Конспект по дисциплине «Автоматизированные процессы в полиграфии»

# Лекция 1. 07.02.2023

Одним из основных направлений деятельности человека является совершенствование процессов производства с целью облегчения тяжелого физического труда и повышения эффективности процесса в целом. Это направление может реализоваться через автоматизацию производственных процессов (АПП).

# Цель АПП:

1) повышение производительности

2) повышение качества изделия

3) улучшение условий труда

Тех процесс состоит из трех основных частей:

1) рабочий цикл (основной технологический процесс)

2) холостых ходов (вспомогательные операции)

3) транспортно-накопительный операции

АПП имеет комплексный подход и не решив одну задачу, может не достигнуть определенного эффекта.

Выделяют два вида: частичная и полная.

Частичная автоматизация – автоматизация части операций по управлению производственным процессом при условии, что остальная часть всех операций выполняется автоматически (пример: автоматические линии).

Полная автоматизация характеризуется автоматическим выполнением всех функций, для осуществления производственного процесса без непосредственного вмешательства человека в работу оборудования. В обязанности человека входит настройка машины или группы машин, включение контроль.

# Гибкие автоматизированные системы

ГАП предназначены для автоматизации тех. Процессов со сменным объектом производства, в том числе для единичного и мелкосерийного производства. Включает в себя основное и вспомогательное оборудование, работающее от одной СУ.

Использование ГАП в Японии и Европе даже в автоматизированном режиме и при недостаточном опыте работы приводит к увеличению отдачи станков на 80-200%, к сокращению продолжительности их обслуживания на 60-70%, что в свою очередь уменьшает время производственного процесса и стоимость живого труда примерно на 80%.

Создание ГАП является очень сложным и многоэтапным процессом, поэтому он по возможности автоматизируется, в результате образуется системное окружение ГАП.

Создание такой сложной системы (рис. 1) должно протекать поэтапно, тем более что эксплуатация ее обуславливает значительные изменения в структуре предприятия. Начальным этапом является разработка подсистемы АСВК (Входной контроль), САПР (NASTRAN, ADAMS), АСТПП(Технология), ГАП, АСКИД (Контроль и измерение детали). Основным элементом при создании такой подсистемы – унификация банка данных (БД) для информационного обеспечения функционирования подсистемы. Унификация БД предполагает, прежде всего применение групповой технологии, технологических процессов обработки деталей объединенных рядом технологических и конструктивных признаков

# Блок-схема алгоритма реализации групповой технологии в ГАП

1. Прогнозирование развития технологии оборудования, изделия

2. Анализ унификации изделий и их элементов

3. Анализ унификации ТП и их элементов

4. Группирование деталей

5. Проектирование ГТП (групповой технический процесс) на уровне компоновочных решений

6. Проектирование оснащения

7. Моделирование обработки детали

8. Проектирование производственных процессов

9. Моделирование работы производственных процессов по прогнозу

10. Анализ результатов моделирования, формирование плана корректировки деталей

В состав ГАП включены подсистемы: обработки, манипулирования, контроля и управления. Проектирование, разработка и внедрение осуществляется в установленном нормами и правилами порядке.

# САУ - Система автоматического управления представляет собой комплекс аппаратных и программных средств, предназначенный для обеспечения автоматического управления отдельными агрегатами или группой оборудования, связанных единым техническим процессом.

САУ выполняют функции автоматического управления, регулирования, контроля и защиты процессов, обеспечивающих безаварийную и длительную работу различного технологического оборудования.

# АСУ Автоматизированная система управления (сокращённо АСУ) — комплекс аппаратных и программных средств, а также персонала, предназначенный для управления различными процессами в рамках технологического процесса, производства, предприятия. АСУ применяются в различных отраслях промышленности, энергетике, транспорте и т. п. Термин «автоматизированная», в отличие от термина «автоматическая», подчёркивает сохранение за человеком-оператором некоторых функций, либо наиболее общего, целеполагающего характера, либо не поддающихся автоматизации. АСУ с Системой поддержки принятия решений (СППР) являются основным инструментом повышения обоснованности управленческих решений.

Важнейшая задача АСУ — повышение эффективности управления объектом на основе роста производительности труда и совершенствования методов планирования процесса управления. Различают автоматизированные системы управления объектами (технологическими процессами — АСУТП, предприятием— АСУП, отраслью — ОАСУ) и функциональные автоматизированные системы, например, проектирование плановых расчётов, материально-технического снабжения и т. д.

Система автоматического управления средствами автоматизации

САУ обеспечивают работу по заранее заданной программе. Система управления автоматами может различаться по признакам:

1) по принципу синхронизации

2) по степени централизации управления

3) по методу воздействия

4) по числу управляемых координат

5) по виду программы-носителя

6) по наличию или отсутствию программной связи

СУ бывают централизованными и децентрализованными. У центра СУ технологический цикл управляется с центрального командного устройства (контроллер или ЭВМ). С центрального ком устройства независимо от действия и положения исполнительных органов.

# Особенности централизированных СУ:

1) Продолжительность рабочего цикла для каждого исполнительного органа является как правило величиной постоянной

2) Простота схем управления

3) Надежность в работе

4) Удобство обслуживания и наладки

5) Необходимость иметь дополнительные предохранительные устройства, то есть команды с центрального командного пункта подаются в независимости от действия и положения исполнительных рабочих органов.

Особенности децентрализованной системы управления

1) управление при помощи датчиков, выключаемых исполнительными рабочими органами

2) исполнительные органы связаны между собой так, что каждое последующее движение одного может происходить после окончания движения предыдущего

3) отсутствие сложной блокировки

4) датчики расположены в рабочей зоне оборудования нередко выходят из строя из-за попадания пыли и тд и выдает неправильную информацию об объекте

Наиболее важным и характерным признаком любой системы управления программного управления является способ задания программы обработки или движения. Наиболее высокой надежностью обладают следующие системы управления: централизованная, разомкнутая система управления без обратной связи, системы, которые обеспечивают надежную и точную синхронизацию всех движений рабочего цикла любой сложности.

Основой систем цифрового программного управления являются следящий привод, который состоит из устройства правления, преобразующего устройства, исполнительных двигателей, усилителя, устройства сравнения.

Процесс автоматического слежения заключается в сведении нулю ошибки согласования

Вопросы:

# Что такое непрерывные (контурные) системы управления?

Контурная система программного управления (КСПУ) - система управления, в которой командная информация содержит кроме признака звена, направления и величины перемещения еще и параметры траектории (контура), по которой осуществляется движение. Такие системы обозначают сокращенно СР (continuous path). Контурное управление обеспечивает перемещение звеньев манипулятора по непрерывной траектории, обладает высокой универсальностью и значительными технологическими возможностями. Промышленные роботы контурной СУ применяются для выполнения, как правило, основных, а не вспомогательных технологических операций

# Комбинированные системы управления

Комбинированные системы сочетают в себе достоинства систем управления по отклонению и по возмущению, что повышает точность управления. Действие неучтенных возмущений в комбинированных системах компенсируется или ослабляется управлением по отклонению.

В комбинированных системах компенсационная связь по основному возмущению (задающему воздействию) устраняет составляющую ошибки, вызываемую этим возмущением (изменением задающего воздействия), а в результате действия обратной связи уменьшаются ошибки, вызываемые второстепенными возмущающими воздействиями, по которым нет компенсационных связей. Если с помощью компенсационных связей не полностью устраняются ошибки, вызываемые основными возмущающими (задающими) воздействиями, то остаточные ошибки также уменьшаются с помощью обратной связи.

# Прямоугольные системы управления рабочими перемещениями

Управляют перемещением вдоль отрезков, параллельных направляющим станка. Прямоугольные системы предназначены для последовательного управления одной из двух взаимно-перпендикулярных координат. Такие системы применяются на токарных станках для управления обработкой деталей типа ступенчатых валиков, а на фрезерных — деталей с прямоугольным контуром.

# Лекция 2. 14.02.2023 Надежность программного обеспечения автоматизированных систем управления технологического процесса

Одной из основных частей СУ является ПО, представляющее собой совокупность взаимосвязанных и автономных программ, описаний, инструкций программистам и пользователя, тестов и тд.

Надежность ПО – св-во ПО своевременно выполнять заранее указанные условия эксплуатации, установленные ф-ии.

Надежность устанавливается по результатам работы ПО, те при динамической проверке всех программ на множестве входной информации. Некорректное ПО заведомо ненадежно, однако некорректное ПО может быть и надежным. Определение надежности базируется на понятии отказа программы.

Отказы ПО делятся на случайные и неслучайные. Неслучайные отказы обусловлены действием так называемых компьютерных вирусов. Случайные отказы ПО наблюдаются в случайные моменты времени работы ЭВМ. По своим последствиям эти отказы классифицируются на случайные сбои программ и устойчивые отказы ПО.

