Классификация электронных систем

1. по принципу действия (аналоговая и цифровая),
2. по назначению (хранение, накопление, обработка данных) и т.д.).
3. основные понятия:
   - электронная система,
   - задача,
   - быстродействие,
   - гибкость,
   - избыточность,
   - интерфейс.
4. микропроцессоры - базовый элемент новой технической революции.

Типы цифровых устройств

- Устройства с «жёсткой» логикой работы (выходные сигналы в каждый момент однозначно определяются значениями входных сигналов и это соответствие не может быть изменено).

  Устройства с «жёсткой» логикой быстрее, проще для простых функций, но сложнее в разработке.

- Устройства с программируемым алгоритмом работы (соответствие выходных сигналов входным сигналам может быть изменено программой — набором управляющих кодов).

  Устройства с программируемой логикой медленнее, но проще для реализации сложных функций и проще в разработке.

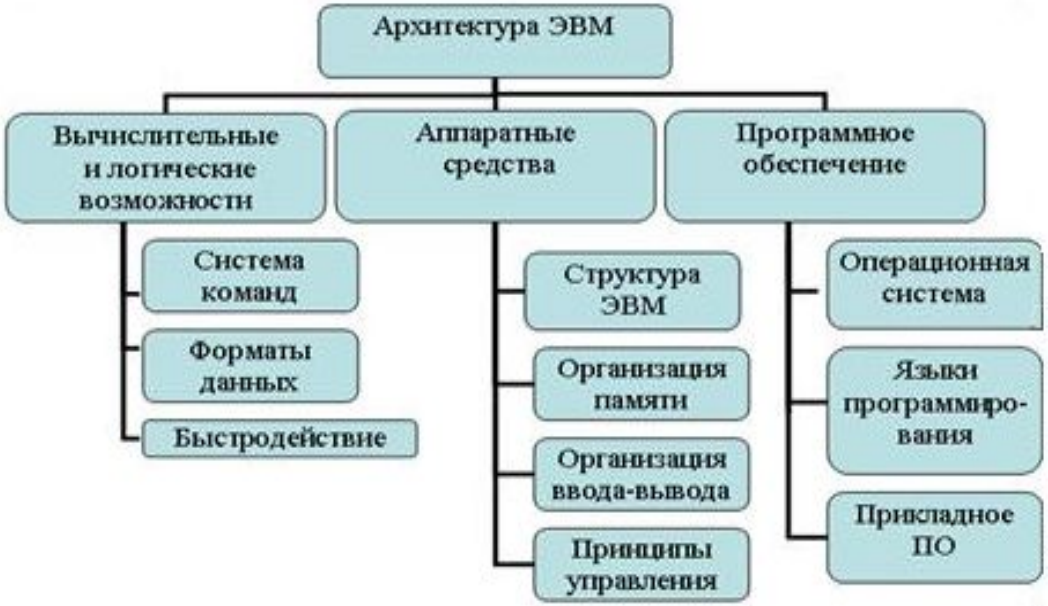
Основные понятия микропроцессорной техники

- *архитектура* (открытая и закрытая),
- *система команд* (ортогональная и не ортогональная).

Под архитектурой ЭВМ понимается совокупность общих принципов организации аппаратно-программных средств и их характеристик, определяющая функциональные возможности ЭВМ при решении соответствующих классов задач.

Архитектура ЭВМ определяет:

- принципы действия,
- информационные связи и
- взаимное соединение основных логических узлов компьютера:
  - процессора,
  - запоминающего устройства – внутреннего и внешнего,
  - периферийных устройств.



Понятие архитектуры ЭВМ

Общность архитектуры разных компьютеров обеспечивает их совместимость с точки зрения пользователя.

Закрытая архитектура – это архитектура, спецификации которой либо не опубликованы либо в них не предусмотрено подключение дополнительных устройств расширяющих функциональность вычислительной системы.
Закрытая архитектура не дает возможности другим производителям выпускать для компьютеров дополнительные внешние компоненты.

Открытая архитектура – имеет открытые спецификации и предполагает наличие единого стандарта при разработке устройств, располагающихся на материнской плате и платах расширения.

Структура компьютера – это некоторая модель, устанавливающая состав, порядок и принципы взаимодействия входящих в нее компонентов.

В настоящее время наибольшее распространение в ЭВМ получили 2 типа архитектуры:

– Принстонская (фон Неймана) и
– Гарвардская.

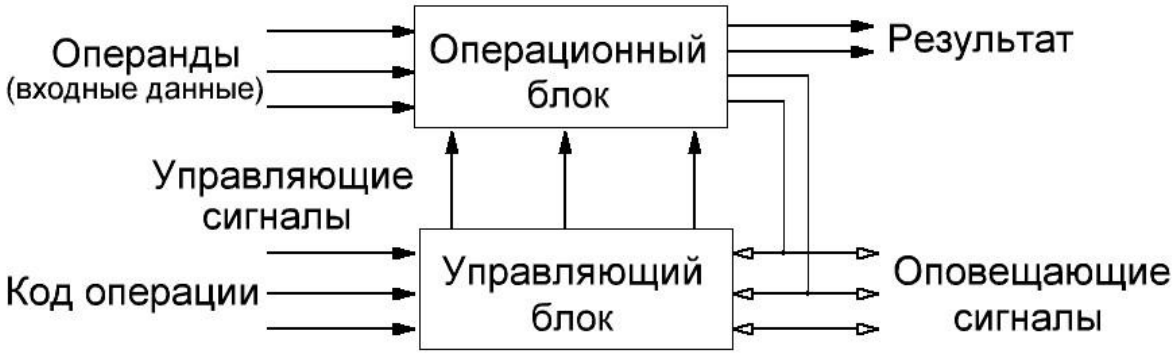
Обе они выделяют 2 основных узла ЭВМ: центральный процессор и память компьютера.

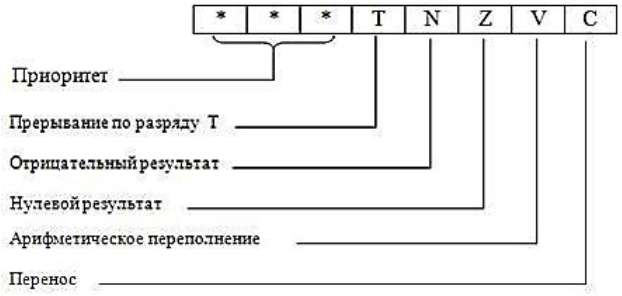
Основное различие заключается в структуре памяти:

- в Принстонской архитектуре программы и данные хранятся в одном массиве памяти и передаются в процессор по одному каналу,
- в Гарвардская архитектура предусматривает отдельные хранилища и потоки передачи для команд и данных.

Декомпозиция вычислительного устройства. Микропрограммная интерпретация языка команд микропроцессора.

Декомпозиция вычислительного устройства





Операционный блок – **совокупность электронных устройств** (регистров, сумматоров и других узлов), производящих **приём** из внешней среды наборов данных, их преобразование и **выдачу** во внешнюю среду **результатов преобразования**, а также выдачу в управляющий блок и внешнюю среду **оповещающих сигналов**, которые могут представлять из себя сообщения о знаках, особых значениях промежуточных и конечных результатов. Например, оповещающие сигналы располагаются в **регистре PSW** (*Processor status word*).

Процесс функционирования во времени устройства обработки цифровой информации (операционного устройства) **состоит из последовательности тактовых интервалов**, в которых операционный блок производит определенные элементарные **операции преобразования кодов** (слов).

Исполнение элементарных **операций в операционном блоке инициируется поступлением** в него соответствующих **управляющих сигналов**.

Управляющий блок – устройство, которое вырабатывает распределённую во времени последовательность управляющих сигналов, которые порождают в операционном блоке нужную последовательность микроопераций.

Последовательность **управляющих сигналов** определяется сигналами **кода операции**, поступающие в управляющий блок извне и **оповещающих сигналов** зависящих от операндов и промежуточных результатов преобразования в операционном блоке.

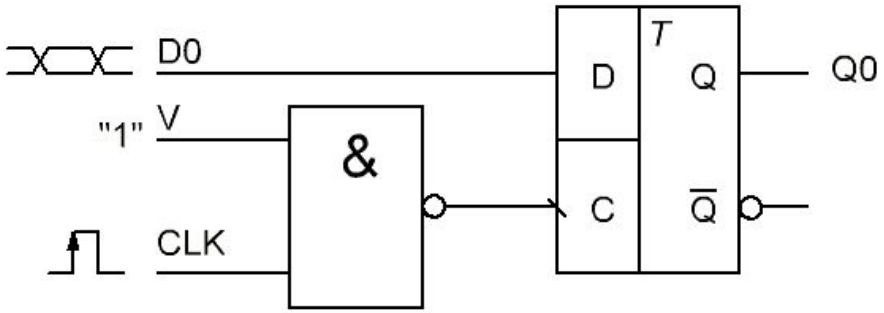
Для сокращения числа управляющих сигналов (цепей), выходящих из управляющего блока, **микрокоманды могут определенным образом кодироваться**, например при помощи шифратора 8 разрядным двоичным кодом может быть закодировано $2^8 = 256$ микрокоманд.

Два подхода реализации управляющего устройства

1. **Использование жесткой логики**
2. **Использование программируемой логики**

Основные определения микропрограммного управления

Микрооперация – это элементарная функциональная операция, выполняемая **за один тактовый интервал** и приводимая в действие **одним управляющим сигналом**



Микрокоманда – это некоторая совокупность одновременно выполняемых микроопераций. Как частный случай, микрокоманда может состоять из одной микрооперации.

Микропрограмма – это последовательность микрокоманд, обеспечивающая выполнение одной функционально законченной операции вычислительного устройства. Например, умножение, деление, десятичная коррекция и т.д.

Функционирование вычислительного устройства может быть описано совокупностью реализуемых в нем микропрограмм.

Назначение и функции микропроцессора в электронной системе. Важнейшие характеристики микропроцессора.

Микропроцессор — это центральный блок персонального компьютера, предназначенный для управления работой всех остальных блоков и выполнения арифметических и логических операций над информацией.

Микропроцессор выполняет следующие основные функции:

1) чтение и дешифрацию команд из основной памяти;
2) чтение данных из основной памяти и регистров адаптеров внешних устройств;
3) прием и обработку запросов и команд от адаптеров на обслуживание внешних устройств;
4) обработку данных и их запись в основную память и регистры адаптеров внешних устройств;
5) выработку управляющих сигналов для всех прочих узлов и блоков компьютера.

В состав микропроцессора входят следующие устройства.

1) Арифметико-логическое устройство предназначено для выполнения всех арифметических и логических операций над числовой и символьной информацией.
2) Устройство управления координирует взаимодействие различных частей компьютера. Выполняет следующие основные функции:
    - формирует и подает во все блоки машины в нужные моменты времени определенные сигналы управления (управляющие импульсы), обусловленные спецификой выполнения различных операций;
    - формирует адреса ячеек памяти, используемых выполняемой операцией, и передает эти адреса в соответствующие блоки компьютера;
    - получает от генератора тактовых импульсов обратную последовательность импульсов.

Важнейшими характеристиками микропроцессора являются:

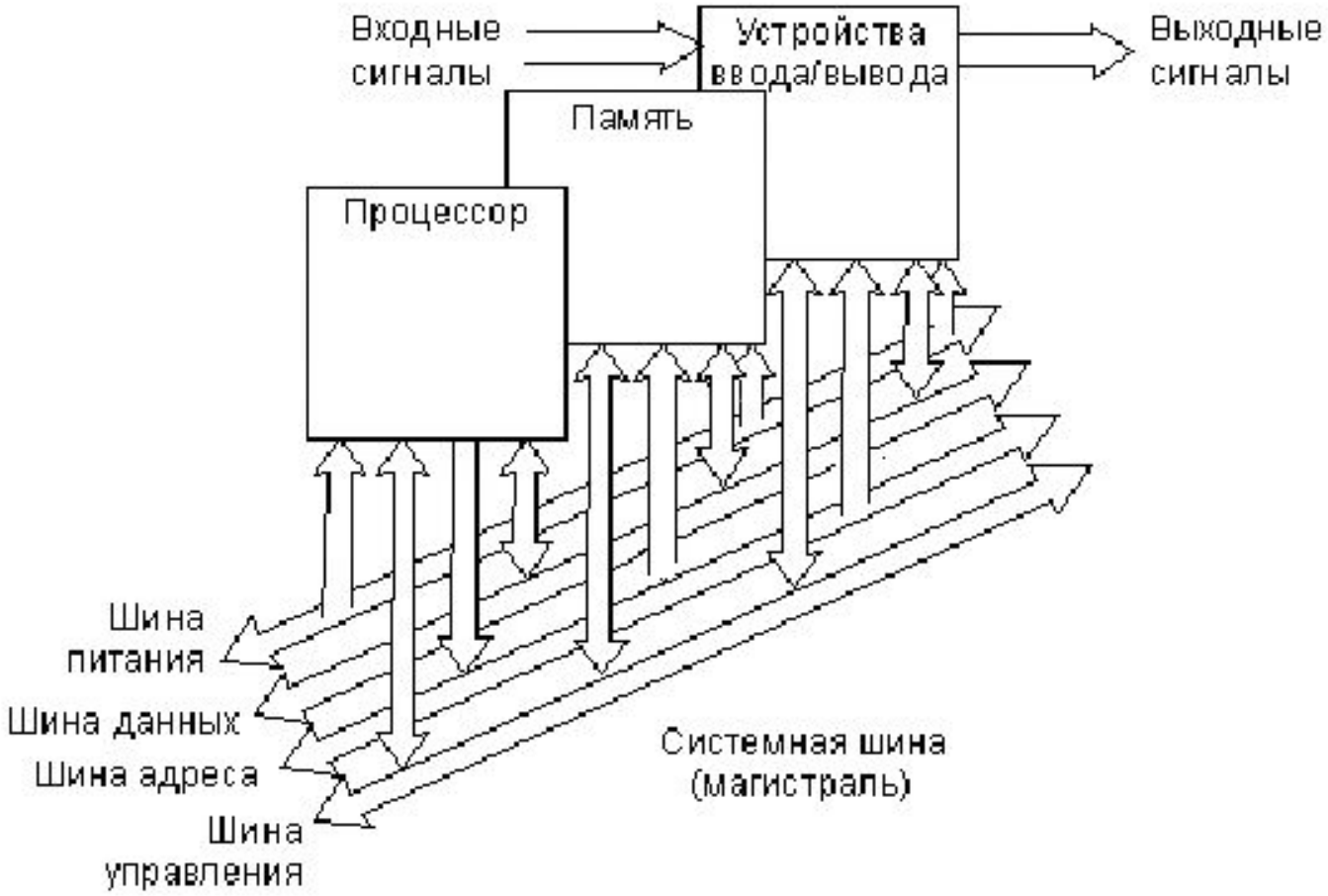
1) тактовая частота. Характеризует быстродействие компьютера. Режим работы процессора задается микросхемой, называемой генератором тактовых импульсов. На выполнение процессором каждой операции отводится определенное количество тактов. Тактовая частота указывает, сколько элементарных операций выполняет микропроцессор за одну секунду. Тактовая частота измеряется в МГц;
2) разрядность процессора — это максимальное количество разрядов двоичного числа, над которым одновременно может выполняться машинная операция. Чем больше разрядность процессора, тем больше информации он может обрабатывать в единицу времени и тем больше, при прочих равных условиях, производительность компьютера;

Процесс взаимодействия пользователя с персональным компьютером (ПК) непременно включает процедуры ввода входных данных и получение результатов обработки этих данных. Поэтому, обязательными составляющими типичной конфигурации ПК являются разнообразные устройства ввода-вывода, среди которых можно выделить стандартные устройства, без которых современный процесс диалога вообще невозможен, и периферийные, т.е дополнительные. К стандартным устройствам ввода-вывода относятся монитор, клавиатура и манипулятор мышка.

Типовая структура и основные элементы микропроцессора.

Типовая структура вычислительной системы

Структура вычислительной системы – это некоторая модель, устанавливающая состав, порядок и принципы взаимодействия входящих в нее компонентов



Все устройства вычислительной (микропроцессорной) системы объединяются **общей системной шиной** (она же называется еще **системной магистралью** или **каналом**).

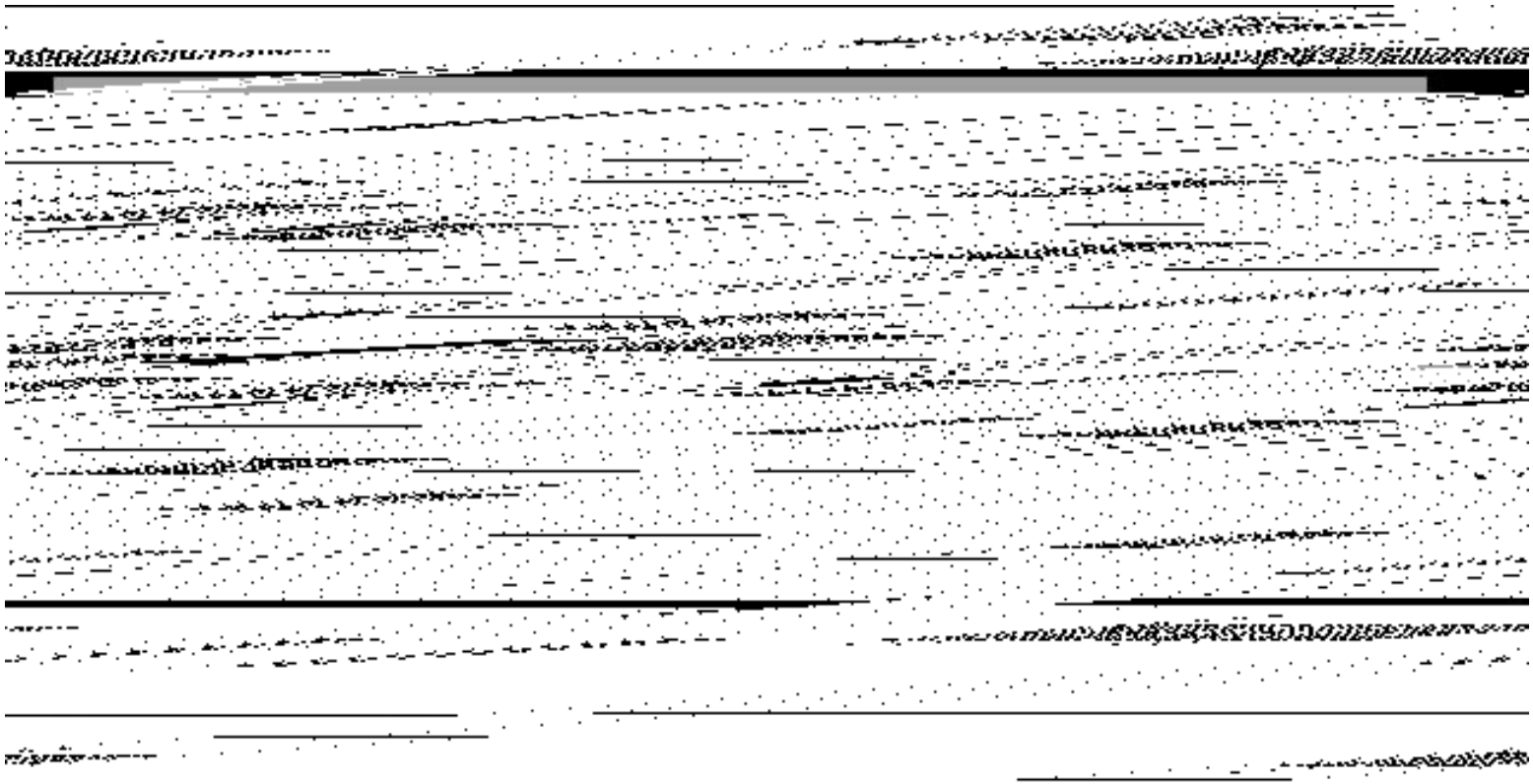
Шина адреса – служит для определения адреса (номера устройства) с которым микропроцессор обменивается в данный момент времени. **ША всегда однонаправлена от микропроцессора к устройству**. Источником адреса практически всегда является микропроцессор. Разрядность ША определяет количество возможных устройств подключённых к микропроцессору.

