вычислений, с другой стороны, ухудшает простоту его применения для реализации выбранной модели. Этот факт определяет множественность применяемых языков программирования для проектирования систем.

Каждая реализация системы (модель соответствующего уровня иерархии) должна подвергаться верификации с целью выявления ошибок и тестированию на предмет определения эффективности реализации. Реализаций может быть несколько, проектирование сводится к отбору наиболее эффективных решений.

Рассмотрим модели организации вычислений (МОВ), применяемые в современных средствах проектирования.

Типы вычислительных систем

Вычислительные системы исторически и по особенностям применения могут быть разделены на 3 группы:

Трансформационные системы – однократное преобразование входных данных и получение результата – программы вычислений, компиляторы и т.п. – работа в пакетном режиме.

Интерактивные системы – функционирование предполагает взаимодействие с пользователем – ОС, базы данных, Интернет.

Реактивные системы – постоянно взаимодействуют со входными воздействиями для формирования выходных сигналов. Устройства обработки сигналов, контроля и управления относятся к реактивным системам.

Реактивные системы могут в свою очередь подразделяться на системы, ориентированные на обработку данных и системы, ориентированные на контроль. Различие между ними заключается в том, что первые формируют на выходе данные, соответствующие входным, в то время как вторые – сигналы, соответствующие входным воздействиям.

Свойства реактивных систем

Основным отличием реактивных систем от интерактивных является *детерминизм* – возможность определения состояния системы на основе последовательности наборов входных воздействий. Две одинаковые последовательности входных воздействий реактивной системы приводят ее в то же самое состояние, а интерактивную – необязательно.

Недетерминированной системой можно считать компьютер под управлением многозадачной ОС, выполняющий задачу пользователя. Состояние системы не детерминировано в силу наличия нескольких задач и механизмов реализации виртуальной памяти, что не позволяет в некоторый момент времени предсказать (определить) характеристики и условия выполнения задачи – время реакции на воздействие, количество доступной памяти, производительность процессора для задачи.

В то же время, компьютер с ОС реального времени, даже многозадачной, выполняющий приложение реального времени может рассматриваться детерминированной системой, поведение которой определено для набора входных воздействий.

Рассматривая поведение системы во времени, можно говорить о *конкурентности* элементов системы – поскольку отдельные ее элементы функционируют параллельно или последовательно по отношению друг к другу, что делает их состязающимися за аппаратные ресурсы. В зависимости от применяемой МОВ конкурентность системы проявляется в упорядочивании взаимодействий ее элементов – синхронные системы, асинхронные, дискретных событий и т.д.

Языки программирования, применяемые для интерактивных систем не удовлетворяют нуждам описания детерминированных реактивных конкурентных систем, т.к. ориентированы на программирование трансформационных задач, синхронизируемых системными вызовами ОС.

Главное требование к языку описания реактивных систем – *безопасность*, что выражается в возможности верификации описания системы или ее элемента. Эта возможность достигается за счет введения формальной модели организации вычислений. Необходимость обеспечения *безопасности* является важным свойством реактивных систем.

17 Рекомендуемая литература

1. Автоматическое управление в химической промышленности: Учебник для вузов./ Под ред. Е.Г.Дудникова.- М.: Химия, 1987. 368с., ил.

2. Шенброт И.М., Антропов М.В., Давиденко К.Я. Распределенные АСУ технологическими процессами. - М.: Энергоиздат, 1985. 240с., ил.

3. Рей У. Методы управления технологическими процессами. -М.: Мир, 1988. 368 с., ил.

4. Стефани Е.П. Основы построения АСУТП: - М.: Энергия, 1982. 352 с., ил.

5. Цирлин Л.М. Оптимальное управление технологическими процессами. - М.: Энергоиздат, 1986. 400 с.

6. Месарович Н., Мако Д., Такахара Н. Теория иерархических многоуровневых систем. - М.: Мир, 1973. 344 с.

7. Денисов А.А., Колесников Д.Н. Теория больших систем управления: Учебное пособие для вузов. - Л.: Энергоиздат, Ленинградское отд., 1982. 288 с.