Сбой ПО – самоустраняющийся отказ программы, возникающий при некоторых возможных случайных состояний ЭВМ и информации, наблюдаемый пользователем случайный момент времени и исчезающий без вмешательства программиста.

Устойчивый отказ ПО наблюдается в случайный момент процессорного времени в форме результата в нормальном функционировании ЭВМ. Причиной отказа ПО служит некоторая систематическая ошибка программы, после устранения которой программистом данный отказ исчезает, те имеет место восстановление ПО.

Различают ошибки первичного и вторичного типа. Ошибки первичного типа связаны с неточностями в текстах программ и возникают при подготовке носителей и документаций ПО при записях кодов на алгоритмических языках и трансляции программ на машинный язык.

Ошибки вторичного типа во многом являются следствием первичных ошибок программ. К ним относят:

- вычислительные (неверная индексация и подсчет временных параметров, расхождение результата ручного и машинного счета, появление неустойчивых операций и тп)

- логические (пропуск логических условий, неверные краевые условия и др)

- сопряжения интерфейсов (межмодульных, программно-технических, информационных)

Ошибки первичного и вторичного типов порождаются на этапах разработки спецификаций на ПО; проектирования ПО; реализации программ.

Отказы ПО при его эксплуатации имеют ряд отличий от отказов технических элементов:

- Отказ ПО не приводит к разрушению или поломке программного элемента. Отказы ПО не связаны с физическим износом элемента (в частности носителя программ)

- Отказ ПО не коррелирован с процессорным и, тем более, астрономическим временем (с процессорным временем или числом прогонов ПО программ пользователем).

- При длительной эксплуатации ПО все его ошибки могут быть устранены и программы становятся абсолютно надежными. Если обозначить через N(t) число не выявленных ошибок ПО в произвольный момент времени t, то формально имеет место соотношение lim N(t) = 0, справедливое при условии, что в процессе восстановления программ в них не вносятся новые ошибки.

Опыт создания и эксплуатации ПО реального времени показывает, что при устранении одних ошибок вносятся другие.

Для описания надежности ПО используют такие же функциональные и числовые характеристики, как и при исследовании надежности технических элементов.

Основные показатели надежности ПО:

1. функция ненадежности или отказа ПО Q(t) = Вер (того, что отказ ПО появится до момента времени t).

2. функция надежности ПО Р(t) = Вер (того, что отказ ПО появится после момента времени t);

3. интенсивность отказов ПО (λ(t) = dQ/dt:

4. средняя наработка на отказ ПО: t = Ка =

Программное обеспечение АСУТП состоит из большого числа программ, подпрограмм и модулей, находящихся под управлением операционной системы реального времени или программы-диспетчера. Выполнение каждой из этих программ осуществляется последовательно во времени на одном и том же процессоре. Если эти программы имеют взаимные информационные связи или предназначены для получения одного результата у (вычисление одной функции), то в надежностном отношении такой программный комплекс представляет собой простую систему без избыточности и вероятность его безотказной работы равна произведению вероятностей безотказной работы каждой i-ой программы:

где m— общее число программ.

Надежность такого ПО определяется надежностью отказов самой «ненадежной» программы, имеющей наибольшее значение λ, i=1, m.

Для повышения надежности нерезервированного ПО следует в первую очередь улучшить характеристики самых «ненадежных» программ.

Резервирование: три вида. Временное, информационное и программное

Временное резервирование ПО заключается в многократном прогоне одних и тех же «ненадежных» программ и сравнении результатов расчета. Такое нагруженное резервирование позволяет устранять влияние случайных сбоев и выявлять случайные ошибки, требующие восстановления программ.

Информационное резервирование ПО основано на дублированных исходных и промежуточных данных. Эти данные могут проходить дополни тельную обработку, например, усреднение, до ввода в ПО, где они обрабатываются один раз; или обрабатываться одной и той же программой дважды, т.е. информационное резервирование подкрепляется временным.

Программное резервирование предусматривает наличие в ПО двух или больше разных программ для получения одного и того же результата у

или реализации одной функции. Здесь возможно нагруженное и ненагруженное резервирование.

Резервирование программного обеспечения распределенных АСУТП часто сопровождается аппаратурным резервированием.

# Лекция 3.

OPC в свете COM

Историческая справка

Сравнительно недавно, в1994 г., под эгидой Microsoft, была создана организация OPC Foundation. как определяет сама OPC Foundation, ее целью создания является разработка и поддержка открытых промышленных стандартов, регламентирующих методы обмена данными в реальном времени между клиентами на базе PC и ОС Microsoft. Сейчас эта организация насчитывает более 220 членов, включая почти всех ведущих поставщиков контрольно измерительного и управляющего оборудования для АСУ ТП. Достаточно назвать такие фирмы, как Siemens, Shneider Automation, Rockwell Software, Wonderware, Intellium, Ci Technologies, не говоря уже о самой Microsoft.

Технология

Как уже отмечалось выше, технология OPC реализована и продолжает реализовываться по типу разработка стандартов. OPC Foundation определяет направления, по которым ведутся разработки, и создает по этим направлениям комитетв. Комитеты делают следующее:

1) разрабатывают спецификации COM-интерфейсов и COM-объектов

2) присваивают им GUID

3) оформляют все в виде стандартов и опубликовывают

4) генерируют или создают вспомогательные файлы: idl-, h- и c- файлы для Custom-интерфейса; библиотеки типов для интерфейса автоматизации; заместители (proxy) и заглушки (stub) для поддержки межпроцессорного взаимодействия.

5) разрабатывают вспомогательные компоненты, например утилиту opeenum, позволяющую OPC-клиенту «увидеть» список всех OPC-серверов локальной сети

Спецификации

В настоящее время имеются следующие OPC-стандарты.

OPC Common Definitions and Interfaces общие для всех ОРС-спецификаций интерфейсов

Data Access Custom Interface Standard – спецификация COM-интерфейсов для обмена оперативными данными, программирование на C++

Data Access Automation Interface Standard – спецификация COM-интерфейсов для обмена оперативными данными, программирование на языках типа Visual Basic

OPC Batch Custom Interface Specification – спецификация COM-интерфейсов конфигурирования оборудования, программирование на C++

OPC Batch Automation Interface Specification – спецификация COM-интерфейс для конфигурирования оборудования, программирование на языках типа Visual Basic

OPC Alarms and Events Interface Specification – спецификации COM-интерфейсов для обслуживания событий (event) и нештатных ситуаций (alarm), программирование на C++

Historical Data Access Custom Interface Standard – спецификация COM-интерфейсов для работы с хранилищами данных, программирование на C++

OPC Security Custom Interface – спецификация COM-интерфейсов для обработки прав доступа к данным, программирование на C++

Как видим, перечень достаточно большой. Консорциум OPC Foundation пытается охватить все аспекты, связанные с взаимодействием с технологическим оборудованием. В разработке самих спецификаций принимают участие ведущие производители оборудования и систем автоматизации, которые стараются максимально учесть свой опыт и предоставить абсолютно «все необходимое» тому, кто будет использовать OPC. Далее мы проиллюстрируем это на примере спецификации Data Access (DA).

OPC-сервер (потребители «снизу»)

Кто же использует ОРС? Первая категория – производители оборудования автоматизации или ОЕМ (Original Equipment Manufacturer – поставщик комплексного оборудования). Предполагается, что тот, кто создает, например плату сбора данных, снабжает ее не только драйвером, но и реализует OPC-сервер, работающий с этой платой через драйвер или даже напрямую. Тем самым ОЕМ-производитель предоставляет стандартный доступ к своей плате.

Каждая переменная обладает свойствами. Различаются обязательные свойства, рекомендуемые и пользовательские. Обязательными свойствами, понятно, обязана обладать каждая переменная. Это, в-первых, текущее значение переменной, тип переменной и права доступа (чтение и/или запись). Во-вторых, очень важные свойства – качество переменной и метка времени. Технология OPC ориентирована на работу с оборудованием, а оборудование может давать сбои, так что корректное значение переменной не всегда известно ЩЗС-серверу, о чем и уведомляется клиент через качество (хорошее/плохое/неопределенное и дополнительная информация). Метка времени сообщает о том, когда переменная получила данное значение и/или качество. Еще одним обязательным свойством является частота опроса переменной OPC-сервером. Не совсем понятно, почему это свойство объявлено обязательным, так как не все ОРС-серверы работают в режиме опроса оборудования. Поэтому существуют серверы, не реализующие это свойство. Последним из обязательных свойств является описание переменной. Это строковое значение, содержащее информацию для пользователя о том, зачем нужна эта переменная.