Шина данных – используется для передачи информационных кодов между всеми устройствами микропроцессорной системы. **ШД всегда двунаправлена**. Разрядность ШД определяет производительность микропроцессора. Чем больше разрядов в ШД тем больший объём информации может быть обработан за один такт синхронизации.

Шина управления – состоит из отдельных управляющих сигналов, каждый из которых во время обмена информацией выполняет свою функцию. Сигналы на ШУ определяют тип текущего цикла обмена и фиксируют моменты времени, соответствующие разным частям или стадиям цикла, а так же обеспечивают согласование работы процессора с работой памяти и устройств ввода/вывода, обслуживают запросы и предоставление прерываний, запросы и предоставление прямого доступа к памяти. Линии ШУ могут быть как одно так и двунаправленными.

Шина питания – служит для подвода питающих напряжений к отдельным элементам системы. Состоит из линий питания и общего провода. Может объединять несколько источников питания.

Основные элементы микропроцессора



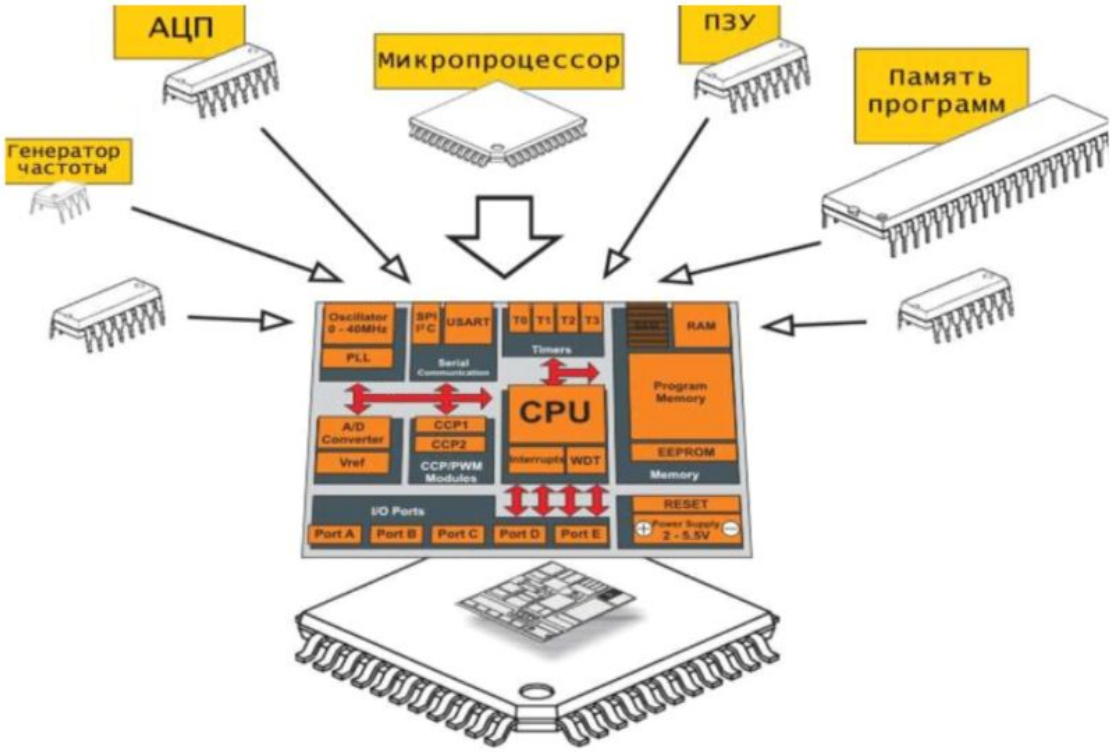
1. схема управления выборкой команд,
2. регистры (общего назначения, аккумулятор, специального назначения),
3. схемы логики,
4. Схемы управления прерываниями и прямого доступа к памяти).

Назначение и организация микроконтроллеров. Основные архитектуры микроконтроллеров

Микроконтроллер – это полноценный **компьютер на одной микросхеме**.

Предназначен для управления различными электронными устройствами и осуществления взаимодействия между ними в соответствии с заложенной в микроконтроллер программой.

В отличие от микропроцессоров, используемых в персональных компьютерах, микроконтроллеры содержат встроенные дополнительные устройства. Эти устройства выполняют свои задачи под управлением микропроцессорного ядра микроконтроллера.



Однокристальная микро ЭВМ

Основным классификационным признаком микроконтроллеров (МК) является разрядность данных, обрабатываемых *арифметико-логическим устройством* (АЛУ).

По этому признаку они делятся на 4-, 8-, 16-, 32- и 64-разрядные.

Все типы **МК** можно условно разделить на три основных класса:

- **8-разрядные *МК* для встраиваемых приложений;**
- **16- и 32-разрядные управляющие *МК*;**
- **цифровые сигнальные процессоры (DSP) для обработки данных.**

Отличительные признаки микроконтроллеров

1. **модульная организация**, при которой на базе одного ***процессорного ядра*** (центрального процессора) проектируется ряд (линейка) ***МК***, различающихся объемом и типом ***памяти программ***, объемом ***памяти данных***, набором периферийных модулей, частотой синхронизации;
2. **использование закрытой архитектуры** ***МК***, которая характеризуется отсутствием линий магистралей адреса и данных на выводах корпуса ***МК***. Таким образом, ***МК*** представляет собой **законченную систему обработки данных**, наращивание возможностей которой с использованием параллельных магистралей адреса и данных не предполагается;
3. **использование типовых функциональных периферийных модулей** (таймеры, процессоры событий, контроллеры последовательных интерфейсов, аналого-цифровые преобразователи и др.), имеющих незначительные отличия в алгоритмах работы в ***МК*** различных производителей;
4. **расширение числа режимов работы периферийных модулей**, которые задаются в процессе инициализации регистров специальных функций ***МК***.

При модульном принципе построения все ***МК*** одного семейства содержат ***процессорное ядро***, одинаковое для всех ***МК*** данного семейства, и изменяемый функциональный блок, который отличает ***МК*** разных моделей.

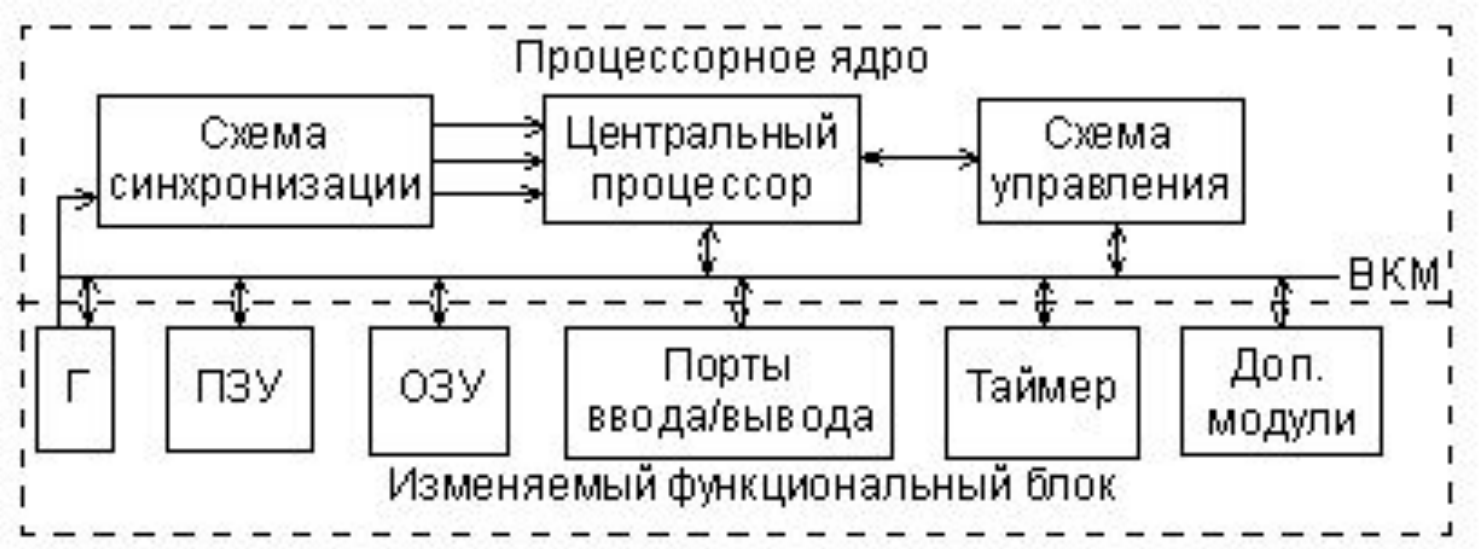


Рис. 1. Модульная организация МК

Процессорное ядро **изменяемого функционального блока** могут входить и такие **дополнительные модули** как:

1. **центральный процессор**;
2. **внутреннюю контроллерную магистраль** (ВКМ) в составе шин адреса, данных и управления;
3. **схему синхронизации *МК***;
4. **схему управления режимами работы *МК***, включая поддержку режимов пониженного энергопотребления, начального запуска (сброса) и т.д.

- **Порты параллельного и последовательного ввода-вывода,**
- **Компараторы напряжения,**
- **Аналого-цифровые преобразователи (АЦП)**
- **Цифро-аналоговые преобразователи (ЦАП)**
- **Генераторы широтно-импульсной модуляции (ШИМ) и другие.**

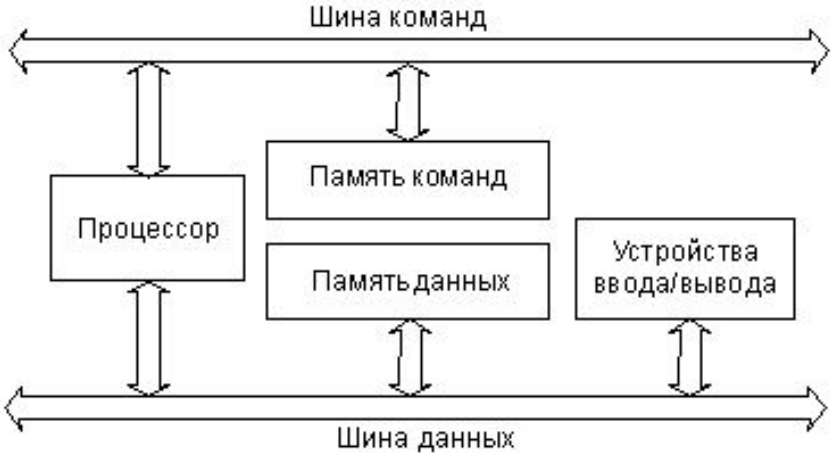
Классическая принстонская архитектура

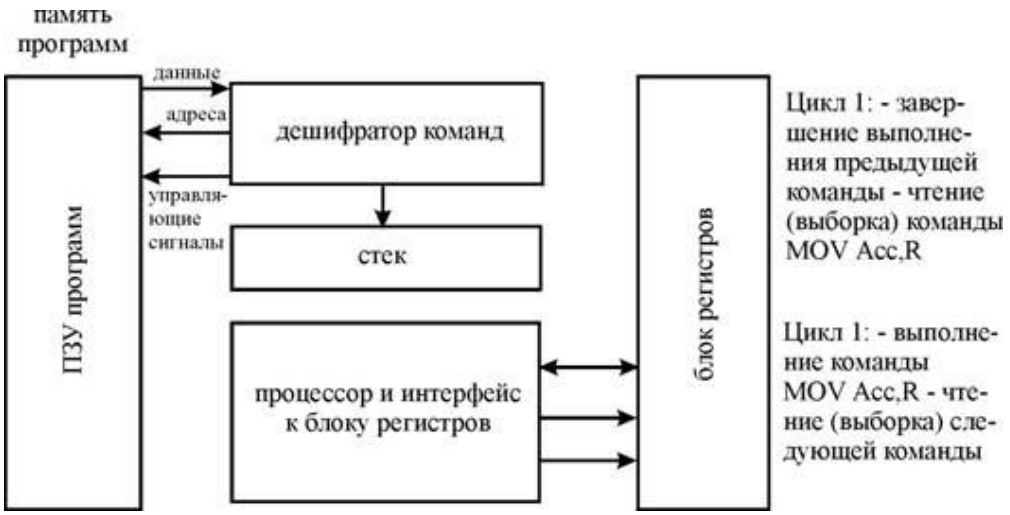
Принстонская архитектура — архитектура микропроцессорной системы с единой шиной для данных и команд (одношинная архитектура)

Классическая Гарвардская архитектура

Гарвардская архитектура — архитектура ЭВМ, отличительным признаком которой является раздельное хранение и обработка команд и данных.

Архитектура была разработана Говардом Эйкеном в конце 1930-х годов в Гарвардском университете.





Организация связи микроконтроллера с внешней средой.

Связь МК с внешней средой производится посредством использования портов ввода-вывода.

Порты ввода-вывода бывают:

1. Цифровые
    1) Параллельные.
    2) Последовательные:
        - USART
        - SPI
        - I2C
        - USB
        - LIN и т.д.
2. Аналоговые
    - АЦП
    - ЦАП

Назначение параллельного порта

Параллельные порты предназначены для обмена информацией микропроцессора с внешними устройствами, при этом в качестве внешнего устройства может использоваться другой микропроцессор или компьютер.

Параллельные порты позволяют согласовывать низкую скорость работы внешнего устройства и высокую скорость работы системной шины микропроцессора.

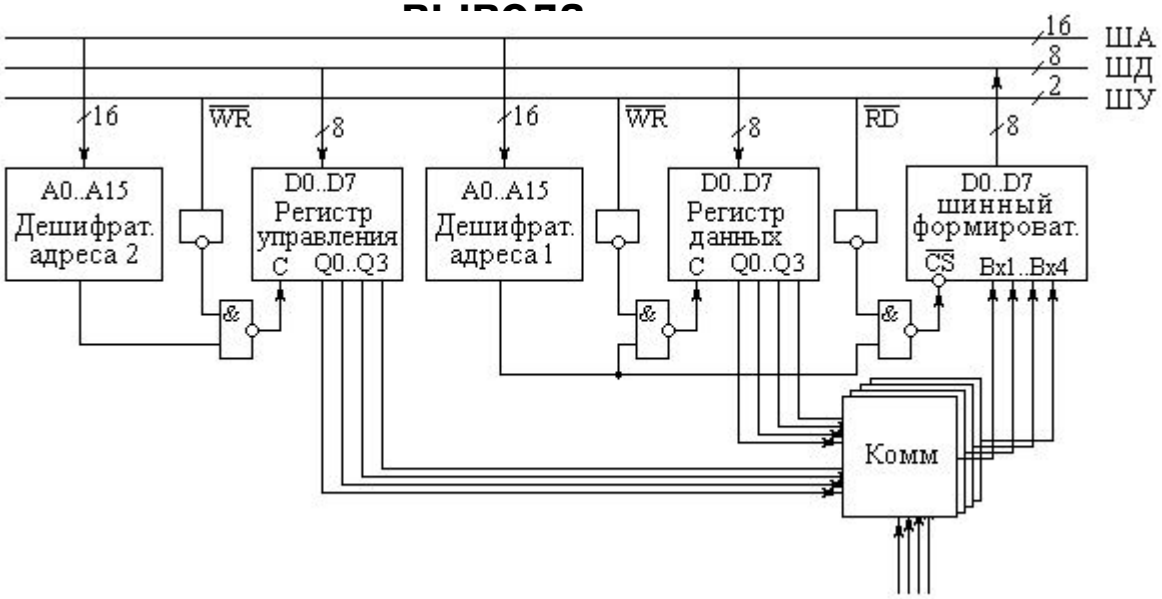
С точки зрения внешнего устройства порт представляет собой обычный источник или приемник информации со стандартными цифровыми логическими уровнями (обычно ТТЛ), а с точки зрения микропроцессора – это ячейка памяти, в которую можно записывать данные или в которой сама собой появляется информация.

В качестве внешнего устройства может служить любой объект управления или источник информации:

- различные кнопки и индикаторы,
- цифровые (бинарные) датчики,
- различные микросхемы:
  - Усилители-формирователи,
  - синтезаторы частот,
  - дополнительная память,
- исполнительные механизмы,
- двигатели, реле и т.д.

В зависимости от направления передачи данных параллельные порты называются портами ввода, **вывода** или портами ввода-вывода.

Порт ввода-вывода



Структурная схема параллельного порта ввода-вывода.

Порты ввода-вывода входят с структуру МК в качестве универсального модуля.

В одной универсальном модуле размещаются и порт ввода и порт вывода информации, а для подключения этих портов к внешним ножкам МК используется *коммутатор*.

Для управления этим коммутатором используется еще один (внутренний) параллельный порт вывода, регистр данных которого называется регистром управления параллельного порта ввода-вывода, а сам порт называется портом ввода-вывода.

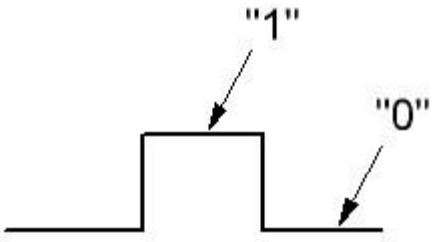
Адрес для регистра управления обычно назначается рядом с *адресом регистра данных* порта ввода-вывода.

Режимы работы микропроцессорных систем. Организация и основные типы программных обменов данными в микропроцессорной системе.

Режимы - синхронный, асинхронный
Обмены - программный обмен, обмен по прерываниям, прямой доступ к памяти

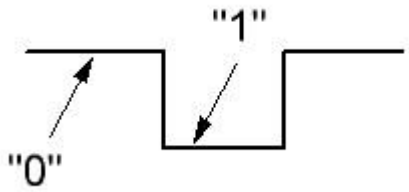
Физическое представление сигналов

1. Положительная логика



Сигналы на ША и ШД передаются в положительной логике

2. Отрицательная логика

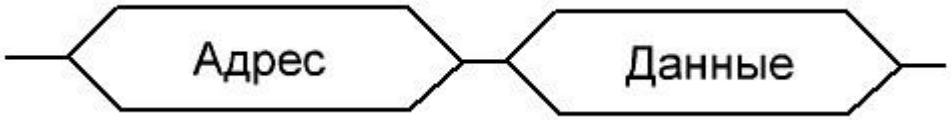


Сигналы на ШУ передаются в отрицательной логике

Немультиплексированная шина

Средний уровень обозначает, что состояние сигналов в данные временные интервалы не важны

Мультиплексированная шина



Недостаток: существенное замедление скорости обмена

Достоинство: сокращение количества выводов на корпусе микропроцессора

Иногда применяют **частичное мультиплексирование**

Самые **главные сигналы** на **ШУ** это **стробы обмена**.

Строб обмена формируется микропроцессором и определяет момент времени когда именно разрешена пересылка данных по ШД.

Типы стробов обмена:

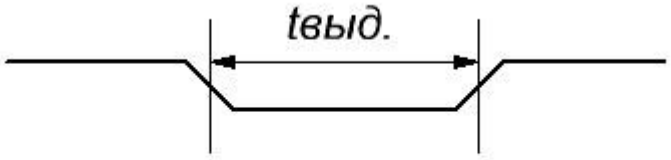
1. **Строб записи** (вывода) – определяет момент времени, когда устройство исполнитель может принимать данные, выставленные микропроцессором на ШД

2. **Строб чтения** (ввода) – определяет моменты времени, когда устройство исполнитель должно выдать на ШД код данных, который будет прочитан микропроцессором.

Большое значение имеет процесс окончания обмена в пределах цикла обмена, **момент завершения строба**.

Типы обменов:

1. Синхронный обмен. Микропроцессор заканчивает обмен данными самостоятельно, через раз и навсегда установленный временной интервал выдержки $t_{выд}$, то есть без учёта интересов устройства исполнителя.

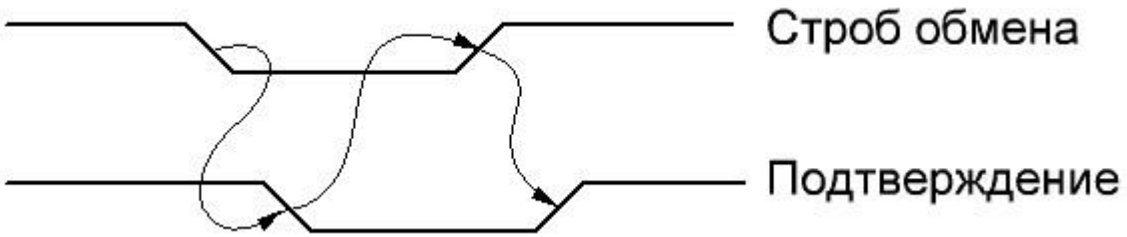


Строб обмена

Достоинства: Простой протокол обмена, малое количество управляющих сигналов.

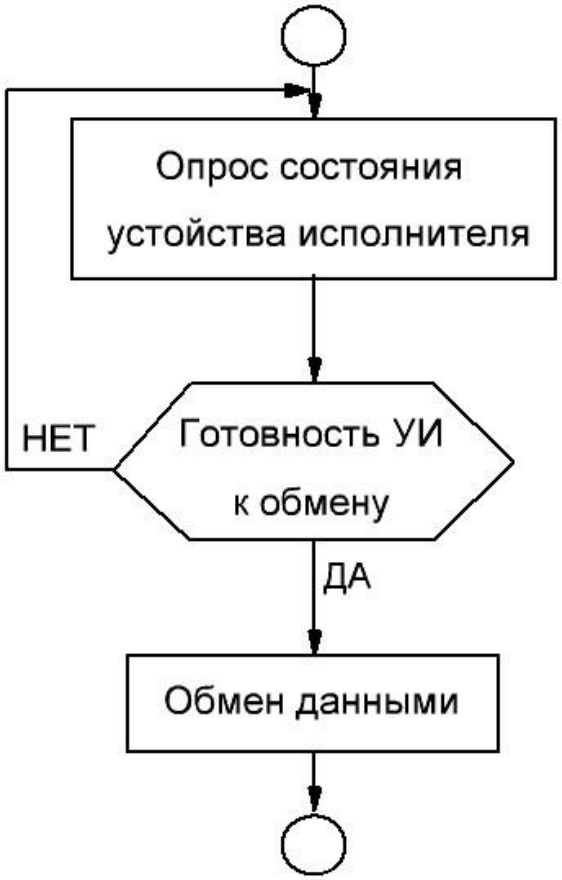
Недостаток: Высокие требования к быстродействию исполнительного устройства, его постоянная готовность к началу обмена. Нет гарантии, что устройство исполнитель успел выполнить требуемую операцию.

2. Асинхронный обмен. Микропроцессор начинает и заканчивает обмен только тогда, когда устройство исполнитель подтвердит свою готовность к обмену и выполнению операции обмена специальными сигналами (режим *handshake* – рукопожатие)



Достоинства: Надёжность пересылки данных. Возможность работы с разными по быстродействию устройствами исполнителями.

Недостаток: Необходимость формирования дополнительных сигналов, т.е. дополнительные аппаратные затраты.



Алгоритм асинхронного обмена между устройством исполнителя и ШД

Циклы обмена

Системная магистраль является самым **главным системообразующим элементом** в микропроцессорной системе с помощью которой производится обмен информацией между основными компонентами микропроцессорной системы**.**

Обмен информацией в микропроцессорной системе **происходит в циклах обмена.**

Под циклом обмена информацией понимается временной интервал, в течении которого происходит выполнение одной элементарной операции обмена по шине данных.

Во время каждого цикла обмена устройства, участвующие в обмене информацией, **передают друг другу *информационные и управляющие сигналы* в строго установленном порядке**, который называется **протокол обмена информацией**.

Длительность цикла обмена может быть **постоянной** или **переменной**, но **всегда включает в себя несколько тактов синхронизации**. Поэтому даже **в идеальной системе частота чтения или записи данных меньше тактовой частоты системы**.

В зависимости от архитектуры системной магистрали **циклы чтения команд** и **пересылки данных могут происходить как одновременно, так и последовательно.**

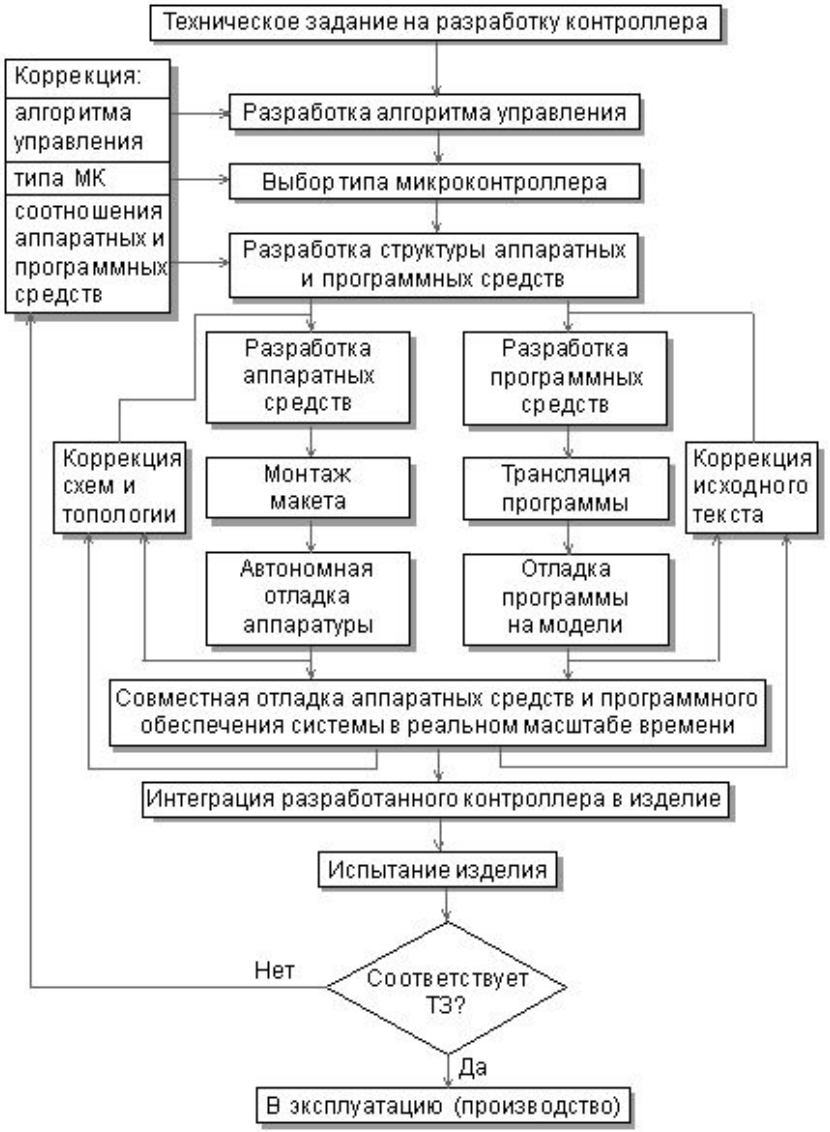
Типы циклов обмена:

1. **Цикл записи** (вывода).
2. **Цикл чтения** (ввода).
3. **Цикл чтение-модификация-запись.**
4. **Цикл обработки прерывания**.
5. **Цикл прямого доступа к памяти** (ПДП).

Особенности и основные этапы разработки и проектирования цифровых устройств на основе микроконтроллеров.

Параллельная разработка и отладка аппаратных средств с разработкой и отладкой программного обеспечения

Разработка микропроцессорной системы на основе микроконтроллера



Разработка микропроцессорной системы основывается на **принципе неразрывного проектирования и отладки аппаратных и программных средств**

1. Техническое задание формулирует требования к контроллеру с точки зрения **реализации определенной функции управления**.
2. Техническое задание включает в себя набор требований, который определяет, **что пользователь хочет от контроллера и что разрабатываемый прибор должен делать.**
3. Техническое задание должно иметь вид текстового описания.

1. На основании требований пользователя **составляется функциональная спецификация**, которая определяет функции, выполняемые контроллером для пользователя после завершения проектирования, уточняя тем самым, насколько устройство соответствует предъявляемым требованиям. Она включает в себя **описания форматов данных**, как **на входе**, так и **на выходе**, а также **внешние условия**, управляющие действиями контроллера. **Функциональная спецификация и требования пользователя являются критериями оценки функционирования контролера после завершения проектирования**. Может потребоваться проведение нескольких итераций, включающих обсуждение требований и функциональной спецификации с потенциальными пользователями контроллера, и соответствующую коррекцию требований и спецификации. Требования к типу используемого микроконтроллера формулируются на данном этапе чаще всего в неявном виде.
2. **Этап разработки алгоритма управления является наиболее ответственным**, поскольку ошибки данного этапа обычно обнаруживаются только при испытаниях законченного изделия и приводят к необходимости дорогостоящей переработки всего устройства. Разработка алгоритма обычно сводится к выбору одного из нескольких возможных вариантов алгоритмов, отличающихся соотношением объема программного обеспечения и аппаратных средств.

При выборе типа микроконтроллера учитываются следующие **основные характеристики:**

1. разрядность;
2. быстродействие;
3. набор команд и способы адресации;
4. требования к источнику питания и потребляемой мощности в различных режимах;
5. объем памяти программ (ПЗУ) и памяти данных (ОЗУ);
6. возможность расширения памяти программ и данных;
7. наличие и возможности периферийных устройств, включая средства поддержки работы в реальном времени (таймеры, процессоры событий и т. п.);
8. возможность перепрограммирования в составе устройства;
9. наличие и надежность средств защиты внутренней информации;
10. возможность поставки в различных вариантах конструктивного исполнения;
11. стоимость в различных вариантах исполнения;
12. наличие полной документации;
13. наличие и доступность эффективных средств программирования и отладки микроконтроллера;
14. количество и доступность каналов поставки, возможность замены изделиями других фирм.

3. **Для реализации на практике возможности выбора оптимального микроконтроллера необходима**

   1) достаточно **глубокая проработка алгоритма управления,**

   2) **оценка объема исполняемой программы**

   3) **числа линий сопряжения с объектом и т.д.**

Допущенные на данном этапе просчеты могут впоследствии привести к необходимости смены модели микроконтроллера и повторной разводки печатной платы макета контроллера. Поэтому **всегда целесообразно выполнять предварительное моделирование** основных элементов прикладной программы **с использованием программно-логической модели выбранного микроконтроллера.**

4. **На этапе разработки структуры контроллера окончательно определяется состав имеющихся и подлежащих разработке аппаратных модулей, протоколы обмена между модулями, типы разъемов и т.д.**. Выполняется предварительная проработка конструкции контроллера.

5. **В части программного обеспечения определяются состав и связи программных модулей, язык программирования**. На этом же этапе осуществляется выбор средств проектирования и отладки.

Разработка и отладка аппаратных средств

Разработка аппаратных средств включает в себя:

1. **разработку общей принципиальной схемы,**
2. **разводку топологии плат**,
3. **монтаж макета и его автономную отладку**.

На этапе ввода принципиальной схемы и разработки топологии используются, как правило, распространенные системы проектирования типа «P-Cad», «Altium Designer», «OrCad» и др.

Автономная отладка аппаратуры на основе микроконтроллера с открытой архитектурой **предполагает контроль состояния многоразрядных магистралей адреса и данных с целью проверки правильности обращения к внешним ресурсам памяти и периферийным устройствам**.

Закрытая архитектура микроконтроллера предполагает реализацию большинства функций разрабатываемого устройства внутренними средствами микроконтроллера. Поэтому разрабатываемый контроллер будет иметь **малое число периферийных ИС**, а **обмен** с ними будет идти **преимущественно по последовательным интерфейсам**. **Наибольшее внимание** нужно уделить вопросам **согласования по нагрузочной способности** параллельных портов микроконтроллера и отладка алгоритмов обмена по последовательным каналам.

Разработка и отладка программного обеспечения

В настоящее время самым мощным средством разработки программного обеспечения для микроконтроллера являются интегрированные среды разработки (**IDE** - англ. Integrated Development Environment), имеющие в своем составе:

- менеджер проектов,
- текстовый редактор,
- компилятор,
- средства отладки программного кода (симулятор, JTAG-отладчик)

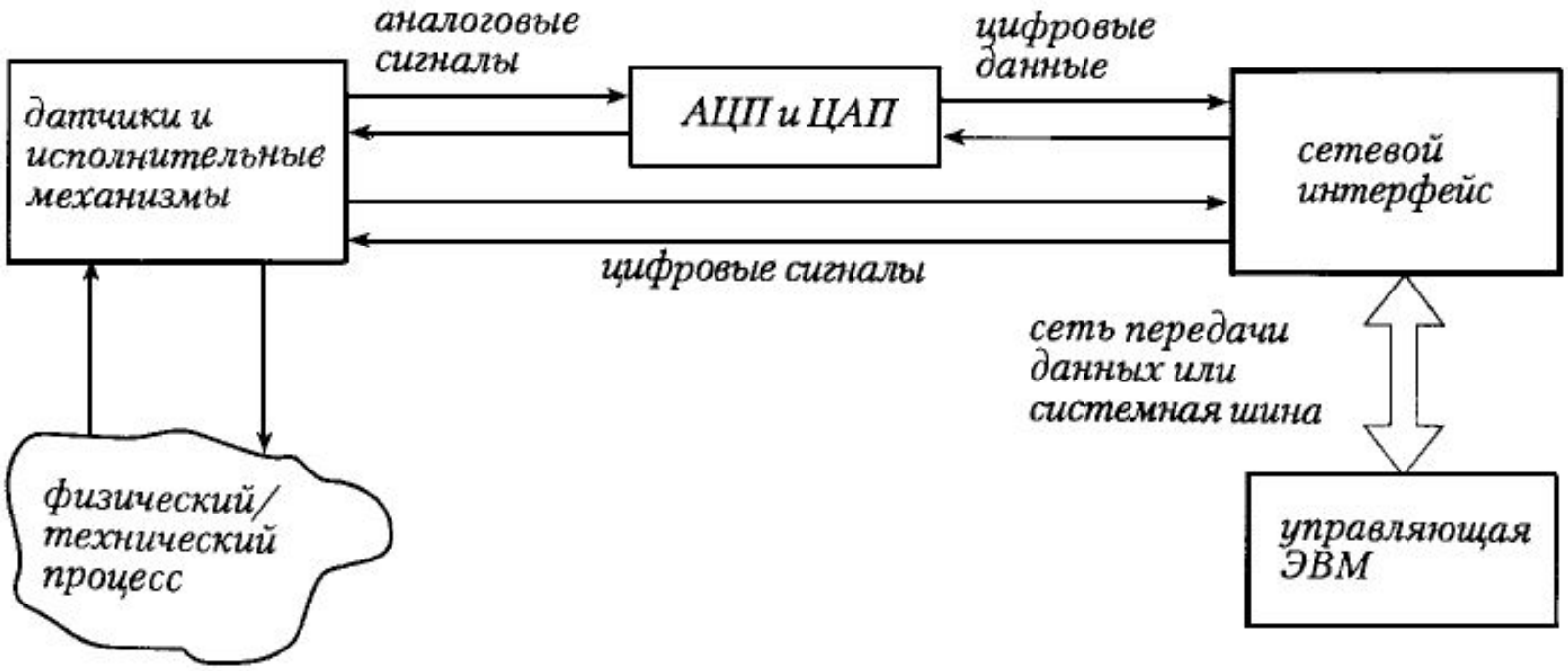
В некоторых случаях допускающие подключение компиляторов языков высокого уровня, например, Basic, Pascal, C++.

Программные **симуляторы используются для проверки и отладки программного обеспечения** и предоставляют пользователю возможность выполнять разработанную программу на программно-логической модели микроконтроллера.

В симуляторе моделируется работа ЦП, всех портов ввода/вывода, прерываний и другой периферии. Карта памяти моделируемого микроконтроллера загружается в симулятор автоматически, отладка ведется в символьных обозначениях регистров.