Дополнительные свойства являются необязательными для реализации в ОРС-сервере. Это, например диапазон изменения (выход за границы диапазона должен специальным образом обрабатываться клиентом) и единица измерения. Есть перечень рекомендуемых свойств, но можно добавить и свои собственные.

Получение данных

Существует три основных способа получения ОРС-клиентом данных от ОРС-сервера: синхронное чтение, асинхронное чтение и подписка. При синхронном чтении клиент посылает серверу запрос со списком интересующих его переменных и ждет, когда сервер его выполнит. При асинхронном чтении клиент посылает серверу запрос, а сам продолжает работать. Когда сервер выполнил запрос, клиент получает уведомление (через интерфейс соответствующего COM-объекта, реализованного в клиенте!). И, наконец, в случае по подписки клиент передает серверу список интересующих его переменных, а сервер затем регулярно присылает клиент информацию об изменившихся переменных из этого списка (опять же, через интерфейс соответствующего COM-объекта клиента!). Эти списки в терминологии OPC называются группами. Каждый клиент может поддерживать одновременно много групп с разной скоростью обновления.

Что должен сделать производитель, если он задался целью обеспечить свой продукт стандартным интерфейсом? Он должен получить нужную спецификацию и прилагаемые программные компоненты. Затем он должен изучить COM-интерфейсы тех СОМ-объектов этой спецификации, которые относятся в ней к модели ОРС-сервера. И, наконец он должен посадить самого опытного программиста за Visual Studio, и тот с помощью ATL-библиотеки реализуют требуемые интерфейсы, а значит и OPC-сервер

ОРС-клиент (потребители «сверху»)

Правила игры заданы – ОРС-сервер поставляет данные, ОРС-клиент потребляет. Этим задается вторая категория пользовательских спецификаций ОРС. И к ней относятся в первую очередь те, кто реализует ПО

# Тема: Архитектура PROM

Все объекты системы управления в производственном процессе загружается желательно целевой ресурс

1. …

2 Сервер среды выполнения представляет базу данных системы управления производственным процессом, работающую в режиме реального времени, и содержит все технологические сообщения

3. Станция оператора используется для управления контролем производственного процесса на предприятии и отображает все данные для оператора

# Промышленная сеть Ethernet

С точки зрения общей системы управления производственными процессами ключевыми факторами являются надежность и готовность к передаче данных реального системы автоматизации производственных процессов (контроллеров) туда, где эти данные обрабатываются (сервер среды выполнения) и база данных, работающая в режиме реального времени и подстанция оператора, которые ответственна за контроль. В этом случае необходимо применять резервирование, чтобы обеспечить наивысшую возможную работоспособность сети.

# Глобальная база производственных данных

# Определение и характеристики ETHERNET POWERLINK

ETHERNET Powerlink: Промышленная сеть Ethernet, работающая в режиме реального времени – это реальность

Ethernet имеет гарантированное будущее.

Базовые концепции Ethernet доступны уже более 30 лет. Длительные циклы амортизации, характерные для индустрии автоматизации. Требуют надежного основания.

Технология Ethernet хорошо известна.

Ethernet и соответствующие протоколы сегодня широко известны. Огромный объем имеющихся инструментов, программ и компонентов приведет к дальнейшему сокращению цен.

Ethernet – прозрачная технология.

Стандарты Ethernet объединяют различные протоколы передачи данных на основе IP. Интеграция информационных технологий и автоматизации при использовании Ethernet дает вам реальную функциональную совместимость с гибкостью Ethernet.

Ethernet является системой реального времени.

С Ethernet Powerlink, Ethernet также включает уровни датчиков и исполнительных механизмов, с временами цикла дом 200 мкс, и радикальное улучшение точности синхронизации, лучше одной микросекунды.

# Структура сети

В Ethernet Powerlink выделяются области реального времени и области, где режим реального времени не требуется. Это разделение соответствует типичной концепции машины и технологического процесса. Оно также удовлетворяет растущим требованиям к безопасности с целью предотвращения хакерских атак на уровне машин или ущерба от ошибочной передачи данных на верхних иерархических уровнях сети. Фактические требования режима реального времени. Менее критические с временной точки зрения данные прозрачным образом передаются между областью реального времени и стандартной областью с использованием стандартных IP кадров. А четкое разграничение между сетью машины и сетью предприятия с самого начала предотвращает потенциальные угрозы безопасности, сохраняя полную прозрачность данных.

Модель ISO. Стандартизация для связи между системами различных изготовителей. Нужно согласовывать типы коммуникационной среды. Процедуры доступа, протоколы. Основой для описания систем связи является общая стандартная модель передачи данных (соединение открытых систем), разработанная международной организация по стандартизации.

Можно выделить 7 уровней:

-физический

- канальный

- сетевой

- транспортный

- сеансовый

- представления

- прикладной

Уровни Ethernet Powerlink

# Уровень 1

Изначально протокол определялся для физического уровня на основе 100Base-TX (Fast Ethernet, 100 Мбит/с по витой паре). В конце 2006 года был разработан вариант Ethernet Powerlink для Gigabit Ethernet (1 Гбит/с).

В сетях Ethernet Powerlink (в сегменте сети, в котором требуется сеть реального времени) рекомендуется использовать концентраторы вместо коммутаторов для минимизации задержек и джиттера. Для проектирования сетей Ethernet Powerlink применяются руководства IAONA (*Industrial Ethernet Planning and Installation Guide*) в части прокладки кабелей. В качестве разъемов применяются как распространенные Ethernet-разъемы 8P8C (RJ45) так и М12.

# Уровень 2 – MAC, коммутатор…

Уровень 2 описывает механизмы, использованные для безопасной передачи данных. На этот уровень включены стандарты Ethernet для управления доступом к среде (MAC). На этом уровне работают мосты и коммутаторы. В отличие от концентраторов, эти устройства интеллектуальны и анализируют полученные кадры Ethernet. Коммутатор анализирует распределение станций на сети, используя адреса источника и адресата. Таким образом, он может посылать полученные пакеты.

# Уровень 3 – IP, маршрутизатор…

Вторая часть TCP/IP – межсетевой протокол IP. Он включен на уровень 3 и ответственен за реальную передачу данных по сети. На этом уровне работают маршрутизаторы. Они соединяют подсети, используя IP-адреса, которые уникальны в сети. Поэтому коммутаторы на уровне 3 – это маршрутизаторы.

# Уровень 4 – TCP, UDP…

Уровень 4 (транспортный уровень) включает первую часть термина TCP/IP. Протокол управления передачей. TCP можно сравнить с UDP (Протоколом пользовательских датаграмм). Оба реализуют обмен данными между тремя нижними уровнями (системой транспортировки) и тремя верхними уровнями (прикладной системой). Единственное отличие – метод, которым обрабатывается поток данных. TCP предлагает больше механизмов защиты, чем UDP, но медленнее из-за обширной обработки данных.

# Уровень 5, Уровень 6, Уровень 7 – Сеанс, презентация, приложение…

Уровни 5, 6 и 7 формируют прикладную систему. Они ответствененны за структурирование связи, представление данных. Все приложения, использующие TCP/IP, включены на эти уровни.

Уровень протокола

Сетевой уровень – уровень 3

На сетевом уровне устанавливается логическое соединение между передатчиком и приемником. Работа сетевого уровня важна для маршрутизации сети, когда данные передаются более, чем между двумя точками. Кроме маршрутизации сети, выполняется управление пакетами. Таким образом, транспортный протокол управляет работой всей сети.

Транспортный уровень – уровень 4.

Транспортный уровень отделяет уровни, технологически зависимые от верхних уровней. На этом уровне логические адреса преобразуются в физические адреса, определяется маршрутизация сети и распределение пакетов.

Сеансовый уровень – уровень 5.

Этот уровень отвечает за синхронизацию задач на уровнях передачи и приема и содержит такие функции, как установка соединений, организация буферизации данных, проверка права доступа и контроль соединений.

Уровень представления – уровень 6.

На этом уровне формат данных узла преобразуется с учетом синтаксиса, используемого в сети. Дополнительные задачи включают, например, кодирование или сжатие данных.

Прикладной уровень – уровень 7

Этот уровень обеспечивает интерфейс между пользователем и сетью для работы с сетевыми приложениями без дополнительных специальных знаний по функционированию сетей

Режимы работы

Совместимые устройства может работать в следующих рабочих режимах:

- базовый. Устройство работающее в сетях Ethernet когда не требуется передача данных в режиме реального времени. Этот режим используется по умолчанию после включения устройства.

# Проектирование программного обеспечения полиграфических машин и систем обработки информации

Новая аппаратная база, микроЭВМ контроллеры позволяет реализовать в системах автоматического управления оптимальные законы управления и этим обеспечивает их более высокие статические и динамические показатели. Типовыми режимами работы электропривода являются:

- разгон

- торможение

- реверс

- стабилизация определенных параметров при сильно изменяющихся внешних воздействиях

- слежение, в том числе программное при наличии внешних воздействий

- согласование конечных состояний, режимов стабилизации или слежения

Если в электроприводе последовательно реализуется хотя бы 2 из указанных режимов, то параметры стационарного регулятора с жесткой структурой следует выбирать с учетом обеспечения приемлемого качества процесса этих режимов. Наивысшее качество процессов может быть достигнуто в том случае, когда для каждого режима работы электропривода используется регулятор соответствующей структуры настройка которого зависит от изменяющихся параметров объекта управления и внешних воздействий. При использовании управляющих ЭВМ структура регулятора изменяется простым переходом от одной программы расчета управляющего воздействия к другой.

Рассмотрим возможные структуры и методы проектирования цифровых регуляторов для каждого режима работы электропривода.

Для режимов стабилизации и слежения при наличии возмущений может быть использован один и тот же типовой алгоритм управления, схема которого изображена на рис 1, а. В блоке 1 этого алгоритма опрашиваются преобразователи информации, используемые в системе управления и измеряющие состояние электропривода и внешних воздействий. Формирование вектора y в блоке 2 заключается в переходе от абсолютных значений выходных величин преобразователей информации к отклонениям относительно номинальных (или программных) значений; здесь же вектор y включается значение управляющего воздействия, рассчитанное в блоке 3 на предыдущем такте. Такое расширение вектора измеренных координат позволяет компенсировать запаздывание, обуславливаемое управляющей ЭВМ. Это запаздывание равно времени выполнения расчетов в блоках 1-4.

В блоке 3 рассчитывается приращение u управляющего воздействия относительно номинального управления, обеспечивающего номинальный (или программный) режим стабилизации. Исходными данными для этого являются вектор y измеряемых координат и вектор z, являющийся оценкой неизмеряемых координат состояния. Матричные коэффициенты K и L являются матрицами-строками, если электродвигатель управляется по одному входу (по цепи якоря или по цепи возбуждения), и имеют две строки, если электродвигатель управляется по двум входам. Выбором матриц K и L обеспечиваются заданные свойства собственного движения системы управления электроприводом, а также компенсируются доминирующие возмущающие воздействия. Для расчета матриц K L используются методы комбинированного, модального и оптимального управления.

В блоке 4 форрмируется полное управляющее воздействие как сумма номинального и рассчитанного отклонений управления; это полное управляющее воздействие выводится на цифроаналоговоый преобразователь.

В блоке 5 вычисляется вектор оценки неизмеряемых координат состояния. Эта оценка используется для расчета управляющего воздействия в следующем такте. Выбором матриц P, G H можно обеспечить требуемую скорость сходимости оценки, фильтрацию ошибок измерения, экстраполяцию отсчетов, снимаемых с преобразователей информации, а также требуемый порядок астатизма системы.

Одновременно требуется обеспечить выполнение некоторых соотношений координат вектора состояния (фазовых координат) x вида φ(x)=0, определяющих, например, условия постоянства частоты вращения двигателя, мощности и т.п. В пространстве состояний рассматриваемое уравнение задает некотороые многообразие, в частности поверхность, отклонение от которого характеризуется ошибкой.

Алгоритм управления, обеспечивающий минимизацию этой ошибки, имеет вид:

Аналитическое выражение для составляющей V(x) может быть получено из условия e=0. Матрица K1 обеспечивает желаемое качество отработки ошибки и выбирается с помощью математического обеспечения, используемого при синтезе регуляторов стабилизации или слежения.

На основании сказанного блок схема алгоритма управления согласованеим конечного и начального состояний режимов разгона и стабилизации имеет вид, изображенный на рисунке 1, б. В блоке 1 опращиваются преобразователи информации и в ЭВМ вводятся координаты вектора состояния x электропривода.

Схема типового алгоритма такого оптимаьного управления приведена на рис.2.

В блоке 1 осуществляется ввод координат вектора состояния x с преобразователей информации, а также ввод с пульта управления координат вектора конечного состояния xк, которое должно быть достигнуто в результате разгона, торможения или реверса. Если число преобразователей информации меньше числа координат вектора х, то после блока l вводится блок оценки координат вектора x. Если с пульта управления поступает новая уставка xk (блок 2), то в блоке 3 рассчитываются новые значения коэффициентов аппроксимирующих функций φi(x)=0.

В блоке 4 путем проверки логических условий, полученных на этапе синтеза, определяется номер I участка оптимальной траектории.

Затем рассчитывается отклонение e изображающего многообразия и выбирается коэффициент усилиения стабилизирующей составляющей управляющего воздействия (блок 5).

В блоке 6 вычисляется основная составляющая V управляющего воздействия, обеспечивающая движение по выбранному многообразию.

В блоке 7 формируется полное управляющее воздействие, которое затем выводится на объект управления (блок 8). Составляющая управления Ke обеспечивает ликвидацию скользящих режимов в том случае, когда основная составляющая V имеет релейный характер.

Все рассмотренные программы выполняются на одной и той же управляющей ЭВМ. Тип программы, т.е. структура регулятора выбирается с помощью специальной планирующей программы (рис.3), исходными данными для которой являются команды с пульта управления, а также показания преобразователей информации о состоянии объекта управления и внешних воздействий.

Планирующая программа начинает выполняться после пуска ЭВМ и прекращает – после ее остановка. Сразу после пуска ЭВМ выполняется специальная программа установки начального состояния регистров и областей ОЗУ (блок 2). При этом задаются начальные условия для всех структур регулятора электропривода. Остальные блоки программы выполняются периодически в процессе работы управляющей ЭВМ.

В блоке 3 опрашиваются уставки с пульта управления, а также показания преобразователей информации. Введенные данные в блоке 4 сравниваются с данными, полученными на предыдущем цикле. Изменяющиеся условия работы электропривода индицируются присвоением логической переменной L единичного значения (блок 6). При этом блоке 5 запоминаются новые условия работы. Если же условия работы не изменяются, то в блоке 8 логической переменной L присваивается нулевое значение.

В блоке 9 в результате проверки логических условий, определяющих последовательность переключения структуры регулятора, находится номер I программы расчета управляющего воздействия, которая выполняется в данное время. При изменившихся условиях работы (L=1 в блоке 10) в блоке 11 рассчитываются параметры программы Пi (элементы матричных коэффициентов алгоритма стабилизации, коэффициенты уравнения многообразия алгоритма согласования и т.п.). Затем выполняется одна из описанных программ Пi расчета управляющего воздействия.

# Этапы проектирования и внедрения АСУ ТП

Стандарт [ГОСТ](https://ru.wikipedia.org/wiki/%D0%93%D0%9E%D0%A1%D0%A2) 34.601-90 предусматривает следующие стадии и этапы создания [автоматизированной системы](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0):

1. Формирование требований к АС

* Обследование объекта и обоснование необходимости создания АС
* Формирование требований пользователя к АС
* Оформление отчета о выполнении работ и заявки на разработку АС

2. Разработка концепции АС

* Изучение объекта
* Проведение необходимых научно-исследовательских работ
* Разработка вариантов концепции АС и выбор варианта концепции АС, удовлетворяющего требованиям пользователей
* Оформление отчета о проделанной работе

3. Техническое задание

* Разработка и утверждение технического задания на создание АС

4. Эскизный проект

* Разработка предварительных проектных решений по системе и её частям
* Разработка документации на АС и её части

5. Технический проект

* Разработка проектных решений по системе и её частям
* Разработка документации на АС и её части
* Разработка и оформление документации на поставку комплектующих изделий
* Разработка заданий на проектирование в смежных частях проекта

6. Рабочая документация

* Разработка рабочей документации на АС и её части
* Разработка и адаптация программ

7. Ввод в действие

* Подготовка объекта автоматизации
* Подготовка персонала
* Комплектация АС поставляемыми изделиями (программными и техническими средствами, программно-техническими комплексами, информационными изделиями)
* Строительно-монтажные работы
* Пусконаладочные работы
* Проведение предварительных испытаний
* Проведение опытной эксплуатации
* Проведение [приёмочных испытаний](https://en.wikipedia.org/wiki/Acceptance_testing)

8. Сопровождение АС.

* Выполнение работ в соответствии с гарантийными обязательствами
* Послегарантийное обслуживание

Эскизный, технический проекты и рабочая документация — это последовательное построение все более точных проектных решений. Допускается исключать стадию «Эскизный проект» и отдельные этапы работ на всех стадиях, объединять стадии «Технический проект» и «Рабочая документация» в «Технорабочий проект», параллельно выполнять различные этапы и работы, включать дополнительные.

Данный стандарт не вполне подходит для проведения разработок в настоящее время: многие процессы отражены недостаточно, а некоторые положения устарели.