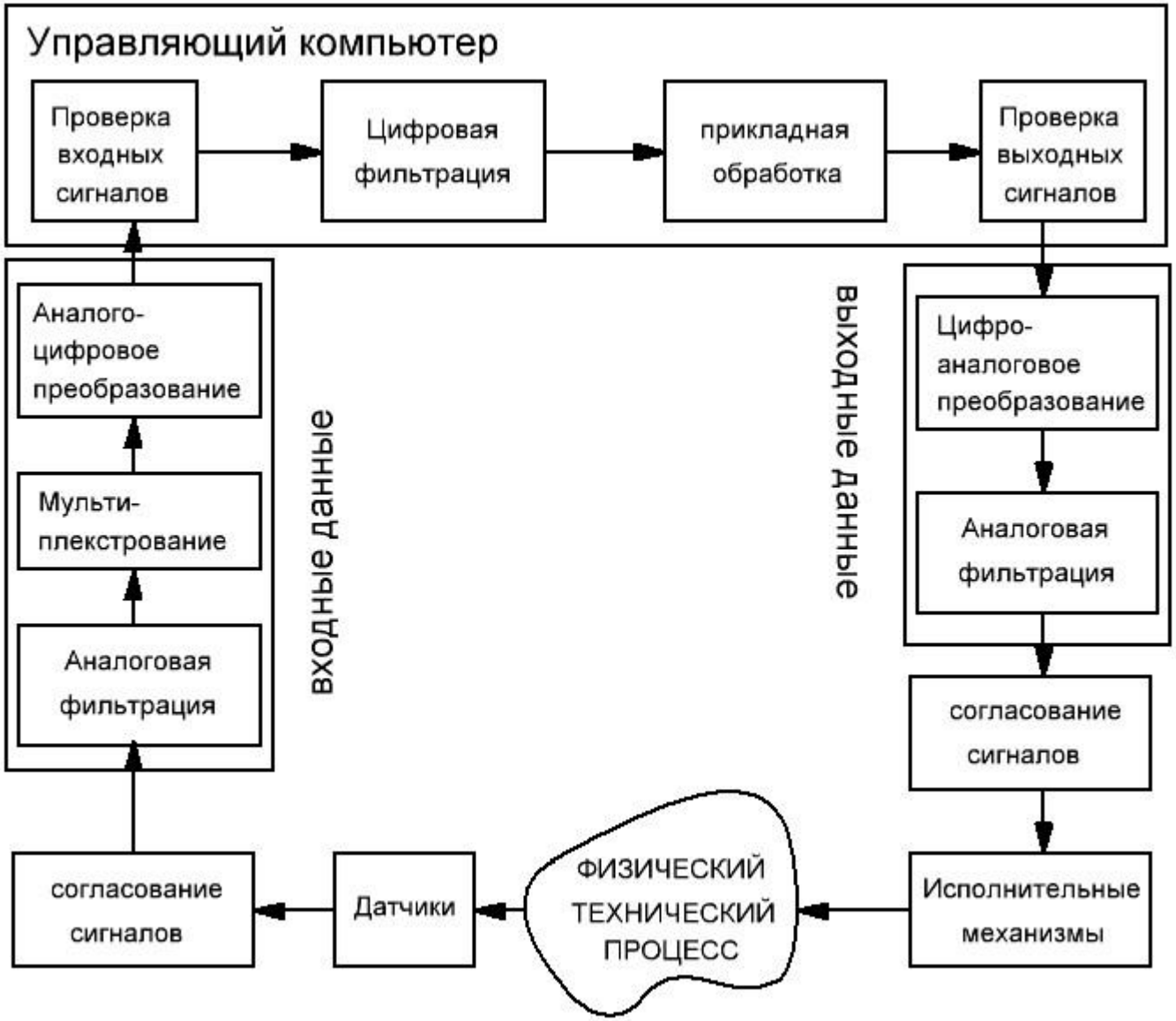
Структура микропроцессорной системы управления и ее основные компоненты.
Особенности цифрового управления техническими процессами.

Основные компоненты микропроцессорной системы управления



Система цифрового управления физическим/техническим процессом состоит из следующих компонентов:

1. Датчики и исполнительные механизмы;
2. Устройства преобразования информации – АЦП и ЦАП;
3. Каналы обмена информацией;
4. Управляющая ЭВМ.


Управляющий компьютер
Проверка входных сигналов
Цифровая фильтрация
прикладная обработка
Проверка выходных сигналов
Аналого-цифровое преобразование
Мульти-плекстрование
Аналоговая фильтрация
входные данные
выходные данные
Цифро-аналоговое преобразование
Аналоговая фильтрация
согласование сигналов
согласование сигналов
Датчики
ФИЗИЧЕСКИЙ ТЕХНИЧЕСКИЙ ПРОЦЕСС
Исполнительные механизмы

Назначение основных компонентов МСУ

Физический процесс контролируется с помощью датчиков.

ДАТЧИКИ – это устройства, преобразующие физические параметры процесса (температуру, давление или координаты) в электрическую величину, которую можно непосредственно измерить (сопротивление, ток или разность потенциалов).

Цифровые системы управления работают только с информацией, представленной в цифровой форме, поэтому полученные в результате измерений электрические аналоговые величины необходимо предварительно обработать с помощью АЦП. Для уменьшения влияния высокочастотных и импульсных помех на результат АЦ преобразования используются аналоговые фильтры НЧ.

УПРАВЛЯЮЩИЙ КОМПЬЮТЕР - центральный элемент системы управления.

1. Интерпретирует все поступающие от физического процесса данные;
2. Принимает решения в соответствии с алгоритмами программ обработки (основная задача);
3. Формирует управляющие сигналы;
4. Взаимодействует (обменивается информацией) с другими управляющими ЭВМ и/или с человеком-оператором и реагирует на его/их команды.

Исполнительные механизмы осуществляют непосредственное влияние на технический процесс.

ИСПОЛНИТЕЛЬНЫЙ МЕХАНИЗМ – преобразуют электрические сигналы в физические воздействия, главным образом движение – перемещение и вращение.

Информация от удаленных объектов поступает к управляющему компьютеру через

Особенности цифрового управления процессами

Свойства процессов, усложняющие управление

Уровень сложности системы управления определяется, в первую очередь, свойствами управляемого процесса.

Факторы усложняющих управление :

- нелинейность технического процесса;
- изменяющаяся внешняя среда;
- изменение условий протекания самого технического процесса;
- значительные временные задержки;
- наличие внутренних связей в техническом процессе.

Надо всегда учитывать, что Практически все физические процессы по своей природе НЕЛИНЕЙНЫ

Линейные соотношения в большинстве случаев фактически представляют собой искусственное упрощение реального состояния технического процесса.

Запаздывание сигналов или наличие зон нечувствительности (мертвых зон) представляет собой серьезную проблему для управления. Из-за этого управляющий компьютер (исполняет алгоритм регулирования) функционирует на основе устаревших данных, вплоть до того, что он может выдавать ложные команд.

Запаздывания всегда присутствуют в тех процессах, где некоторые параметры нельзя измерить непосредственно.

Многие типы датчиков и исполнительных механизмов характеризуются некоторым временем, необходимым для получения нового значения, измеряемой величины и формирования непосредственного управляющего воздействия.

На исполнение алгоритма управления микроконтроллером (управляющей ЭВМ) тоже требуется определённое время.

Отличие управляющего компьютера от обычного

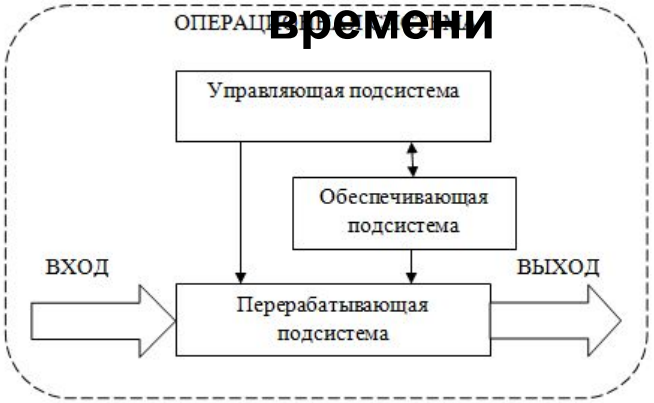
1. Управление процессами всегда происходит в реальном времени
2. Принципиально другой подход к программированию. Параллельное исполнение алгоритма обработки информации вместо последовательного.
3. Управляющий компьютер должен работать со скоростью, соответствующей скорости процесса. Само понятие "реальное время" указывает на то, что в реакции компьютерной системы на внешние события не должно быть заметного запаздывания. Это накладывает серьёзные требования на эффективность использования ресурсов компьютерной системы с учетом временные ограничений.
4. Ход исполнения программы нельзя определить заранее. Внешние сигналы могут прерывать или изменять последовательность исполнения операторов программы, причем для каждого нового прогона по-разному.
5. Особая специфика тестирования систем реального времени ввиду отсутствия предсказуемого порядка выполнения операторов программы по сравнению с обычными компьютерными системами.

Понятие реального времени.
Особенности программирование систем реального времени.

Под реальным временем понимается количественная характеристика, которая может быть измерена реальными физическими часами, в отличие от **логического времени**, определяющего лишь качественную характеристику, выражаемую относительным порядком следования событий.

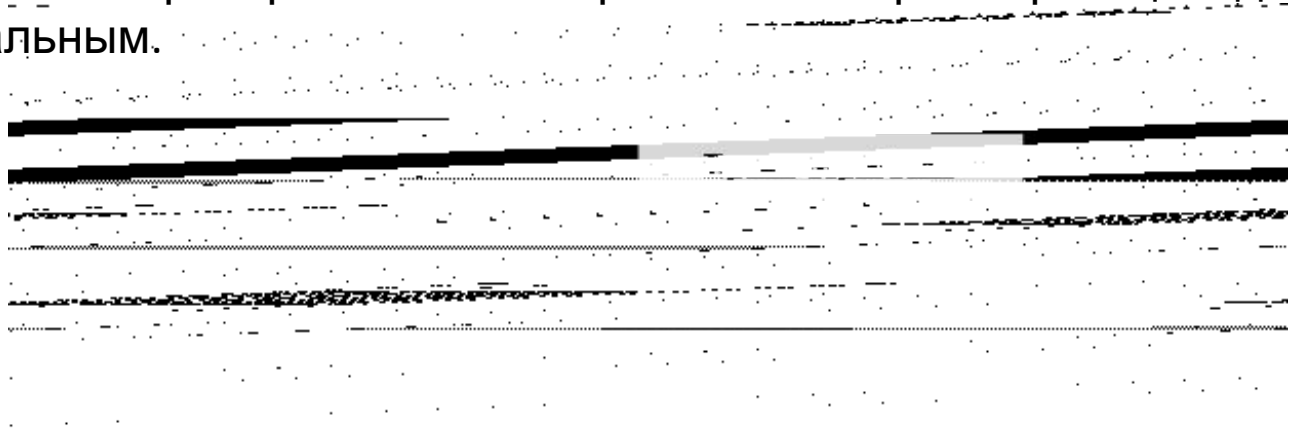
Говорят, что система работает в режиме реального времени, если для описания работы этой системы требуются количественные *временны́е* характеристики.

Характеристика реального времени

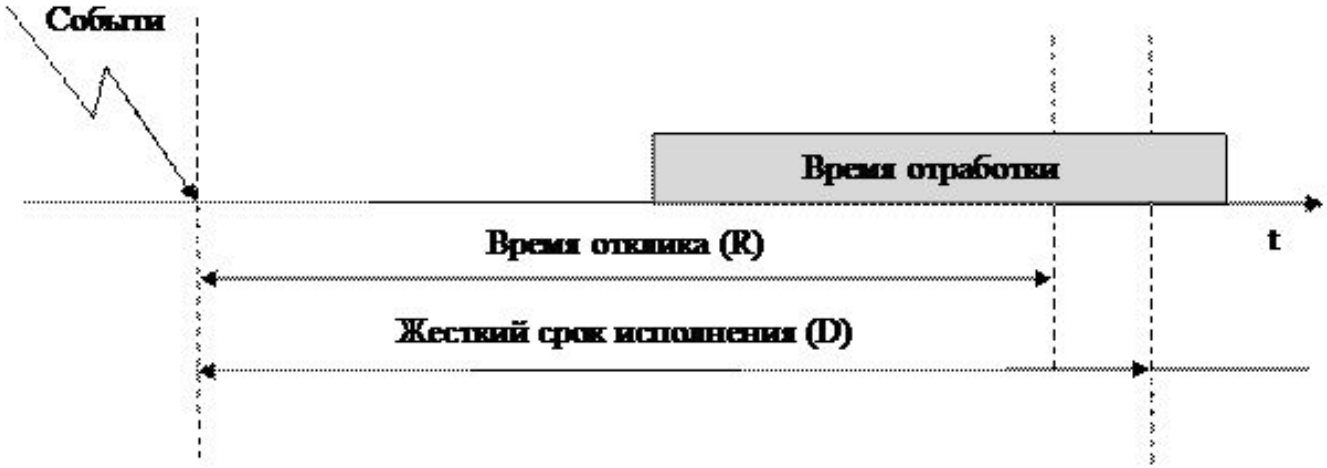


Динамические свойства программ реального времени принято характеризовать тремя определениями:

Жесткое реальное время. Предусматривает наличие гарантированного времени отклика системы на конкретное событие, например, аппаратное прерывание, выдачу команды управления и т.п. Абсолютная величина времени отклика большого значения не имеет и зависит от объекта управления. При практическом применении время реакции должно быть минимальным.



Мягкое реальное время. В этом случае ожидающееся время отклика системы является величиной скорее индикативной, нежели директивной. Предполагается, что в большинстве случаев отклик уложится в заранее заданные пределы, но и задержка в реакции системы некатастрофична.



Интерактивное реальное время. Является скорее психологической, нежели технической характеристикой. Определяет время, в течение которого оператор-человек способен спокойно, без нервозности, ожидать реакции системы на данные им указания.

Последовательное программирование

Программа – это описание объектов (констант и переменных) и операций, совершаемых над ними. Программа – это чистая информация.

Наиболее распространенный способ создания программ – это **последовательное программирование.** Оно подразумевает, что операторы (команды) программы выполняются в определённой, заранее известной последовательности (последовательный алгоритм).

Целью последовательной программы является **преобразование входных данных**, заданных в определенной форме, **в выходные данные**, имеющие другую форму, **в соответствии с некоторым алгоритмом** – методом решения.

Рисунок 4 – Обработка данных последовательной программой

Последовательная программа работает как фильтр для исходных данных.

Результат (выходные данные) полностью определяются входными данными и алгоритмом их обработки (программа). Временны'е показатели достижения результата выполнения алгоритма играют второстепенную роль.

Результат не зависит ни **от инструментальных** (определяют усилия и время, затраченные на разработку и характеристики исполняемого кода) ни **аппаратных средств** (определяют скорость выполнения программы).**В любом случае результаты (выходные данные) будут одинаковыми.**

Параллельное программирование

Параллельный алгоритм – алгоритм, который может быть реализован по частям на одном или на множестве различных вычислительных устройств связанных вычислительной сетью с последующим объединением полученных результатов и получением корректного результата.

Параллельное программирование – *метод разработки* **независимых программных модулей** для решения задач, которые исполняются (активны) на одном и том же временном интервале, то есть работают параллельно, при этом каждая задача выполняет свои специфические функции, а отдельные программные модули взаимодействуют между собой.

Отличительные этапы параллельного программирования:

- Выявление параллелизма: анализ задачи с целью выделить подзадачи, которые могут выполняться одновременно.
- Определение параллелизма: изменение структуры задачи таким образом, чтобы можно было эффективно выполнять подзадачи. Для этого часто требуется найти зависимости между подзадачами и организовать исходный код так, чтобы ими можно было эффективно управлять.
- Выражение параллелизма: реализация параллельного алгоритма в исходном коде с помощью системы обозначений параллельного программирования.

В большинстве случаев применение обычных приемов последовательного программирования не позволяет построить систему реального времени.

Программирование для системы реального времени сильно отличается от последовательного программирования. Необходимо постоянно иметь в виду условия (окружающую среду), в которой работает программа.

В *системах реального времени* (СРВ) внешние сигналы, как правило, требуют немедленной (или фиксированной по времени) реакции процессора.

Одна из наиболее важных особенностей СРВ является **конкретное, чётко определённое время реакции** на входные сигналы, которое должно удовлетворять заданным ограничениям.

Время реакции системы на внешние события

ОСРВ должна обеспечить требуемый уровень сервиса за заданный интервал времени. Этот интервал времени задается обычно периодичностью и скоростью процессов, которым управляет система и не зависит от разработчика программы.

Время реакции системы на события определяется **от** поступления сигнала события на объекте и **до** исполнения последней инструкции в программе обработки этого события и является требуемым интервалом времени. Проектируя систему реального времени, необходимо уметь вычислять этот интервал.

Поэтому к программе для СРВ предъявляются специальные требования (с заданной скоростью реагировать на внешние события (запросы)), которые нельзя адекватно реализовать с помощью обычных приемов последовательного программирования.

Поэтому в системах реального времени, чаще всего используются, именно

Программируемые промышленные контроллеры.

Программируемые логические контроллеры

Программируемые логические контроллеры — это специальные микрокомпьютеры, предназначенные для выполнения операций исполнения управляющего алгоритма в промышленных условиях.

ПЛК генерирует выходные сигналы "включить/выключил," для управления исполнительными механизмами — электродвигателями, клапанами лампочками и т.п., которые являются неотъемлемой частью систем автоматизации во всех отраслях промышленности.

Основные операции ПЛК соответствуют комбинационному управлению логическими схемами. Кроме того, современные ПЛК могут выполнять другие операции, например функции счетчика и интервального таймера, обрабатывать задержку сигналов и т. д. Более сложные ПЛК обрабатывают аналоговые сигналы, производят математические операции и даже содержат контуры управления обратной связи, как ПИД-регуляторы,

Основное преимущество ПЛК заключается в том, что одиночная компактная схема может заменить сотни реле.

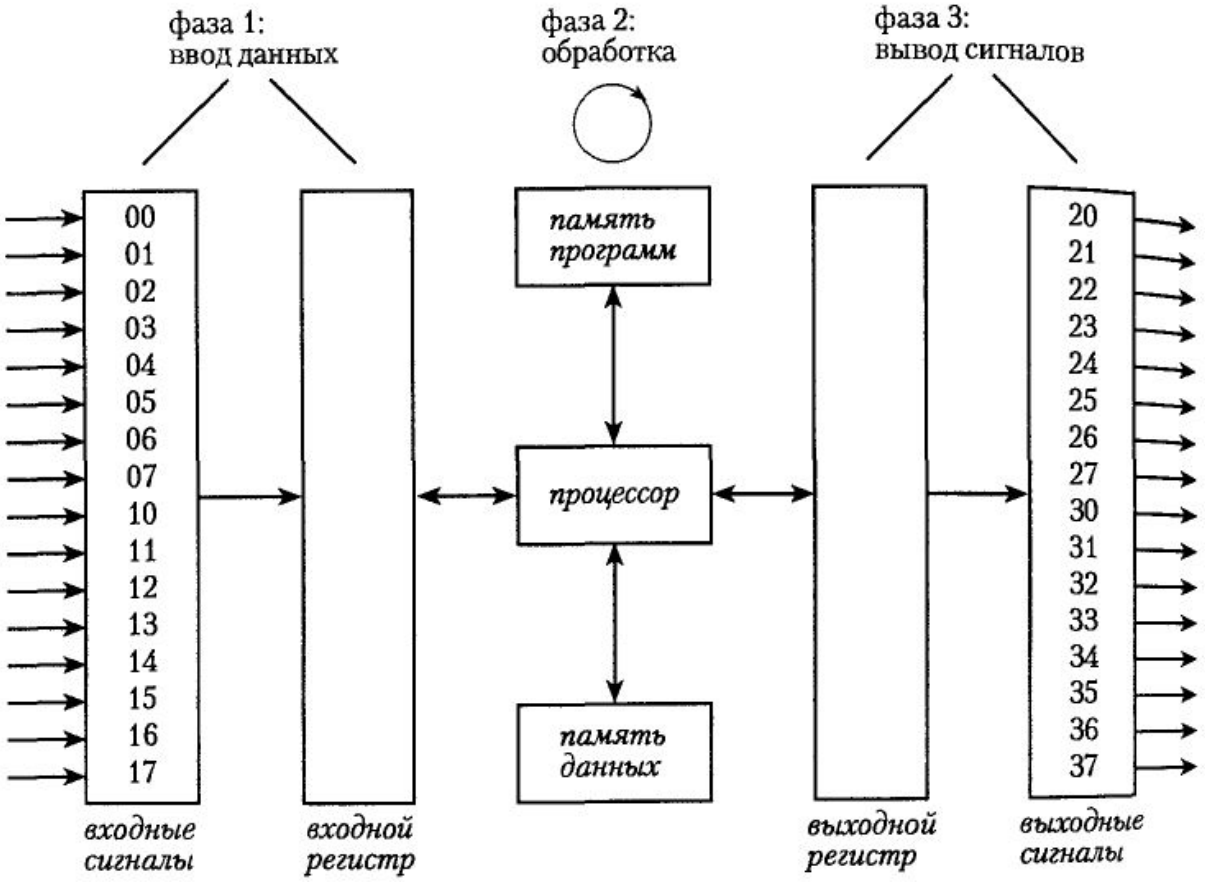
Другое преимущество — функции ПЛК реализуются программно, а не аппаратно, поэтому его поведение можно изменить с минимальными усилиями.