# Проектирование микропроцессорных систем управления полиграфическим оборудованием

# 1. Технология разработки и отладки микропроцессорной системы

Микропроцессорная система может быть описана на различных уровнях абстрактного представления.

Существующую микропроцессорную систему можно описать на любом известном уровне представления, но в начальной стадии проектирования ее можно описать только на концептуальном уровне. В процессе разработки системы происходит переход от одного уровня ее представления к другому, более детальному. Каждая абстракция несет в себе только информацию, которая соответствует данному уровню, и не содержит каких-либо сведений относительно более низких уровней. Микропроцессорная система может быть описана, например, на одном из следующих уровней абстрактного представления:

1) "черный ящик";

2) структурный;

3) программный;

4) логический;

5) схемный.

На уровне "черного ящика" микропроцессорная система описывается внешними спецификациями; перечисляются внешние характеристики.

Структурный уровень создается компонентами микропроцессорной системы: микропроцессорами, запоминающими устройствами, устройствами ввода/вывода, внешними запоминающими устройствами, каналами связи. Микропроцессорная система описывается функциями отдельных устройств и их взаимосвязью, информационными потоками.

Программный уровень разделяется на два подуровня: команд процессора и языковой. Микропроцессорная система интерпретируется как последовательность операторов или команд, вызывающих то или иное действие над некоторой структурой данных.

Логический уровень присущ исключительно дискретным системам. На этом уровне выделяются два подуровня: переключательных схем и регистровых пересылок. Подуровень переключательных схем образуется вентилями и построенными на их основе операторами обработки данных. Переключательные схемы подразделяются на комбинационные и последовательностные; первые в отличие от последних не содержат запоминающих элементов. Поведение системы на этом уровне описывается алгеброй логики, моделью конечного автомата, входными/выходными последовательностями 1 и 0. Комбинационные схемы представляются таблицей истинности, в которой каждому входному набору значений сигналов ставится в соответствие набор значений сигналов на выходах. Последовательностные схемы могут описываться диаграммами или таблицами входов/выходов, в которых определены взаимно однозначные соответствия между входами схемы, внутренними состояниями (комбинациями значений элементов памяти) и выходами. Подуровень регистровых пересылок характеризуется более высокой степенью абстрагирования и представляет собой описание регистров и передачу данных между ними. Он включает в себя две части: информационную и управляющую. Информационная часть образуется регистрами, операторами и путями передачи данных. Управляющая часть определяет зависящие от времени сигналы, инициирующие пересылку данных между регистрами.

Сxемный уровень образуется резисторами и конденсаторами. Показателями поведения системы на этом уровне служат напряжение и ток, представляемые в функции времени или частоты. Этот уровень описания дискретной системы широко используется в описаниях аналоговых систем и не является ни наинизшим из возможных, ни достаточным для полной характеристики системы.

Этапы проектирования микропроцессорных систем

Микропроцессорные системы по своей сложности, требованиям и функциям могут значительно отличаться надежностными параметрами, объемом программных средств, быть однопроцессорными и многопроцессорными, построенными на одном типе микропроцессорного набора или нескольких, и т.д. В связи с этим процесс проектирования может видоизменяться в зависимости от требований, предъявляемых к системам. Например, процесс проектирования МПС, отличающихся одна от другой содержанием ПЗУ, будет состоять из разработки программ и изготовления ПЗУ.

При проектировании многопроцессорных микропроцессорных систем, содержащих несколько типов микропроцессорных наборов, необходимо решать вопросы организации памяти, взаимодействия с процессорами, организации обмена между устройствами системы и внешней средой, согласования функционирования устройств, имеющих различную скорость работы, и т. д. Ниже приведена примерная последовательность этапов, типичных для создания микропроцессорной системы:

1. Формализация требований к системе.

2. Разработка структуры и архитектуры системы.

3. Разработка и изготовление аппаратных средств и программного обеспечения системы.

4. Комплексная отладка и приемосдаточные испытания.

Этап 1. На этом этапе составляются внешние спецификации, перечисляются функции системы, формализуется техническое задание (ТЗ) на систему, формально излагаются замыслы разработчика в официальной документации.

Этап 2. На данном этапе определяются функции отдельных устройств и программных средств, выбираются микропроцессорные наборы, на базе которых будет реализована система, определяются взаимодействие между аппаратными и программными средствами, временные характеристики отдельных устройств и программ.

Этап 3. После определения функций, реализуемых аппаратурой, и функций, реализуемых программами, схемотехники и программисты одновременно приступают к разработке и изготовлению соответственно опытного образца и программных средств. Разработка и изготовление аппаратуры состоят из разработки структурных и принципиальных схем, изготовления прототипа, автономной отладки.  
Разработка программ состоит из разработки алгоритмов; написания текста исходных программ; трансляции исходных программ в объектные программы; автономной отладки.

Этап 4. см. Комплексная отладка.

На каждом этапе проектирования МПС людьми могут быть внесены неисправности и приняты неверные проектные решения. Кроме того, в аппаратуре могут возникнуть дефекты.

Источники ошибок

Рассмотрим источники ошибок на первых трех этапах проектирования.

Этап 1. На этом этапе источниками ошибок могут быть: логическая несогласованность требований, упущения, неточности алгоритма.

Этап 2. На данном этапе источниками ошибок могут быть: упущения функций, несогласованность протокола взаимодействия аппаратуры и программ, неверный выбор микропроцессорных наборов, неточности алгоритмов, неверная интерпретация технических требований, упущение некоторых информационных потоков.

Этап 3. На этом этапе источниками ошибок могут быть: при разработке аппаратуры - упущения некоторых функций, неверная интерпретация технических требований, недоработка в схемах синхронизации, нарушение правил проектирования; при изготовлении прототипа - неисправности комплектующих изделий, неисправности монтажа и сборки; при разработке программных средств - упущения некоторых функций технического задания, неточности в алгоритмах, неточности кодирования.

Субъективные неисправности отличаются от физических тем, что после обнаружения, локализации и коррекции больше не возникают. Однако, как следует из перечня источников ошибок, субъективные неисправности могут быть внесены на этапе разработки спецификации системы, а это означает, что даже после самых тщательных испытаний системы на соответствие ее внешним спецификациям в системе могут находиться субъективные неисправности.

Процесс проектирования - итерационный процесс. Неисправности, обнаруженные на этапе приемосдаточных испытаний, могут привести к коррекции спецификаций, а следовательно, к началу проектирования всей системы. Обнаруживать неисправности необходимо как можно раньше, для этого надо контролировать корректность проекта на каждом этапе разработки.

Основные методы контроля правильности проектирования следующие: верификация - формальные методы доказательства корректности проекта; моделирование; тестирование.

Для контроля корректности проекта на каждом этапе проектирования необходимо проводить моделирование на различных уровнях абстрактного представления системы и проверку правильности реализации заданной модели путем тестирования. На этапе формализации требований контроль корректности особо необходим, поскольку многие цели проектирования не формализуются или не могут быть формализованы в принципе. В идеальном случае разрабатываются тесты, целиком основанные на этой спецификации и дающие возможность проверки любой реализации системы, которая объявляется способной выполнять функции, оговоренные в спецификации. Этот способ - полная противоположность другим, где тесты строятся применительно к конкретным реализациям. Независимая от реализации функциональная проверка обычно заманчива лишь в теоретическом плане, но практического значения не имеет из-за высокой степени общности.

После обнаружения ошибки должен быть локализован ее источник, чтобы провести коррекцию на соответствующем уровне абстрактного представления системы и в соответствующем месте. Ложное определение источника ошибки или проведение коррекций на другом уровне абстрактного представления системы приводит к тому, что информация о системе на верхних уровнях становится ошибочной и не может быть использована для дальнейшей отладки при производстве и эксплуатации системы.

Отладка микропроцессорных систем. О правильности функционирования микропроцессорной системы на уровне "черного ящика" с полностью неизвестной внутренней структурой можно говорить лишь тогда, когда произведены ее испытания, в ходе которых реализованы все возможные комбинации входных воздействий, и в каждом случае проверена корректность ответных реакций. Однако исчерпывающее тестирование имеет практический смысл лишь для простейших элементов систем. Следствием этого является тот факт, что ошибки проектирования встречаются при эксплуатации, и для достаточно сложных систем нельзя утверждать об их отсутствии на любой стадии жизни системы. В основе почти всех методов испытаний лежит та или иная гипотетическая модель неисправностей, первоисточником которой служат неисправности, встречающиеся в практике. В соответствии с моделью в рамках каждого метода предпринимаются попытки создания тестовых наборов, которые могли бы обеспечить удовлетворительное выявление моделируемых неисправностей. Любой метод тестирования хорош ровно настолько, насколько правильна лежащая в его основе модель неисправности.