Однако **необходимо учитывать**, что ПЛК могут быть медленнее, чем релейная аппаратная логика.

Оптимальное решение для каждого конкретного приложения **можно получить**, **применяя обе технологии в одной системе** так, чтобы использовать преимущества каждой из них.

Конструктивно ПЛК обычно приспособлены для работы в типовых промышленных условиях, с учетом уровней сигналов, термо- и влагостойкости, ненадежности источников питания, механических ударов и вибраций, ПЛК также содержат специальные интерфейсы для согласования и предварительной обработки различных типов и уровней сигналов. Функции ПЛК все чаще применяются в устройствах ввода/ вывода, входящих в состав больших интегрированных систем управления.

ПЛК можно программировать различными способами – как с помощью ассемблероподобных команд, так и с помощью проблемно-ориентированных языков высокого уровня или прямым описанием операций последовательностного управления с помощью функциональных блоков с графическими символами логических элементов или символов принципиальных схем В настоящее время эти методы заменяются BASIC подобными языками программирования



Основная структура программируемого логического контроллера (цифрами обозначены различные входные и выходные каналы)

Устройства сопряжения с объектами.

Неотъемлемой частью любой автоматизированной системы управления технологическими процессами являются **устройства связи с объектом** (**УСО**).

УСО предназначено для сопряжения измерительной (датчиковой) аппаратуры и исполнительных механизмов контролируемого объекта и/или технологического процесса с вычислительными средствами системы.

УСО представляет собой комплекс конструктивных и электронных средств в виде специализированных функциональных блоков, осуществляющий необходимый информационный обмен между технологическим объектом и управляющей информационной системой.

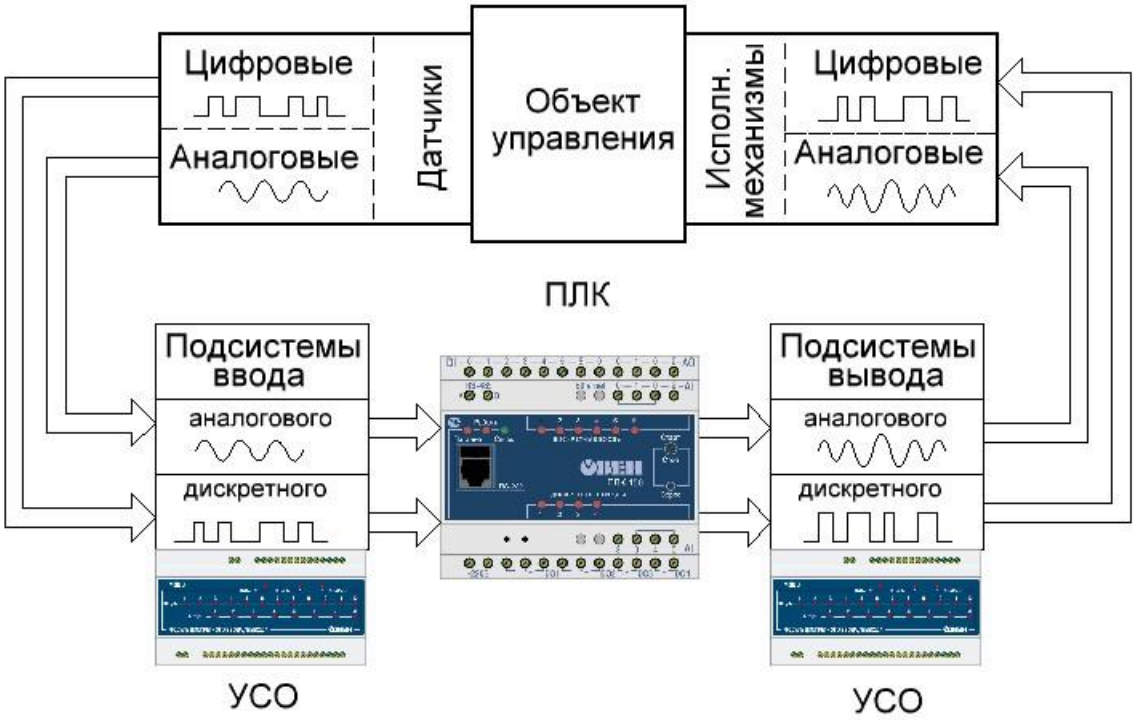


Рисунок 1 – Использование УСО в структуре системы автоматизации

Функции УСО

- **Нормализация аналогового сигнала** – приведение границ шкалы первичного непрерывного сигнала к одному из стандартных диапазонов входного сигнала аналого-цифрового преобразователя измерительного канала. Наиболее распространены следующие диапазоны: по току 0÷5 мА, 0÷20 мА, 4÷20 мА, или по напряжению – от 0 до 5 В; от –5 до +5В; от 0 до 10В. а также сигналы датчиков с естественными выходными сигналами (термопары, термометры сопротивления, тензометрические датчики и т.п.).
- **Предварительная низкочастотная фильтрация аналогового сигнала** – ограничение спектра частот первичного сигнала с целью снижения влияния на результаты измеренных помех различного происхождения. На промышленных объектах наиболее распространены периодические помехи с частотой сети переменного тока, а также хаотические импульсные помехи, вызванные влиянием на технические средства измерительного канала переходных процессов и наводок при коммутации исполнительных механизмов повышенной мощности.
- **Обеспечение гальванической развязки** между источником аналогового или дискретного сигнала и измерительными каналами системы. В равной степени это относится к изоляции между каналами дискретного вывода системы и управляемым силовым оборудованием. Помимо собственно защиты выходных и входных цепей, гальваническая изоляция позволяет снизить влияние на систему помех по цепям заземления за счет полного разделения вычислительной системы и контролируемого оборудования.

Устройства (модули) ввода-вывода или УСО являются передаточным звеном между процессором ПЛК и техническим процессом.

Измерительные каналы всегда имеют ограниченную пропускную способность, поэтому измеренные значения поступают в процессор в определённые дискретные моменты времени.

Современный модуль УСО обычно имеет свой собственный встроенный микроконтроллер. Поэтому такой модуль часто называют *интеллектуальным*. Он выполняет *циклический опрос* всех своих измерительных каналов и помещает полученные данные в *буфер данных*. При поступлении в модуль команды считывания значений со входов собранные данные передаются из буфера данных модуля в ПЛК, где помещаются в буфер OPC сервера или в определенную область ОЗУ.

Модули УСО в системах промышленной автоматизации обязательно имеют *гальваническую изоляцию* между входными (выходными) зажимами и шиной контроллера. Напряжение изоляции обычно составляет от 2500 В (реже от 500 В) до 15000 В.

Опрос модулей УСО может выполняться как циклически с одинаковой частотой для всех модулей, так и с разной частотой.

Циклический опрос всех модулей УСО с заранее заданной частотой сильно загружает шину по которой модули ввода связываются с процессором.

Опрос с разной частотой позволяет уменьшить загруженность канала (шины) передачи данных, по которой выполняется обмен данными между модулями УСО и процессорным модулем.

Цифровые коммуникации в системах управления техническими процессами. Основные протоколы обмена в системах промышленной автоматизации.

Иерархическая структура технического процесса

В большинстве процессов можно выделить несколько иерархических или административных уровней. Они более или менее соответствуют различным решениям которые должны приниматься для управления процессом.

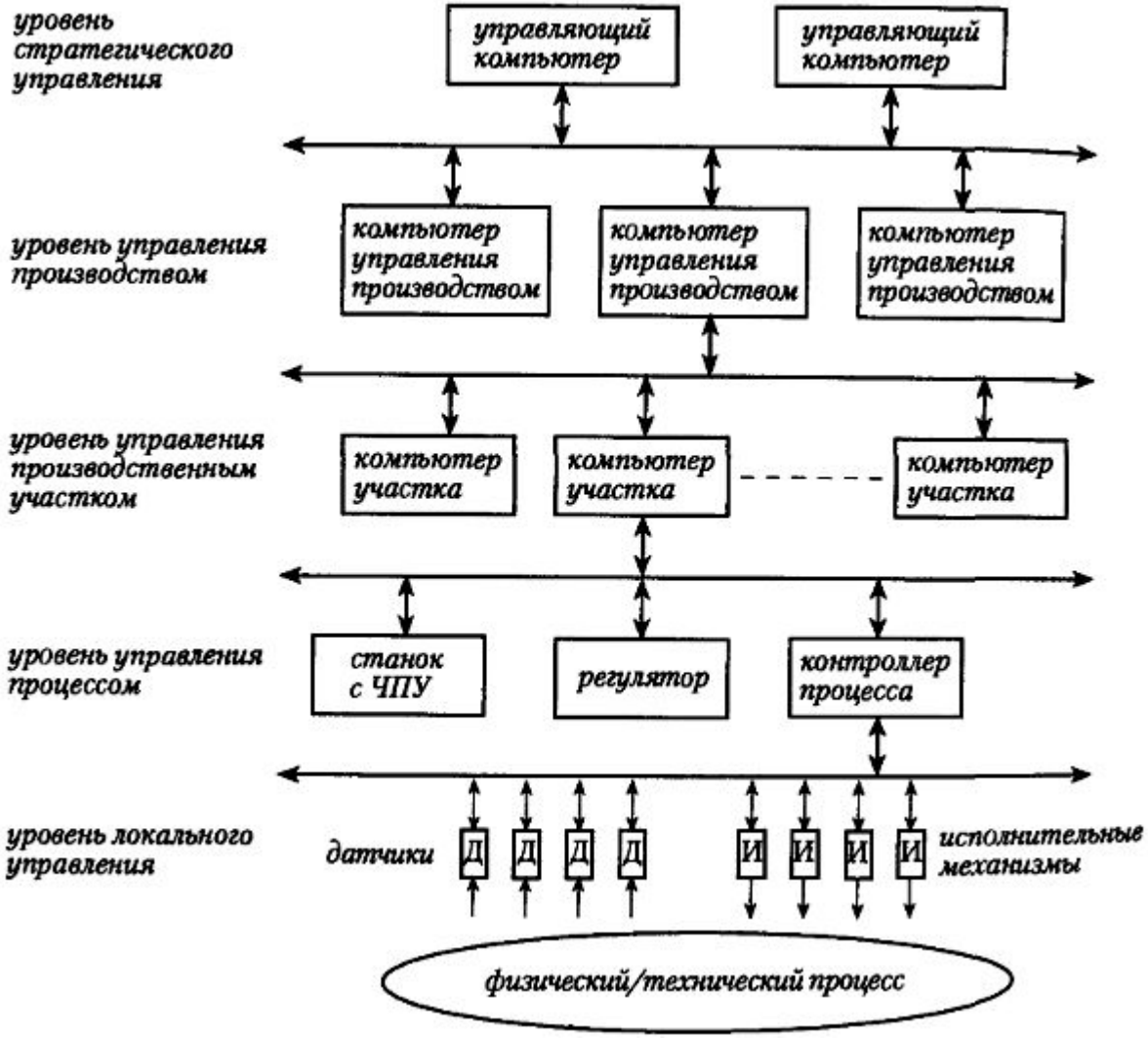
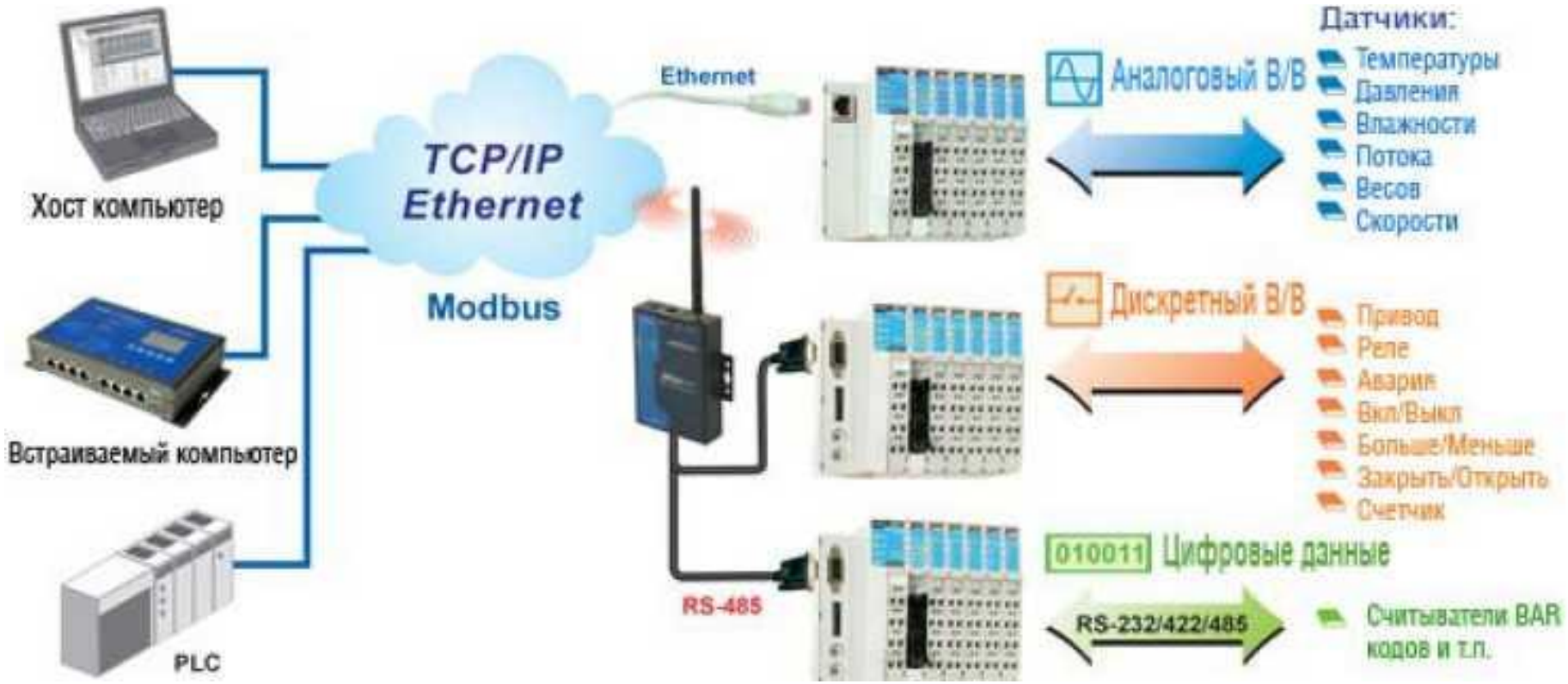


Рис. 1. Иерархическая структура распределенной системы управления

Распределённая система



В **распределенных системах** модули или даже отдельные входы-выходы, образующие единую систему управления, могут быть разнесены на значительные расстояния

Иерархическая модель позволяет определить объёмы потоков информации на разных уровнях управления.

В общем случае все объекты, расположенные на одинаковых уровнях иерархии, интенсивно обмениваются информацией между собой; обмен данными между уровнями обычно менее интенсивен и не критичен ко времени.

Предприятие может рассматриваться как строго упорядоченная система реального времени, в которой информация на каждом уровне должна обрабатываться с соответствующей скоростью.

В таблице показаны типичные объемы информации, частота ее обновления и время реакции для нужд управления на разных уровнях руководства предприятием с развитыми техническими и организационными функциями.

Уровень управления	Объем данных	Время реакции	Частота обновления
Стратегическое управление	Мбайты	дни	дни
Управление производством	Кбайты	часы, минуты, секунды	часы, минуты, секунды
Управление участком	байты	Доли секунды (100 мс)	секунды
Управление процессом	биты	миллисекунды	миллисекунды
Локальное управление (датчики, исполнительные механизмы)	биты	миллисекунды	миллисекунды

Основные методы сбора данных от датчиков и передча их от местных (локальных) регуляторов к более высоким в иерархии устройствам и в центр управления.

Первый метод. Телеметрия – это предпочтительный способ передачи данных от периферийных устройств к центральному в случае, когда квитирование или двунаправленная передача неудобны или вообще невозможны (например, в случае космических объектов).

Все данные передаются непрерывно в заранее определенном формате. После завершения одного цикла передачи начинается новый.

Каждый параметр определяется его положением в потоке данных.

Второй метод. Сбор данных выполняется по опросу. Опрос — это типовой метод, который в основном используется периферийными процессорами для получения информации от датчиков, однако иногда он применяется и центральными процессорами для обновления своих баз данных.

Опрос применяется на уровне управления процессом. Управляющий компьютер циклически опрашивает текущее состояние датчиков и периодически обновляет данные в своей внутренней базе данных. При опросе периферийные устройства должны отвечать управляющему компьютеру, и таким образом гарантируется периодическое обновление базы данных.

Третий метод. Метод основан на прерываниях, которые генерируются датчиками, когда они должны передавать информацию. Метод заключается в передаче только тех переменных, которые изменили значение по сравнению с предыдущим циклом. Цифровые переменные передаются при каждом изменении, адля аналоговых переменных задается определенная переходная зона.

Новая информация поступает к центральному процессору только в том случае, когда аналоговая переменная изменяется на определенный процент по отношению к предыдущему переданному значению.

Цифровые сети

К наиболее важным способам передачи цифровых данных относятся:

- коммутация цепей (устройств);
- выделенные линии;
- коммутация сообщений;
- коммутация пакетов.

Коммутация цепей (устройств) – это непосредственное соединение двух устройств с помощью физического тракта передачи, например модемов через телефонную сеть. Название происходит от старых декадно-шаговых АТС при установке соединения происходит замыкание контактов соответствующих реле, после него возникает прямой физический тракт передачи между двумя оконечными устройствами.

Выделенная линия – это постоянно установленное соединение между двумя определенными точками. Для передачи данных используются обычные аналоговые линии повышенного качества, т. е. специально протестированные и экранированные, чтобы обеспечивать большую полосу пропускания, отношение сигнал/шум и, соответственно, пропускную способность.