Диагностика неисправности - процесс определения причины появления ошибки по результатам тестирования. Отладка - процесс обнаружения ошибок и определение источников их появления по результатам тестирования при проектировании микропроцессорных систем. Средствами отладки являются приборы, комплексы и программы.

Точность, с которой тот или иной тест локализует неисправности, называется его разрешающей способностью. Требуемая разрешающая способность определяется конкретными целями испытаний.

Так как процесс проектирования микропроцессорной системы содержит неформализуемые этапы, то отладка системы предполагает участие человека.

Свойство контролепригодности системы.

Успех отладки зависит от того, как спроектирована система, предусмотрены ли свойства, делающие ее удобной для отладки, а также от средств, используемых при отладке. Для проведения отладки проектируемая микропроцессорная система должна обладать свойствами управляемости, наблюдаемости, предсказуемости.

**Лекция 5 14.03.2023**

Структура обеспечения АСУ

1. Организационное обеспечение – выработка общей концепции функционирования АСУ, разработка структуры системы, правил взаимодействия между ее подсистемами и отдельными элементами, обеспечивающих ее устойчивое и эффективное функционирование. Организационное обеспечение АСУ вводится как на действующих предприятиях путем модернизации, так и на новых.

Процесс создания АСУ включает в себя следующие стадии:

1. Научно-исследовательские и опытно-конструкторские работы. Выработка общей концепции функционирования АСУ и формирования состава задач по управлению, которые разделяются по уровню иерархии объекта управления предприятием, цеха или участка.

2. Предпроектные работы. В процессе предпроектных и проектных работ главное внимание должно быть уделено проверка экономической эффективности спроектированных систем.

3. Изготовление АСУ, наладка и ввод ее в эксплуатацию.

2. Информационное обеспечение – совокупность директивных данных памяти, воплощённой нормативной справочной информации , слежения (сбора оперативной информации по учету и контролю), документооборот, кодирования и шифровка данных, технологическая схема обработки данных.

Сбор учетных данных, представляющий собой информационные потоки о состоянии объекта, может осуществляться тремя способами:

1. Автоматический режим – данные о состоянии объекта фиксируются с помощью датчиков и непосредственно вводятся в ЭВМ.

2. Автоматизированный режим – данные оцениваются оператором и вводятся в ЭВМ, но анализируется оператором.

3. Неавтоматизированный режим – данные считываются и анализируются оператором.

3. Алгоритмическое обеспечение

В процессе функционирования АСУ ТП управляющий вычислительный комплекс может решать следующие задачи:

1. Сбор данных по параметрам объекта, их обработка и регистрации.

2. Определение параметров процесса или объекта.

3. Анализ режима протекания процесса с целью выработки рекомендаций для управления или команд для непосредственного управления.

Нормальное функционирование АСУ ТП возможно в том случае, когда при решении указанных выше задач отдельные операции выполняются по определенным правилам и в необходимой последовательности.

Алгоритм – это предписание, определяющее содержание и последовательность выполнения операций, переводящих исходные данные в требуемый результат. Алгоритм представляется в виде схемы последовательности действия на входящую информацию.

4. Программное обеспечение.

Программа – данные, используемые для управления ЭВМ при обработке информации в соответствии с принятым алгоритмом управления.

Программирование – это процесс составления упорядоченных последовательностей действий (программ) для ЭВМ.

Программное обеспечение управляющего вычислительного комплекса – совокупность программ систем обработки информации и документов, необходимых для эксплуатации этих программ.

Программное обеспечение делится на:

- общее (системное)

Это математическое обеспечение, разрабатываемое одновременно с проектированием ЭВМ.

- специальное (функциональное)

В состав операционной системы входит специальный язык, файловая система и драйверы. Файловая система представляет собой хранилище программ и данных.

**Технологическое обеспечение –** совокупность технологической документации, основу которой составляют инструкции для каждого пользователя и обслуживающего персонала. Кроме инструкций, в состав технологического обеспечения входят рекомендуемые технологические схемы прохождения заказов в производстве, журналы учета прохождения заказов, документы, фиксирующие сроки и исполнителей заказов, документы, фиксирующие сроки и исполнителей заказов на каждом этапе технологического процесса.

# ЛЕКЦИЯ 7 - Проектирование программного обеспечения полиграфических машини систем обработки информации 21.03.2023

## Синтез алгоритмов управления исполнительными системами

Новая апппаратная база - микро-ЭВМ контроллеры — позволяет

реалищзовать в системах автоматического управления отпимаольные законы управления и этим обеспечивает их более высокие статические и динамические показатели.

Типовыми режимами работы электропривода является разгон, тороможение, реверс; стабилизация определенных параметров при сильно изменяющихся внешних воздействиях; слежение в том числе программное, при наличии внешних воздействий; согласование конечных состояний ркжимов пуска - тороможения с начльным состоянием режимов стабилизации или слежения

Наивысшее качество процессов может быть достигнуто в том случае, когда для каждого режима работы электропривода используется регулятор соответствующей структуры, настройка которого зависит от изменений параметров объекта управления и внешних воздействий.

Рассмотрим возможные

Для режимов стабилизации и слежения при наличии возмущений может быть использован один и тот же типовой алгоритм управления, схема которого изображена на рис. 6.1, а в блоке 1 этого алгоритма опрашиваются преобразователи информации, используемые в системе управления и измеряющие состояние электропривода и внешних воздействий. Формирования вектора у в блоке 2 заключается в переходе от абсолютных значений выходных величин преобразователей информации к отклонениям относительно номинальных (или программных) значений: здесь же в вектор у включается значение управляющего воздействия, рассчитанное в блоке 3 на предыдущем такте. Такое расширение вектора измеренных координат позволяет компенсировать запаздывание, обусловливаемое управляющей ЭВМ. Это запаздывание равно времени выполнения расчетов в блоках 1-4.

В блоке 3 рассчитывается приращение и управляющего воздействия относительно номинального управления, обеспечивающего номинальный (или программный) режим стабилизации Исходными Данными для этого являются вектор у измеряемых координат и Вектор z, являющийся оценкой неизмеряемых координат состояния. Матричные коэффициенты к и с являются матрицами-строками, если электродвигатель управляется по одному входу (по цепи якоря или по цели возбуждения), и имеют две строки, если электродвигатель управляется по двум входам. Выбором матриц K и L обеспечиваются заданные свойства собственного движения системы управления электроприводом, а также компенсируются доминирующие возмущающие воздействия. Для расчета матриц K и L используются методы комбинированного, модального и оптимального управления. В блоке 4 формируется полное управляющее воздействие как сумма номинального и рассчитанного отклонений управления это полное управляющее воздействие выводится на цифроаналоговый преобразователь.

В блоке 5 вычисляется вектор оценки неизмеряемых координат состояния. Это оценка используется для расчета управляющего воздействии в следующем такте. Выбором матриц P, G и H можно обеспечить требуемую скорость сходимости оценки, фильтрацию ошибок измерения, экстраполяцию отсчетов, снимаемых с преобразователей информации, а также требуемый порядок астелома системы.

Одновременно требуется обеспечить выполнение некоторых соотношений координат вектора состояния (фазовых координат хрида ф(х) = 0, определяющих, например, условия постоянства Настоты вращения двигателя, мощности т. п. в пространстве состояний рассматриваемое уравнение задает некоторое многообразие, в частности поверхность, отклонение от которого характеризуется ошибкой

**e = ф(х)**

Алгоритм управления, обеспечивающий минимизацию этой ошибки, имеет вид

u= V(x) +K₁e.

Аналитическое выражение для составляющей V(x) может быть палучено из условия = 0. Матрица K1 обеспечивает желаемое качество отработки ошибки и выбирается с помощью математического обеспечения, используемого при синтезе регуляторов стабилизации или слежения.

На основании сказанного блок-схема алгоритма управления согласованием конечного и начального состояний режимов разгона и стабилизации имеет вид, изображенный на рисунке 6.1, б. В блоке 1 опрашиваются преобразователи информации и в ЭВМ вводятся координаты вектора состояния х электропривода. Если число преобразователей информации меньше числа координат вектора состояния и если выход преобразователей сильно зашумлен, то после блока 1 необходимо выполнить блок оценки координат вектора состояния. В рассматриваемом алгоритме предполагается, что все координаты вектора состояния измеряются допустимой ошибкой. В блоке 2 вычисляется ошибка е, характеризующая отклонение изображающей точки от заданного многообразия. В блоке 3 вычисляется основная составляющая управляющего воздействия V(x). обеспечивающая движение по заданному многообразию.