Коммутация сообщений. Цифровое сообщение целиком отправляется в сеть и передается от узла к узлу, пока не достигнет конечного пункта, при этом используется метод передачи данных с промежуточным хранением. Этот способ применяется при передаче телексных сообщений и сообщений электронной почты и практически не представляет интереса в задачах промышленной автоматизации.
Особым типом сети коммутации сообщений является Интернет, который является исключительно гибкой сетью где используются все возможные комбинации цифровых сетей. В основном применяется протокол TCP/IP.
Однако Интернет не дает ни гарантий какого-либо определенного качества работы, ни того, что сообщение достигнет адресата или что это произойдет за определенное время.

Поэтому Интернет нельзя рассматривать в качестве надежного средства передачи информации для производственной системы реального времени.

Коммутации пакетов — это современное состояние в области цифровых коммуникаций, устанавливающая виртуальный канал связи между узлами. Сообщение от передатчика делится на пакеты ограниченной длины. Каждый пакет содержит протокольную информацию, в частности, последовательный номер пакета и адрес получателя.

Пакеты направляются к адресату по виртуальным каналам.

Маршрут каждого пакета вырабатывается независимо, поэтому пакеты, предназначенные одному адресату, могут передаются по разным физическим каналам.

Пакеты от различных пользователей *мультиплексируются* в магистральных каналах, поэтому пропускная способность сети используется более эффективно, чем при выделенных линиях или коммутации цепей.

Загрузка сети становится более равномерной, поскольку каждый абонент не использует постоянно всю емкость канала связи, а спорадически передает значительные объемы данных в течение коротких промежутков времени.

Другим важным достоинством коммутации пакетов является возможность прозрачного для пользователя изменения маршрута при выходе из строя промежуточного узла или канала связи. Благодаря этим свойствам, сети с коммутацией пакетов обычно имеют хороший показатель готовности.

Основные протоколы обмена в системах промышленной автоматизации

В промышленных системах наибольшая часть работы (и стоимости) по сбору и обработке данных связана не с центральным вычислительным комплексом, а относится к локальному уровню, где установлены основные устройства.

Шины локального управления (Fieldbus)

Стандарт шин локального управления должен обеспечить взаимодействие различных устройств, присоединенных к одной и той же физической среде передачи.
Очевидным преимуществом цифровой техники по сравнению с аналоговой является заметное уменьшение кабельных связей – один цифровой канал может заменить большое число проводников с токами в диапазоне 4-20 мА.

Применение шин локального управления дает значительные преимущества.

Существенная доля "интеллекта", необходимого для управления процессом, переносится і локальный уровень.

Обслуживание датчиков значительно облегчается, поскольку такие операции, как тестирование и калибровка датчиков, можно выполнять дистанционно и без непосредственного участия наладчиков.

Естественно, что качество управления находится в прямой связи с достоверностью и качеством собираемых данных

Шина Bitbus

Шина Bitbus разработана Intel в 1984 году. Соответствует двум первым уровням **Модели Взаимодействия Открытых Систем** (Open Systems Interconnection, OSI) – физическому (1-й уровень) использует витую пару в соответствии со стандартом RS-485 и канальному (2-й уровень).

К одной шине можно подключить до 28 устройств, а несколько шин можно соединить с помощью повторителей. Возможные скорости передачи – 62.5, 375 Кбит/с и 2.4 Мбит/с. При наименьшей скорости (62.5 Кбит/с) допустимое расстояние между повторителями составляет 1200 м.

Шина Bitbus иерархически структурирована. Одно из присоединенных устройств является ведущим, остальные – ведомыми.

Ведущее устройство всегда управляет процессом передачи сообщений с помощью механизма опроса: ведущий посылает запрос ведомому, который должен ответить; ведомое устройство не может передать сообщение по своей инициативе, а должно ждать запроса ведущего. Когда несколько таких шин объединены в сеть, ведущее устройство служит и сетевым интерфейсом.

Протокол Bitbus не предусматривает несколько ведущих устройств и поэтому не определяет способ передачи прав ведущего. В обычной практике устройство, например управляющий компьютер, является ведущим, а все остальные устройства с менее сложной электроникой – ведомыми.

Шина Modbus

Modbus — коммуникационный протокол, основанный на архитектуре «клиент-сервер». Широко применяется в промышленности для организации связи между электронными устройствами. Может использовать для передачи данных последовательные линии связи RS-232, RS-485, RS-422, а также сети TCP/IP (Modbus TCP).
Modbus относится к **протокол**ам **прикладного уровня** сетевой модели **OSI**.

Контроллеры на шине Modbus взаимодействуют, используя **клиент-серверную модель**, основанную на транзакциях, состоящих из **запрос**а и **ответ**а.

Обычно **в сети** есть только **один сервер** – главное устройство (*master*), и **несколько клиентов** – подчиненных (*slaves*) устройств.

Главное устройство инициирует запросы. Подчиненные устройства передают запрашиваемые главным устройством данные, или производят запрашиваемые действия.

Главное устройство может адресоваться к конкретному подчиненному устройству или инициировать передачу широковещательного сообщения для всех подчиненных устройств.
Подчиненное устройство формирует сообщение и возвращает его в ответ на запрос, адресованный именно ему.

При получении широковещательного запроса ответное сообщение подчинёнными устройствами не формируется.

Спецификация Modbus описывает структуру запросов и ответов. Их основа – *элементарный пакет протокола*, так называемый **PDU** (*Protocol Data Unit*).

Структура PDU не зависит от типа линии связи и включает в себя код (номер) функции и поле данных. Код функции кодируется однобайтовым полем и может принимать значения в диапазоне 1…127. Диапазон значений 128…255 зарезервирован для кодов ошибок. Поле данных может быть переменной длины. Размер пакета PDU ограничен 253 байтами.

Modbus PDU

номер функции	данные
1 байт	N < 253 (байт)

Для передачи пакета по физическим линиям связи PDU помещается в другой пакет, содержащий дополнительные поля. Этот пакет носит название **ADU** (*Application Data Unit*).

Формат ADU зависит от типа линии связи.

Максимальный размер ADU:

- для последовательных сетей RS-232/RS-485 – 256 байт,
- для сетей TCP – 260 байт.

В сетях MODBUS может быть использован один из двух способов передачи:

ASCII или **RTU**.

Пользователь выбирает необходимый режим вместе с другими параметрами (скорость передачи, режим паритета, количество стоповых бит и т.д.) во время конфигурации каждого контроллера.

Режим ASCII. При использовании ASCII-режима каждый байт сообщения передается как два ASCII символа. Например, A5h = 41h, 35h

Главное преимущество: время между передачей символов может быть до 1 сек. без возникновения ошибок при передаче.

Формат каждого байта в ASCII-режиме:

Система кодировки:
Шестнадцатиричная, ASCII-символы 0-9, A-F

Назначение битов:
1 старт бит, 7 бит данных(младшим битом вперед), 1 бит паритета, 1 стоп бит если есть паритет; 2 бита если нет паритета

Контрольная сумма:
Longitudinal Redundancy Check (LRC) (проверка чётности)

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	del

Шина PROFIBUS

Шина PROFIBUS сочетает функции уровней 1, 2 и 7 модели OSI (физический, канальный и приложений).

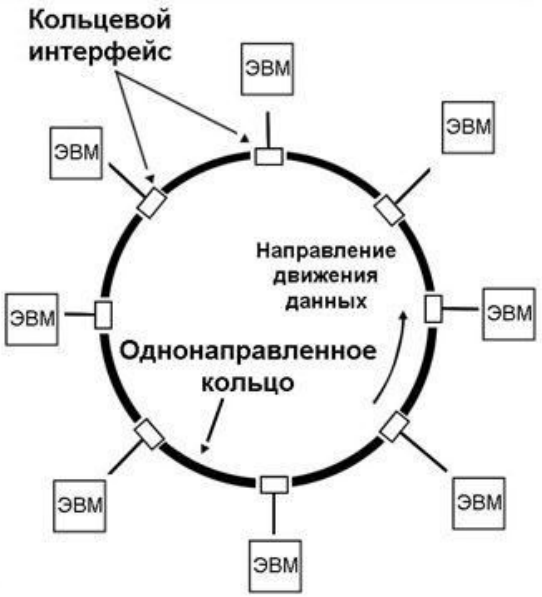
PROFIBUS разработана для поддержки на одной шине интеллектуальных датчиков одновременно с более сложными устройствами (типа ПЛК, регуляторов, небольших управляющих компьютеров и т. д.).

Физическая среда передачи PROFIBUS — это экранированная витая пара в соответствии со спецификацией интерфейса RS-485 с максимальной длиной 1200 м. Скорости передачи данных — 9.6,19.2, и 500 Кбит/с.

PROFIBUS может работать как с одним постоянным ведущим устройством, так и в конфигурации с несколькими ведущими. Обмен происходит при помощи сообщений.

Коммуникации на основе сообщений обеспечивают гибкость в отношении типа и количества данных, которыми обмениваются устройства, однако это достигается за счет быстродействия. PROFIBUS не может гарантировать, что данные от каждого датчика поступают с постоянной скоростью. Соответствует ли это требованиям процесса или нет, нужно решать в каждом конкретном случае.

Управление доступом к среде основано на маркерной шине, включающей ведущую и ведомую станции. К шине можно присоединить до 127 активных и пассивных станций. Между активными станциями циркулирует маркер — станция может передавать информацию, если владеет маркером. Маркер недоступен пассивным станциям — активная станция должна запросить пассивную, после чего она может ответить сообщением.



Принцип работы маркерного кольца

Маркер циркулирует между станциями. Станция может послать сообщение, только если она владеет маркером. Каждая станция распознает сообщения, предназначенные именно ей, и ретранслирует остальные дальше. Сообщение передается до тех пор, пока не достигнет отправителя.

Когда станция, имеющая сообщение для передачи, получает маркер, она удаляет его из кольца и в течение максимально разрешенного времени передает свои пакеты данных. По истечении этого времени маркер передается следующей станции. Сообщения могут передаваться вместо маркера (передатчик не отдает маркер до тех п *у* пока не закончил передачу) или вместе с маркером, при этом маркер может находиться как в начале, так и в конце пакета.

PROFIBUS использует обмен данными **между ведущим и ведомыми** устройствами (протоколы **DP** и **PA**) или **между несколькими ведущими устройствами** (протоколы **FDL** и **FMS**).

Сеть **PROFIBUS** построена в соответствии с многоуровневой моделью OSI.

PROFIBUS определяет следующие уровни:

1. **физический уровень** – отвечает за характеристики физической передачи
2. **канальный уровень** – определяет протокол доступа к шине
3. **уровень приложений** – отвечает за прикладные функции

Семейство **PROFIBUS** состоит из трех совместимых друг с другом версий:

- **PROFIBUS PA**,
- **PROFIBUS DP** и
- **PROFIBUS FMS**.

- **PROFIBUS DP** (*Decentralized Peripheral* – распределенная периферия) – протокол, ориентированный на обеспечение скоростного обмена данными между:
  - системами автоматизации (ведущими DP-устройствами)
  - устройствами распределённого ввода-вывода (ведомыми DP-устройствами).
- **PROFIBUS PA** (*Process Automation* – автоматизация процесса) – протокол обмена данными с оборудованием полевого уровня, расположенным в обычных или взрывоопасных зонах. Протокол отвечает требованиям международного стандарта IEC 61158-2. Позволяет подключать датчики и приводы на одну линейную шину или кольцевую шину.
- **PROFIBUS FMS** (*Fieldbus Message Specification* – спецификация сообщений полевого уровня) – универсальный протокол для решения задач по обмену данными между интеллектуальными сетевыми устройствами (контроллерами, компьютерами/программаторами, системами человеко-машинного интерфейса) на полевом уровне. Некоторый аналог промышленного *Ethernet*, обычно используется для высокоскоростной связи между контроллерами и компьютерами верхнего уровня и

Системы диспетчерского управления и сбора данных

Основная проблема при создании АСУ ТП любой сложности – это как заставить разработчиков программного обеспечения и технологов понимать друг друга.

Поэтому особое место в создании информационных систем принадлежит программному обеспечению, предназначенному для эксплуатации на среднем и верхнем уровнях промышленных информационных систем, например, пультов управления сложными агрегатами.

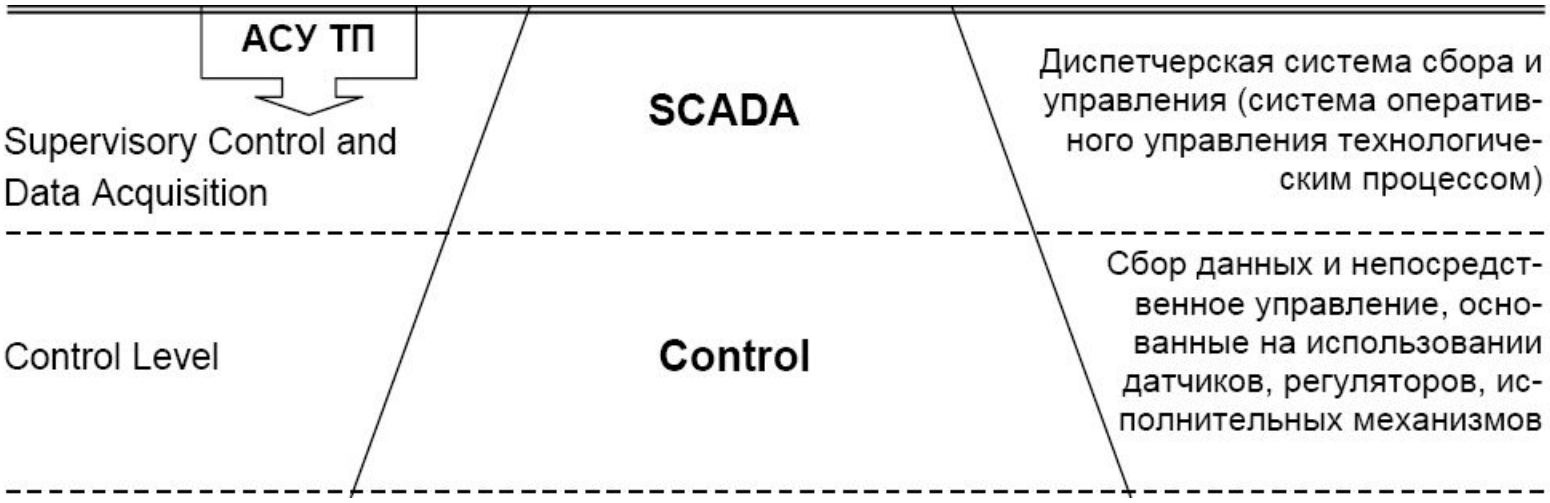


Рисунок 5 – Верхний и средний уровни автоматизированной информационной системы промышленного предприятия

Решением явились создание специализированных программных пакетов:

- **программное обеспечение операторских станций** технологических процессов для оперативного (диспетчерского) управления и сбора данных (Supervisor Control And Data Acquisition – **SCADA).**
- **интерфейс человек-машина** (Human Machine Interface, HMI). Обеспечивает взаимодействие между технологическим оборудованием и человеком с целью выполнения определенных функций, для которых создавалось это технологическое оборудование

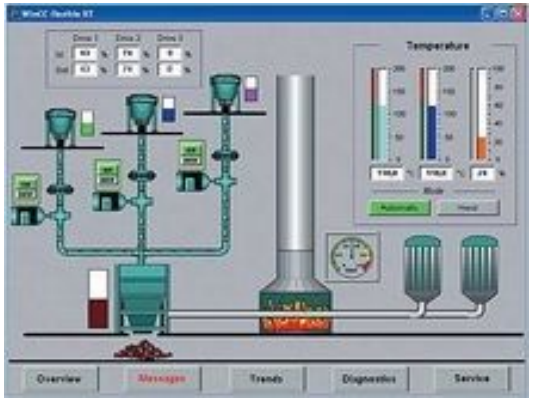
SCADA (Диспетчерское управление и сбор данных) является основным ПО систем промышленной автоматизации и в настоящее время остается одним из наиболее перспективных методов автоматизированного управления сложными динамическими техническими системами. Именно на принципах диспетчерского управления строятся крупные автоматизированные системы управления в промышленности.

Граница между программным обеспечением класса SCADA и HMI весьма условна, поэтому многие системы SCADA могут работать в качестве HMI интерфейса для систем нижнего уровня.

Главными функциями таких SCADA-программ являются отображение технологического процесса в виде мнемосхем, сигнализация об аварийных ситуациях, обеспечение общего управления процессом со стороны оператора-технолога и т.д.

SCADA – осуществляет процесс сбора и отображения информации в реальном времени с удаленных точек (объектов) для обработки, анализа и возможного управления удаленными объектами.

Требование обработки реального времени обусловлено необходимостью доставки (выдачи) всех необходимых сообщений и данных на центральный интерфейс диспетчера.



Основные требования к SCADA-системам:

- надежность системы (технологическая и функциональная);
- безопасность управления:
  - никакой единичный отказ оборудования не должен вызвать выдачу ложного выходного воздействия (команды) на объект управления;
  - никакая единичная ошибка оператора не должна вызвать выдачу ложного выходного воздействия (команды) на объект управления;
  - все операции по управлению должны быть интуитивно-понятными и удобными для оператора (диспетчера);
- точность обработки и представления данных;
- простота расширения системы.

Функции человека-оператора в системе диспетчерского управления представляют собой набор вложенных циклов, в которых оператор;

- планирует, какие следующие действия необходимо выполнить;
- обучает (программирует) компьютерную систему на последующие действия;
- отслеживает результаты (полу)автоматической работы системы;
- вмешивается в процесс в случае критических событий, когда автоматика не может справиться, либо при необходимости подстройки (регулировки) параметров процесса;
- обучается в процессе работы (получает опыт).

Особенности SCADA как процесса управления

Особенности процесса управления в современных диспетчерских системах:

- процесс SCADA применяется системах, в которых обязательно наличие человека (оператора, диспетчера);
- процесс SCADA был разработан для систем, в которых любое неправильное воздействие может привести к отказу (потере) объекта управления или даже катастрофическим последствиям;
- оператор несет, как правило, общую ответственность за управление системой, которая, при нормальных условиях, только изредка требует подстройки параметров для достижения оптимальной производительности;
- активное участие оператора в процессе управления происходит нечасто, но в непредсказуемые моменты времени, обычно в случае наступления критических событий (отказы, нештатные ситуации и пр.);
- действия оператора в критических ситуациях могут быть жестко ограничены по времени (несколькими минутами или даже секундами)

Все современные SCADA-системы (рисунок 6) включают три основных структурных компонента.