Полное управляющее воздействие и вычисляется блоке 4 как сумма основной составляющей V(x) и стабилизирующей составляющей, обеспечивающей устойчивое движение. Рассчитанное управляющее воздействие выводится на объект управления. Принципиальное решение задачи синтеза оптимального управления электроприводом получено для различных критериев качества. Если в процессе синтеза удается получить оптимальное управление как функцию состояния uopt=u(x), то реализация такого управления на ЭВМ не встречает принципиальных затруднений. Однако для нелинейного электропривода такая ситуация встречается редко. В процессе синтеза оптимального управления обычно удается получить только типы управляющих воздействий и последовательность их переключения, причем моменты переключения установить не удается. Алгоритм функционирования ЭВМ, реализующей подобное оптимальное управление, может быть синтезирован следующим образом. Для каждого (-го типа управляющих воздействий, составляющих в совокупности оптимально управление, численными методами рассматриваются участки оптимальной траектории для различных комбинаций начальных и конечных условий.

В блоке 1 осуществляется ввод координат вектора состояниях с преобразователей информации, а также ввод с пульта управления координат вектора конечного состояния хк, которое должно быть достигнуто в результате разгона, торможения или реверса. Если число преобразователей информации меньше числа координат вектора х, то после блока 1 вводится блок оценки координат вектора х. Если с пульта управления поступает новая уставка хк (блок 2), то в блоке 3 рассчитываются новые значения коэффициентов аппроксимирующих функций ф (х) = 0. В блоке 4 путем проверки логических условий, полученных на этапе синтеза, определяется номер і участка оптимальной траектории. Затем рассчитывается отклонение е изображающей точки от соответствующего многообразия и выбирается коэффициент усиления стабилизирующей составляющей управляющего воздействия (блок 5). В блоке 6 вычисляется основная составляющая V управляющего воздействия, обеспечивающая движение по выбранному многообразию. В блоке 7 формируется полное управляющее воздействие, которое затем выводится на объект управления (блок 8), Составляющая управления ке обеспечивает ликвидацию скользящих режимов в том случае, когда основная составляющая V имеет релейный характер.

Все рассмотренные программы выполняются на одной и той же управляющей ЭВМ. Тип программы, т. е. структура регулятора выбирается с помощью специальной планирующей программы (рис. 6 3), исходными данными для которой являются команды пульта управления, а также показания преобразователей информации о состоянии объекта управления и внешних воздействий. Планирующая программа начинает выполняться после пуска ЭВМ и прекращает после ее останова. Сразу после пуска ЭВМ выполняется специальная программа установки начального состояния регистров и областей ОЗУ (блок 2). При этом задаются начальные условия для всех структур регулятора электропривода. Остальные блоки программы выполняются периодически в процессе работы управляющей ЭВМ. В блоке 3 опрашиваются уставки пульта управления, а также показани преобразователей информации. Введенные данные в блоке 4 сравниваются с данными, полученными на предыдущем цикле. Изменяющиеся условия работы электропривода индицируются присвоением логической переменной L единичного значения (блок 6) При этом в блоке 5 запоминаются новые условия работы. Если же условия работы не изменяются, то в блоке 8 логической переменной L присваивается нулевое значение.

В блоке 9 в реультате проверки логических условий, определяющих последовательность переключения структруры регулятора, находится номер i программы расчета управляющего воздействия, которая выполняется в данное время. При изменившихся условиях работы (L = 1 в блоке 10) в блоке 11 рассчитывается параметр программы П, (элементы матричных коэффициентов алгортима стабилизации, коэффициенты…)

## 6.2 Математическое описание управления технологическим процессом

Математическое описание процесса управления подачи материала. Подачу необходимого количества материала обеспечивает исполнительный механизм подачи материала. Основой математического описания автоматического управления подсистемой подачи материала является Структурная схема, представленная на рис. 6.4. На схеме приняты следующие обозначения: -Press\_pairs\_task заданная скорость подачи материала;

* Up\_mat\_task - выходное напряжение датчика подачи материала;
* Ucondit\_mat\_gas выходное напряжение датчика степени открытия исполнительного механизма подачи материала;
* P\_pairs - значение выхода материала; -P\_mat\_task - задание на исполнительный механизм;
* Condit\_mat\_gas состояние исполнительного механизма подачи материала;
* Квх - коэффициент передачи аналоговых входов контроллера К;
* Квых - коэффициент передачи аналоговых выходов контроллера K;
* Кдд - коэффициент передачи датчика выхода материала;
* Кос - коэффициент передачи датчика степени открытия исполнительного механизма;
* Ким коэффициент передачи исполнительного механизма;
* Кро- коэффициент передачи регулирующего органа;
* Крд - коэффициент передачи регулятора;
* Кп - коэффициент передачи управляющего воздействия;
* К,к- коэффициент связи выходного и входного параметров.

### 6.3. Графическая среда разработки ПО для микроконтроллеров с архитектурой AVR “Algorithm Builder”

Среда обеспечивает полный цикл разработки (ввод-вывод, отладку)

Логическая структрура программы становится более наглядной, Основное предназначение — максимальное приведение интерфейса разработки к природе человеческого восприятия.

Время разработки сокращается в 3-5 раз в сравнении с ассемблером

ПО предназначено для ОС Windows.

Разработка сводится к формированию блоков

Блоки находятся в пункте меню Elements

* field - поле
* label - метка - обозначает место в алгоритме, куда можно осуществление условных и безусловных переходов, к метке должен примыкать один из векторов перехода
* vertex - вершина блока - по отображению и назначению идентичен метке, но, в отличие от нее, задает расположение блока на рабочей плоскости и всегда является его началом, имя вершине назначается, только если она ялвляется началом подпрограммы или макроса
* condition - условный переход, представляет собой овальный контур, внутри которого вписывается условие перехода и возможный вектор в виде ломаной линии со стрелкой на конце, возле которой возможн необязательное имя вектора. Конец вектора должен либо заканчиваться на какой-то метке, или вершине, либо на отрезке другого вектора, либо иметь имя адресуемой метки.
* jmp vector - относительный короткий безусловный переход. Представляет собой ломаную линию, исходящую из середины блока со стрелкой на конце, аналогичную вектору объекта “Condition”
* setter - настройщик перефирийных устройств -серый прямоугольник, внутрь которого вписано имя настраиваемого перефирийного компонента микроконтроллера, такого как таймер, АЦП, регистр маски прерываний и пр. Настройщик предназначен для формирования последовательности операций микроконтроллера, которые обеспечивают загрузку необходимых констант в соответствующие управляющие регистры ввода-вывода в соответствии с выбранными свойствами. Перед испольщованием этого элемента, тип микроконтроллера должен быть определен (пункт меню Options\Project options закладка “Chip”).

Setter - макрооператор. После компиляции он преобразуется в последовательность команд микроконтроллера, которые обеспечивают загрузку необходимый констант в соответсвующие управляющие регистры.

* text - строка текстового редактора

### 6.4 Принципиальная схема цифрового вольтметра

Описание работы устройства. По сути устройство, представляет собой цифровой вольтметр [4-5]. На входе вольтметра стоит операционный усилитель (DA1), имеющий высокое входное сопротивление. За операционным усилителем следует АЦП (DD2), позволяющий оцифровать интересующее нас напряжение для последующей передачи в микроконтроллер. Микроконтроллер DD1 является главным управляющим звеном устройства, так как он считывает информацию из Ацп общается с персональным И компьютером по последовательному каналу. В устройство также входит преобразователь питания +/-10 В для операционного усилителя. Значение, посылаемое B компьютер, лежит в диапазоне 0...4095 (что соответствует разрядности АЦП), 0 соответствует входному уровню ОВ, 4095- уровню 2,58, зависимость линейная.

# ЛЕКЦИЯ 8

### Цифровой и аналоговый сигнал - графическое представление

### Что такое сигнал? Виды сигналов

Сигнал данных \*\*\*\*– форма представления сообщения данных с помощью физической величины, изменения одного или нескольких параметров которой, отображает его изменение.

Сигналы бывают цифровыми, аналоговыми, дискретными, квантоваными, двоичными

### Отладчик

В Automation Studio отладчик представляет собой инструмент, который позволяет программистам отслеживать и отлаживать программы, написанные для автоматизированных систем управления (АСУ). С помощью отладчика можно следить за выполнением программы пошагово, контролировать значения переменных и условий, проверять корректность работы алгоритмов и тестировать их на различных условиях.

Отладчик в Automation Studio позволяет выполнять следующие задачи:

1. Установка точек останова - программист может установить точки останова в программе, чтобы при выполнении кода отладчик останавливался на определенном месте.
2. Пошаговое выполнение кода - отладчик позволяет выполнить код программы пошагово, чтобы программа могла быть проверена на каждом шаге.
3. Контроль за значениями переменных - отладчик позволяет отслеживать значения переменных на каждом шаге выполнения программы.
4. Проверка состояния системы - отладчик позволяет проверить состояние системы и диагностировать возможные ошибки.
5. Использование команд отладки - отладчик предоставляет множество команд отладки, которые позволяют выполнять различные задачи в процессе отладки.