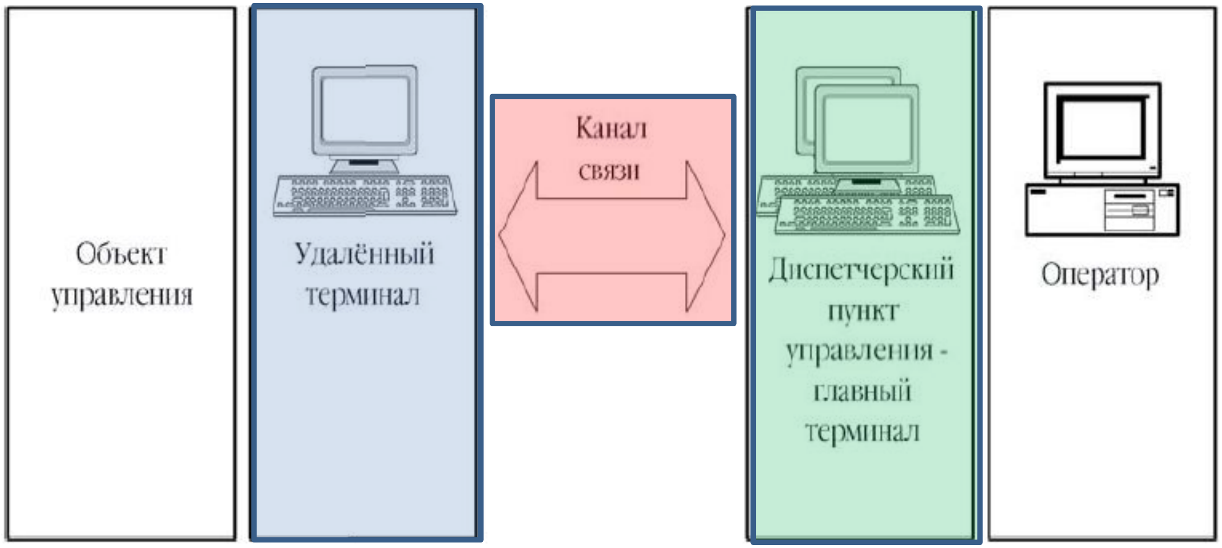


Рисунок 6 – Основные структурные компоненты SCADA-систем

Удаленный терминал (*Remote Terminal Unit, RTU*) осуществляет обработку задачи в режиме реального времени. Вариантов его реализаций может быть много – от простейших интеллектуальных датчиков, осуществляющих съем информации с объекта, до специализированных многопроцессорных отказоустойчивых вычислительных комплексов, осуществляющих обработку информации и управление в режиме жесткого реального времени. Использование устройств низкоуровневой обработки информации позволяет снизить требования к пропускной способности каналов связи с центральным диспетчерским пунктом.

В современной SCADA-системе число контролируемых удаленных данных может достигать 100 тысяч.

Коммуникационная система (*каналы связи*) (*Communication System, CS*) необходима для передачи данных с удаленных точек (объектов, терминалов) на центральный интерфейс оператора-диспетчера и передачи сигналов управления на удаленный терминал (объект) в зависимости от конкретного исполнения.

Диспетчерский пункт управления – *главный терминал* (*Master Terminal Unit, MTU*) осуществляет обработку данных и управление высокого уровня, как правило, в режиме мягкого реального времени. Одна из основных функций – обеспечение интерфейса между человеком-оператором и системой. В зависимости от конкретной системы диспетчерский пункт управления может быть реализован в самом разнообразном виде – от единичного компьютера с дополнительными устройствами подключения к каналам связи до больших вычислительных систем или объединенных в локальную сеть рабочих станций и серверов.

Основные функциональные компоненты систем диспетчерского управления и сбора данных (рисунок 7):

- человек-оператор;
- компьютер для взаимодействия с человеком;
- компьютер для взаимодействия с задачей (объектом):
- задача (объект управления).

Рисунок 7 – Функциональные компоненты SCADA-систем

Современные SCADA-системы обеспечивают:

- **обмен данными с ПЛК** и **УСО** (устройства связи с объектом, то есть с промышленными контроллерами и платами ввода/вывода) в реальном времени через драйверы;
- **обработка информации в реальном времени**;
- **наглядное отображение** процесса производства;
- **накопление**, **хранение** и **анализ** полученных данных;
- **аварийная сигнализация** и управление тревожными сообщениями – определяют критические ситуации и производят оповещение персонала по каналам телефонной и радиосети;
- **создание сценариев управления**,
- **подготовка и генерирование отчетов** о ходе технологического процесса – формируют данные для анализа экономических характеристик производства.

SCADA-системы позволяют разрабатывать АСУ ТП в клиент-серверной или в распределенной архитектуре.

Для создания рабочего места оператора технологического процесса обычно необходимо реализовать следующий набор функций:

- **органы управления** различных типов;
- **экранные формы отображения параметров** процесса типа стрелочных, полосковых или цифровых индикаторов, а также сигнализирующие табло различной формы и содержания;
- **многооконный графический интерфейс** и другие функции;
- возможность создания **архив**ов **аварий**, **событий** и **поведения переменных процесса** во времени (так называемые *тренды*), а также полное или выборочное хранение параметров процесса через заданные промежутки времени постоянно или по условию;
- **язык для реализации алгоритмов управления**, математических и логических вычислений;
- **средства документирования** как самого алгоритма, так и технологического процесса;
- **драйверы** для обслуживания оборудования нижнего уровня АСУ ТП;
- **сетевые функции**;
- **средства защиты** от несанкционированного доступа в систему.

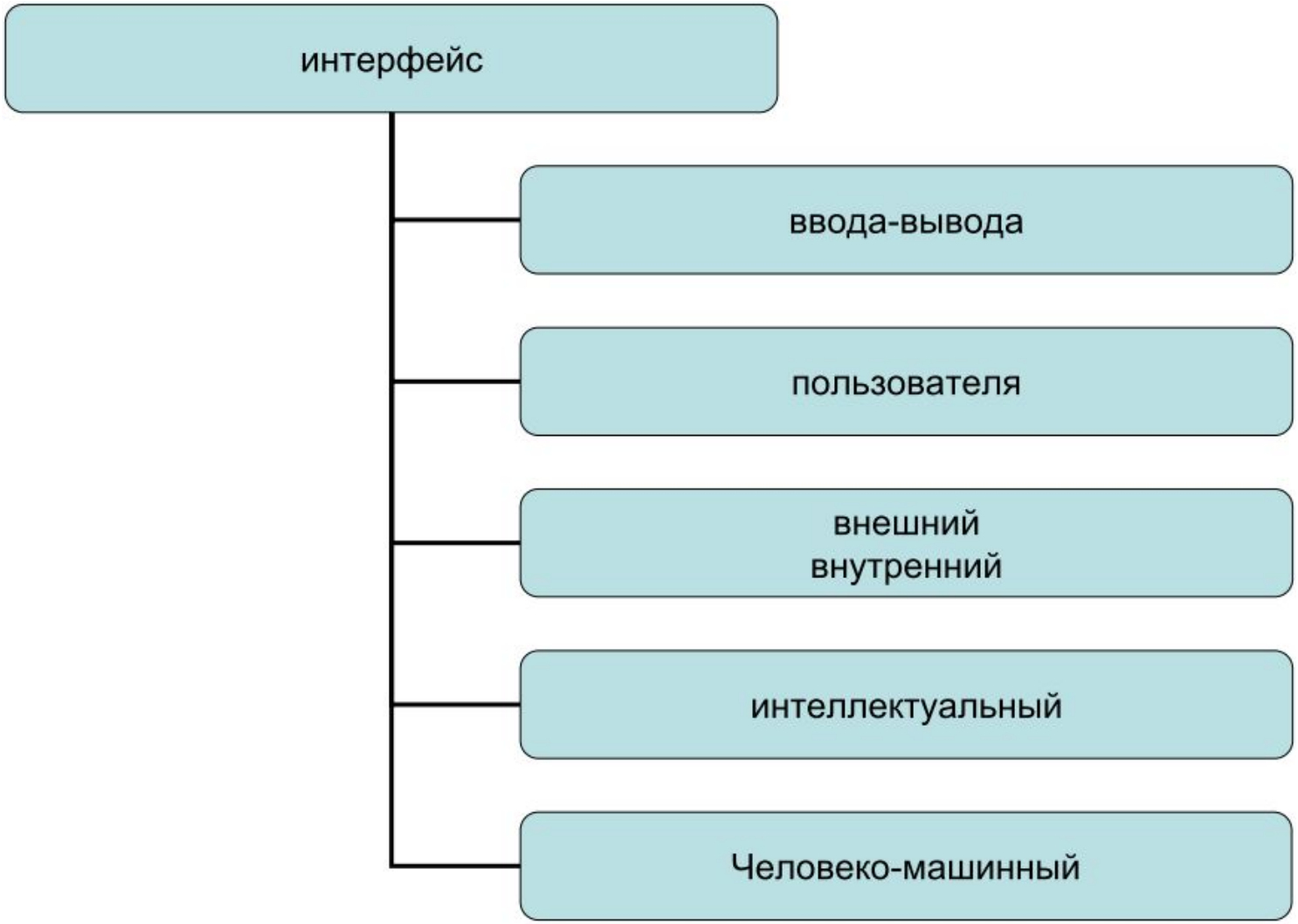
Понятие Человеко-машинного интерфейса его значение для систем управления

Основные понятия

Интерфейс (англ. *interface* – сопряжение, поверхность раздела, перегородка) — *совокупность возможностей, способов и методов взаимодействия двух систем.*

Человеко-машинный интерфейс (ЧМИ) – *это методы и средства обеспечения непосредственного взаимодействия между оператором и технической системой, представляющих возможности оператору управлять этой системой и контролировать ее работу.*

Интерфейс пользователя (ИП), он же **пользовательский интерфейс** (UI — англ. *user interface*) — разновидность интерфейсов, в котором одна сторона представлена человеком (пользователем), другая — машиной/устройством.
Пользовательский интерфейс *представляет собой совокупность средств и методов, при помощи которых пользователь взаимодействует с различными, чаще всего сложными, машинами, устройствами и аппаратурой*.
Обычно именно этот термин используется по отношению к взаимодействию между оператором ЭВМ и программным обеспечением, с которым он работает.


интерфейс
ввода-вывода
пользователя
внешний
внутренний
интеллектуальный
Человеко-машинный

Человеко-машинный интерфейс как элемент системы управления

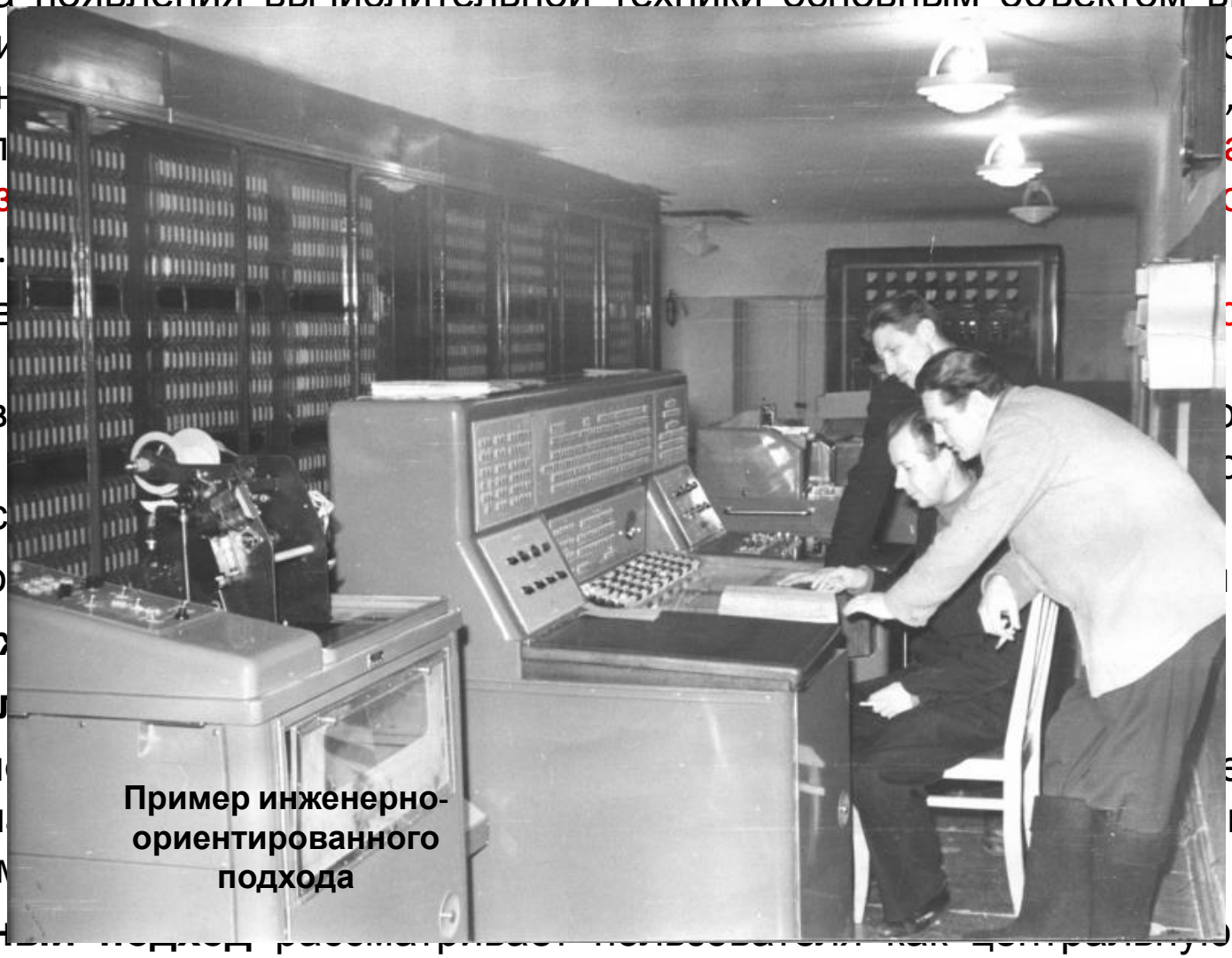
С момента появления вычислительной техники основным объектом внимания были сами
о, чтобы
эффективн
, а не на
то, чтобы п
алисты с
глубоким з
огли ими
управлять.

По мере в
о трудно
управлять
подбора
соответств
о том, как
поставить
ствовала
возможнос

Сформиро

1) **Инж**
2) **Чел**

Инженерн
ерфейса
основан н
подобно
самому ком



Пример инженерно-ориентированного подхода

Когнитивн фигуру
процесса взаимодействия с системой. Ориентация на характеристики пользователя, исследование перцептивных и когнитивных возможностей и

Интерфейс пользователя – это важнейший элемент любой технической системы.

Назначение человеко-машинного интерфейса – обеспечить обмен информацией между оператором (пользователем) и технической системой.

Хорошо организованный интерфейс делает рабочую обстановку более комфортной помогает уменьшить число ошибок оператора и таким образом ограничивает возможный ущерб от неправильных действий для управляемой системы.

Хороший интерфейс (то есть дружественный к человеку-оператору) даёт возможность интуитивно понять суть (то есть функции) технической системы.

Разработка современного интерфейса базируется на представлениях науки *эргономики*.

Эргономика – *наука о приспособлении должностных обязанностей, рабочих мест, оборудования и компьютерных программ для наиболее безопасного и эффективного труда работника, исходя из физических и психических особенностей человеческого организма.*

Эргономика посвящена **эффективному использованию человеческих способностей в технической среде.**

Эргономика – междисциплинарная наука, которая **объединяет знания** из области **техники, физиологии и психологии.**
Эргономика в первую очередь **указывает на то**, **что не нужно делать чтобы не перейти границ обычного человеческого восприятия.**

Психологические модели восприятия

Психология — **наука о человеческом поведении, опыте и о соответствующих мыслительных процессах.** Многие результаты, полученные в ходе психологических исследований, напрямую используются при разработке **интерфейса пользователя**.

Модель человеческого поведения

Модель человеческого поведения дает упрощенную основу для описания взаимодействия между человеком и окружающей средой.

Модель человеческого действия с точки зрения **принятия решений** и **поведения** можно разделить на **три уровня:**

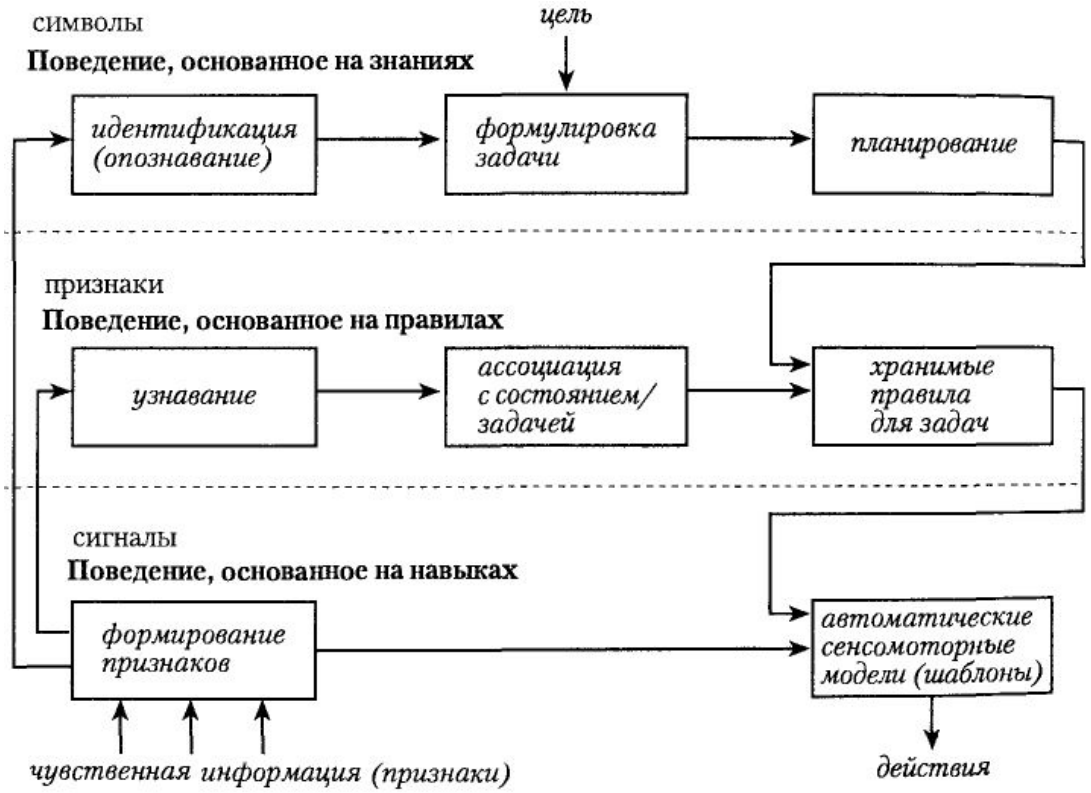


Рисунок 1 – Упрощенная иллюстрация уровней человеческого поведения. Модель действия

1. **Действия на уровне навыков** (на самом нижнем уровне) являются наиболее эффективными, поскольку не требуют *обдумывания* в явном виде и реакция следует сразу за раздражителем.

2. **Действия на уровне правил** выбираются из нескольких хранящихся в памяти образцов и, следовательно, **выполняются с некоторой задержкой**.

3. **Действия на уровне знаний. Для того чтобы определить новое действие в необычной ситуации**, необходимо аналитическое мышление и сравнение с имеющимися знаниями и опытом, что обычно **требует большего времени и значительных умственных усилий.**

Модели поведения могут оказать заметную помощь при создании интерфейса оператора для управления процессом.

Разделение поведения на три категории — основанное **на навыках**, **правилах** и **знаниях** — облегчает классификацию задач и определяет, какой тип поддержки пользователя должен обеспечивать интерфейс на каждом уровне.

Общие принципы проектирования интерфейса пользователя

Три основных принципа справедливы для любого прикладного или функционального проектирования и, следовательно, для **интерфейсов пользователя:**

- **простота**,
- **наглядность,**
- **последовательность**.

Простота — **наиболее важный принцип** для всех видов проектирования. Простота означает, что вместе с важными данными **не выводится бесполезная, несущественная или избыточная информация.** С другой стороны, простота вовсе не означает скудность изобразительных средств. Поскольку **степень простоты** нельзя объективно измерить, ее **нужно рассматривать как общий принцип в контексте методов проектирования и оценки.**

Наглядность — это **степень прозрачности** функционирования системы. В идеале пользователь должен иметь ощущение прямого контакта с техническим процессом, а не с автоматизированной системой управления.

Наглядность позволяет опознавать цели и функции устройств по некоторым визуальным образам интерфейса (цвет, форма, вид).

Наглядность должна обеспечивать связь между техническим процессом, его режимами и **мысленной моделью пользователя**.

Система управления должна поддерживать и улучшать наглядность управляемого процесса.

Последовательность означает, что **для отображения одинаковых** или **аналогичных элементов системы** применяются **однотипные обозначения**. Чтобы описание или визуализация системы были последовательными, **необходимо установить** или **разработать принципы структурирования.**

Последовательность можно рассматривать как **наглядность**, **основанную на аналогии.**

Последовательность, **наиболее трудно реализуемая характеристика интерфейса пользователя.** Для ее достижения необходимо классифицировать используемые сущности, а затем применять одинаковые правила (язык, сокращения, цвет) для идентификации объектов каждого класса.

Принцип последовательности, с другой стороны, **требует, чтобы количество классов было сведено к минимуму.**

Препятствия на пути достижения последовательности:

- **трудности, когда в проекте участвуют несколько человек**
- когда к определенному представлению **применимы несколько противоречивых правил и не очевидно, какое из них выбрать.**

Для преодоления такой ситуации используются **два подхода:**

1) **каждый элемент связывается только с одним зрительным признаком** (величина, цвет, форма), так что разные ситуации могут отображаться одновременно, например с помощью размера и цвета;
2) **введение определенной иерархии**, когда определенный тип информации перекрывает другие.

Взаимодействие компонентов системы управления и подходы к их интеграции.

Уровень управления производством

Сервер

ПО систем MES и ERP уровней

Уровень управления технологическим процессом

ПО сервера, SCADA-системы:
- на базе WinCC (Siemens);
- на базе Cimplicity GE Fanuc;
- на базе In Touch (Wonderware).

Сервер

АРМ оператора

АРМ старшего мастера

АРМ химлаборанта

АРМ технолога

Switch

Средний уровень автоматического контроля и учета

Аппаратно-программные средства:
- на базе ПЛК SIMATIC (Siemens);
- на базе ПЛК GE Fanuc;
- на базе ПЛК Modicon;
- другие.

Панель оператора:
- на базе сенсорной панели фирмы Siemens;
- на базе панели фирмы GE Fanuc;
- на базе панели фирмы Schneider Electric;
- другие.

Панель оператора GE Fanuc

Панель оператора Siemens

Панель оператора Schneider Elecntric

ПЛК GE Fanuc

ПЛК Simatic

ПЛК Modicon

Нижний уровень автоматического контроля и учета

БИ-М5 блок интерфейсный

Оборудование производства НПП "ТЭК"

УОНК-Л программно-технический комплекс устройства определения нагрузки от колесных пар локомотивов

МИ-12М модуль индикации

ТП-03 тензометрический преобразователь

ТензоТЭК индикатор весовой

СД-01 система дозирования до 36 компонентов

СД-02 система дозирования до 36 компонентов

СД-03 система дозирования до 36 компонентов

КВ-03 контроллер взвешивания

ЩКУ-СМ щит контроля и управления силозадающей машиной

CAN

БВ-02 блок взвешивания

микроконтроллер дозирования ТЭК

микроконтроллер дозирования ТЭК

модуль дозирования ТЭК

Полевой уровень

Конвейерные дозаторы

Бункерные дозаторы симплексного (непрерывного) действия

Бункерные дозаторы дискретного действия

Технологическое оборудование производства НПП "ТЭК" и партнеров.

Конвейерные весы

Автомобильные весы

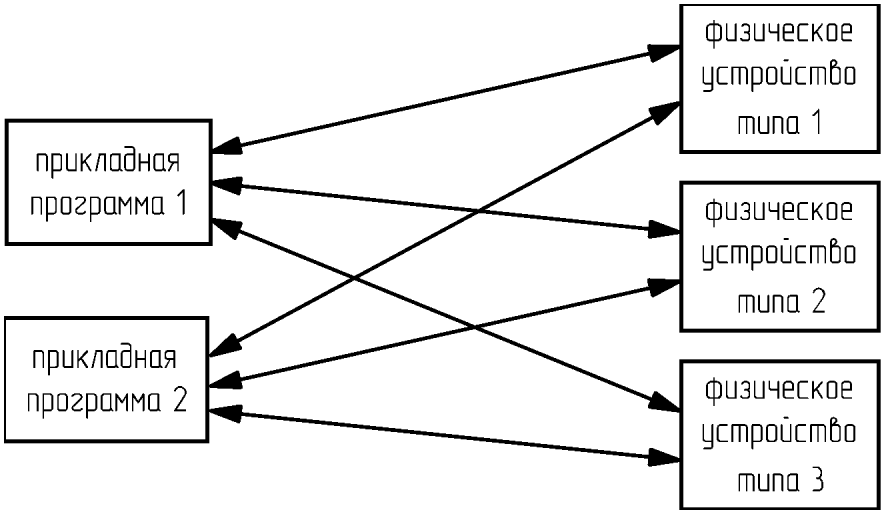
Вагонные и железнодорожные весы

Платформенные весы

Крановые весы

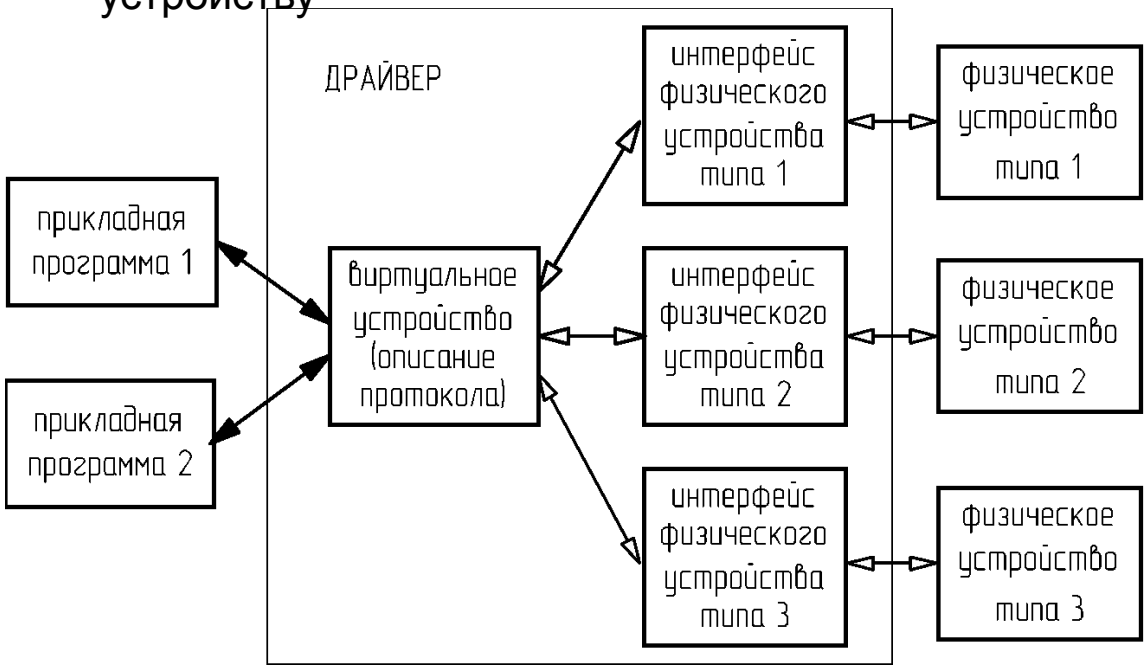
Бункерные весы БВ

Весоповерочные установки



Без применения концепции виртуального устройства программист должен учитывать различия в управляющих команд для выполнения одних и тех же действий на разных типах оборудования.

Непосредственная привязка прикладных программ к физическому устройству



При использовании виртуального устройства достаточно записать управляющие команды на абстрактном языке. Драйвер конкретного устройства преобразует эти команды в управляющие последовательности для реального устройства.

Привязка к физическому устройству через виртуальное

Стандарт OPC

OPC *(**O**LE (object linking and embedded – объединение и встраивание объектов) for **P**rocess **C**ontrol (для управления процессом)) — это стандарт взаимодействия между промышленными компонентами системы сбора данных и управления (SCADA).*

Главной целью стандарта OPC явилось обеспечение возможности совместной работы средств автоматизации, функционирующих на разных аппаратных платформах, в разных промышленных сетях и производимых разными фирмами.

OPC-взаимодействие основано на схеме клиент-сервер.

OPC-клиент, например SCADA, вызывая определенные функции объекта OPC-сервера, подписывается на получение определенных данных с определенной частотой. В свою очередь, OPC-сервер, опросив физическое устройство, вызывает известные функции клиента, уведомляя его о получении данных и вручая сами данные.

Таким образом, при OPC-взаимодействии используются как прямые (от клиента к серверу), так и обратные (от сервера к клиенту) вызовы.

Через интерфейсы OPC одни приложения могут считывать или записывать данные в другие приложения, обмениваться событиями (сообщениями), оповещать друг друга о нештатных ситуациях (тревогах), осуществлять доступ к данным, зарегистрированным в архивах. Эти приложения могут располагаться как на одном компьютере, так и быть распределенными по сети.

Стандарт OPC состоит из нескольких частей:

- **OPC DA** (*OPC Data Access*) – спецификация для обмена данными между клиентом (например SCADA) и аппаратурой (контроллерами, модулями ввода-ввода и др.) в реальном времени;
- **OPC Alarms & Events** (*A&E*) – спецификация для уведомления клиента о событиях и сигналах тревоги, которые посылаются клиенту по мере их возникновения. Этот сервер пересылает аварийные сигналы, действия оператора, информационные сообщения, результаты контроля состояния системы;
- **OPC HDA** (*Historical Data Access*) – спецификация для доступа к предыстории процесса (то есть к сохраненным в архиве данным). Сервер обеспечивает унифицированный способ доступа с помощью DCOM технологии. Обеспечивает чтение, запись и изменение данных;
- **Batch** – спецификация для особых физико-химических технологических процессов обработки материалов, которые не являются непрерывными. В таких процессах выполняется загрузка нескольких видов сырья в определенных пропорциях согласно рецепту, устанавливаются режимы обработки, а после выполнения цикла обработки и выгрузки готового материала загружается новая партия сырья. Здесь OPC сервер выполняет обмен между клиентом и сервером рецептами, характеристиками технологического оборудования, условиями и результатами обработки;

- **OPC Data eXchange** – спецификация для обмена данными между двумя OPC DA серверами через сеть Ethernet;
- **OPC Security** – спецификация, которая определяет методы доступа клиентов к серверу, которые обеспечивают защиту важной информации от несанкционированной модификации;
- **OPC XML-DA** – набор гибких, согласующихся друг с другом правил и форматов для представления первичных данных с помощью языка XML (англ. eXtensible Markup Language) — расширяемый язык разметки – предназначен для хранения и передачи данных), веб технологий и сообщений SOAP (*Simple Object Access Protocol* — простой протокол доступа к объектам);
- **OPC Complex Data** – дополнительные спецификации к OPC DA и OPC XML-DA, которые позволяют серверам работать со сложными типами данных, такими как бинарные структуры и XML-документы;
- **OPC Commands** – набор программных интерфейсов, который позволяет OPC клиентам и серверам идентифицировать, посылать и контролировать команды, исполняемые в техническом устройстве (в контроллере, модуле ввода-вывода);
- **OPC Unified Architecture** – принципиально новый набор спецификаций, который уже не базируется на DCOM технологии.

Из перечисленных спецификаций в России широко используются только две: **OPC DA** и реже – **OPC HDA**.

OPC DA сервер

Сервер OPC DA является наиболее широко используемым в промышленной автоматизации. Он обеспечивает обмен данными (запись и чтение) между клиентской программой и физическими устройствами.

Данные состоят из трех полей:

- значение,
- качество и
- временная метка.

Параметр качества данных позволяет передать от устройства клиентской программе информацию о выходе измеряемой величины за границы динамического диапазона, об отсутствии данных, ошибке связи и другие.

Существует четыре стандартных режима чтения данных из OPC сервера:

- **синхронный режим**: клиент посылает запрос серверу и ждет от него ответ;
- **асинхронный режим**: клиент отправляет запрос и сразу же переходит к выполнению других задач. Сервер после выполнения функции запроса посылает клиенту уведомление и тот забирает предоставленные данные;
- **режим подписки**: клиент сообщает серверу список тегов, значения которых сервер должен отправлять клиенту только в случае их изменения. Для того, чтобы шум данных не был принят за их изменение, вводится понятие "мертвой зоны", которая слегка превышает максимально возможный размах помехи;
- **режим обновления данных**: клиент вызывает одновременное чтение всех активных тегов. Активными называются все теги, кроме обозначенных как "пассивные". Такое деление тегов уменьшает загрузку

В каждом из этих режимов данные могут читаться либо из кэш-памяти OPC сервера, либо непосредственно из физического устройства. Чтение из кэш-памяти выполняется гораздо быстрее, но данные к моменту чтения могут устареть. Поэтому сервер должен периодически освежать данные с максимально возможной частотой. Для уменьшения загрузки процессора используют параметр частоты обновления, которая может быть установлена для каждой группы тегов индивидуально. Кроме того, некоторые теги можно сделать пассивными, тогда их значения не будут обновляться данными из физического устройства.

Запись данных в физическое устройство может быть выполнена только двумя методами: синхронным и асинхронным и выполняется сразу в устройство, без промежуточной буферизации.
В синхронном режиме функция записи выполняется до тех пор, пока из физического устройства не поступит подтверждение, что запись выполнена. Этот процесс может занимать много времени, в течение которого клиент находится в состоянии ожидания завершения функции и не может продолжать выполнение своей работы.
В асинхронном режиме записи клиент отправляет данные серверу и сразу продолжает свою работу. После окончания записи сервер отправляет клиенту соответствующее уведомление.

OPC DA сервер может иметь (не обязательно) пользовательский интерфейс, который позволяет выполнять любые вспомогательные функции для облегчения работы с оборудованием.

Понятие резервирования в системах управления.

Резервирование является практически единственным и широко используемым **метод**ом **кардинального повышения надежности** систем автоматизации.

Резервирование позволяет создавать системы:

- аварийной сигнализации,
- противоаварийной защиты,
- автоматического пожаротушения,
- контроля и управления взрывоопасными технологическими блоками

и другие, относящиеся к уровням безопасности SIL1...SIL3 по стандарту МЭК 61508-5, а также системы, в которых даже короткий простой ведет к большим финансовым потерям (системы распределения электроэнергии, непрерывные технологические процессы и так далее).

Резервирование позволяет создавать высоконадежные системы из типовых изделий широкого применения.

Составной частью систем с резервированием является:

- ***подсистема автоматического контроля работоспособности и***
- ***подсистема диагностики неисправностей.***

Примечание - В настоящее время наибольшая доля отказов в системах автоматизации приходится на программное обеспечение.

Целью резервирования может быть обеспечение:

- **безотказности** или
- **безопасности**.

Методы резервирования, используемые для достижения этих двух целей, **существенно различаются.**

Основное различие состоит в том, что

– **для обеспечения безопасности достаточно снизить вероятность *только опасных отказов***,

в то время как

– **для обеспечения безотказности требуется обеспечить работоспособность системы *при всевозможных отказах***.

Поэтому системы, связанные с безопасностью, получаются проще, чем отказоустойчивые системы при условии одинаковой наработки до отказа.

Несмотря на существование большого разнообразия методов резервирования, в системах промышленной автоматизации получили распространение только два из них:

- **резервирование замещением** и
- **резервирование методом голосования** (по стандарту МЭК 61508 обозначаются как (**N out of M**), например **2oo3** (2 из 3) *voting*, **1oo2** (1 из 2) *voting* и др.).

Резервирование может быть

- **общим**, когда резервируется система в целом, и
- **поэлементным** (раздельным), когда резервируются отдельные элементы системы. В случае, когда в системе много однотипных элементов (например, модулей ввода сигналов термопар), число резервных элементов может быть в несколько раз меньше, чем резервируемых.

Кратность резерва – отношение числа резервных элементов к числу резервируемых, которое выражается несокращаемой дробью, например 3:2.

(В частности, в соответствии с ГОСТ 27.002-89, кратность резерва 3:2 нельзя представлять как 1,5 и поэтому, иногда используемый термин "полуторное резервирование" не соответствует стандарту. При сокращении дроби исчезает важная информация об общем количестве элементов в системе).

Дублированием называют резервирование с кратностью резерва один к одному (1:1).

Постоянное резервирование (к нему относится мажоритарное резервирование или метод голосования) – **резервирование с нагруженным резервом**, при котором все элементы в резервированной системе выполняют одну и ту функцию и являются равноправными, а выбор одного из сигналов на их выходе выполняется схемой "голосования", без переключений. **Постоянное резервирование позволяет получить системы с самым высоким коэффициентом готовности.**

Резервирование замещением – резервирование, при котором функции основного элемента передаются резервному только после отказа основного элемента. **Резервирование замещением** может быть **с холодным, теплым** или **горячим резервом**. Его **недостатком** является **зависимость от надежности переключающих устройств.**

Нагруженный резерв ("горячий резерв") – резервный элемент, который находится в таком же режиме, как и основой. **Недостатком горячего резерва является уменьшение ресурса с течением времени. Достоинство: в системах автоматизации с горячим резервом переход на резерв может занимать кратчайшее время.**

Облегченный резерв ("теплый резерв") – резервный элемент, находящийся в менее нагруженном состоянии, чем основной. Например, резервный компьютер в "спящем" режиме является облегченным резервом.

Ненагруженный резерв ("холодный резерв") – резервный элемент, находящийся в ненагруженном режиме до начала его использования вместо основного элемента. **Ненагруженный резерв позволяет получить системы с самой высокой надежностью, но с низким коэффициентов готовности**. Они эффективны в случае, когда система некритична к времени простоя величиной в несколько минут.

Основное отличие между "горячим", "холодным" и "теплым" резервом состоит в длительности периода переключения на резерв.

Время переключения , например, контроллеров составляет:

- при горячем резервировании – от долей (единиц) миллисекунд до долей секунды,
- при теплом – секунды,
- холодном – минуты, часы, дни.

Поэтому время переключения на резерв иногда рассматривают как основной признак безотказности при классификации резервирования замещением.