### Многозадачность

Многозадачность - это способ организации работы компьютерной системы, при котором на одном компьютере или в одной программе может одновременно выполняться несколько задач, каждая из которых работает независимо от остальных.

Многозадачность может быть реализована на уровне операционной системы, которая распределяет ресурсы компьютера между запущенными задачами, или на уровне приложения, которое само управляет своими потоками выполнения.

Многозадачность позволяет повысить эффективность использования компьютера, ускорить выполнение задач и улучшить пользовательский опыт. Однако для правильной работы многозадачности необходимо уметь правильно распределять ресурсы и управлять потоками выполнения, чтобы избежать конфликтов и необходимости выполнения задач в последовательном порядке.

### Какие существуют топологии. Преимущества и недостатки

1. Звездообразная топология: в этой топологии все устройства сети подключены к одному центральному узлу. Достоинства: простота управления и обслуживания, возможность быстро обнаружить и устранить неисправности в сети. Недостатки: ненадежность центрального узла - если он выходит из строя, вся сеть может остановиться.
2. Шина: в этой топологии все устройства сети подключены к одной линии, которая называется шиной. Достоинства: простота управления, низкая стоимость и легкость установки. Недостатки: отказ одного устройства или поломка шины может привести к полной остановке сети.
3. Кольцевая: в этой топологии устройства сети соединены в кольцо. Достоинства: высокая производительность, поскольку данные передаются только в одном направлении. Недостатки: отказ одного устройства может привести к проблемам со всей сетью.
4. Деревообразная: в этой топологии устройства сети соединены в виде дерева с корневым узлом в вершине. Достоинства: хорошая масштабируемость, возможность добавлять новые узлы без перестройки всей сети. Недостатки: сложность управления, высокая стоимость.
5. Смешанная: в этой топологии комбинируются несколько типов топологий. Достоинства: возможность комбинировать преимущества разных типов топологий. Недостатки: сложность управления, высокая стоимость.

# ЛЕКЦИЯ 9 25.04.2023

Структура сети B ETHERNET Powerlink выделяются области реального времени и области, где реального времени не требуется. Это разделение соответствует типично концепция машины и технологического процесса. Оно также удовлетворяет рас хушим требованиям безопасности с целью предотвращения хакерских атак на уровне машин или ущерба от ошибочной передачи данных на верхних иерархи ческих уровнях сети. Фактические требования режима реального времени удов летворяются в области реального времени. Менее критические с временной точ рання дане прозрачным образом передаются между областью реального времени и стандартной областью с использованием стандартных IP четко разграничение между сетью машины и сетью предприятия с самого нача да предотвращает потенциальные угрозы безопасности, сохраняя полную, прозрачность данных.

1. IP-адрес (англ. Internet Protocol address) — это уникальный числовой идентификатор, присваиваемый каждому устройству в компьютерной сети, который используется для идентификации и адресации устройств в сети. IP-адрес состоит из четырех чисел, разделенных точками, каждое из которых может быть от 0 до 255. Например, 192.168.1.1 - это IP-адрес роутера или компьютера в локальной сети.
2. Маска сети (англ. subnet mask) — это числовой параметр, который используется для определения того, какая часть IP-адреса устройства относится к адресу сети, а какая — к адресу устройства в этой сети. Маска сети задает количество бит, которые используются для определения адреса сети. Например, если маска сети состоит из 24 бит (255.255.255.0), то первые три числа IP-адреса указывают на адрес сети, а последнее число указывает на адрес устройства в этой сети.
3. Протокол передачи данных (англ. data transmission protocol) — это стандарт, который определяет, как данные передаются между устройствами в компьютерной сети. Протоколы передачи данных определяют способ кодирования данных, формат заголовков и другие параметры, которые используются для обеспечения надежной и эффективной передачи данных в сети. Например, протоколы передачи данных включают TCP (Transmission Control Protocol), UDP (User Datagram Protocol) и ICMP (Internet Control Message Protocol).
4. Роутер (англ. router) и коммутатор (англ. switch) являются сетевыми устройствами, используемыми для соединения нескольких устройств в одну сеть. Однако, у этих устройств есть некоторые различия.

Роутер является устройством, которое обеспечивает маршрутизацию данных между различными сетями. Роутеры используются для соединения локальной сети с интернетом или другой удаленной сетью. Роутеры могут также выполнять функции брандмауэра, которые обеспечивают защиту сети от внешних угроз и контролируют доступ к ресурсам сети.

Коммутатор является устройством, которое обеспечивает коммутацию данных между устройствами внутри локальной сети. Коммутаторы используются для соединения компьютеров, серверов и других сетевых устройств в одну локальную сеть. Коммутаторы обычно имеют более высокую пропускную способность и ниже задержку, чем роутеры, что делает их более подходящими для использования в локальной сети с высоким трафиком данных.

Таким образом, основное различие между роутером и коммутатором заключается в их функциях. Роутеры используются для соединения различных сетей, в то время как коммутаторы используются для соединения устройств в одной локальной сети

1. Первичный алфавит на уровне представления - это набор символов, который является стандартным для данного протокола или приложения. Например, ASCII (American Standard Code for Information Interchange) - это первичный алфавит, используемый для представления текстовой информации в большинстве сетевых протоколов и приложений.

Вторичный алфавит на уровне представления - это набор символов, который может быть использован для перевода первичного алфавита в другой набор символов, необходимый для конкретного приложения или протокола. Например, Unicode - это вторичный алфавит, который может быть использован для представления символов, отсутствующих в ASCII, таких как символы на других языках или математические символы. Вторичный алфавит может быть определен и использован приложением или протоколом, чтобы обеспечить правильное отображение текстовой информации на разных устройствах или в разных системах.

1. Помехоустойчивое кодирование - это техника обработки данных, которая позволяет обеспечить надежную передачу информации через шумные каналы связи, где возможно возникновение ошибок в передаче данных.

Основной принцип помехоустойчивого кодирования заключается в добавлении в исходный поток данных дополнительной информации (кодовых символов), которая позволяет обнаруживать и исправлять ошибки в передаче. Таким образом, полученный кодовый поток становится более устойчивым к помехам и искажениям, которые могут возникать в канале связи.

Существуют различные методы помехоустойчивого кодирования, включая блочные коды, коды Хэмминга, циклические коды и т.д. Каждый из этих методов имеет свои преимущества и недостатки, а также свои области применения.

В общем, помехоустойчивое кодирование является важным инструментом для обеспечения надежной передачи данных в условиях шумных каналов связи, и широко применяется в различных областях, таких как телекоммуникации, компьютерные сети, сенсорные сети, радиосвязь и т.д.

## Семиуровеневая модель ISO/OSI

1. Физический уровень (Physical Layer): определяет физические аспекты передачи данных, такие как метод передачи данных, тип кабеля и интерфейсы.
2. Канальный уровень (Data Link Layer): обеспечивает надежную передачу данных между двумя устройствами в локальной сети, выполняя управление доступом к среде передачи данных, обнаружение ошибок и их исправление.
3. Сетевой уровень (Network Layer): определяет маршрутизацию и пересылку данных между различными сетями, а также управляет трафиком на сетевом уровне.
4. Транспортный уровень (Transport Layer): обеспечивает надежную передачу данных между двумя устройствами на разных сетях, управляет качеством обслуживания (QoS) и контролирует поток данных.
5. Сеансовый уровень (Session Layer): устанавливает, управляет и завершает сеансы связи между устройствами.
6. Уровень представления (Presentation Layer): обеспечивает стандартные методы кодирования, шифрования и форматирования данных для их передачи между устройствами.
7. Прикладной уровень (Application Layer): предоставляет приложениям доступ к сети и реализует прикладные протоколы, такие как HTTP, FTP, SMTP и т.д.

Режимы работы устройств

ETHERNET Powerlink совместимое устройство может работать в следующих рабочих режимах:

* Базовый Ethernet-режим: Устройство работает прямо в существующих сетях Ethernet, когда не требуется передача данных в режиме реального време. Этот рехи используется по умолчанию после включения устройства
* Предпусковой режим: В ходе запуска системы или после подсоединения уст ройств к существующей сети, данные мишкализации и конфигурации - загружаются по асинхронному каналу.
* Рeжим ETHERNET Powerlink: После завершения процедуры загрузки, устройст во работает в реальном масштабе времени. Управляющий узел проверяет синхронизацию. Время цикла зависит от объема кохранных и асинхронных данных, отество узлов. Базовый цикл состоит из следующих фаз:
  + Фаза запуска: Все сетевые узлы синхронизируются с часами управляющего узла
  + Изохронная фаза: Управляющий узел присваивает каждому узлу фиксированое временное окно для передачи крических с временной точки зрения данных. На этой фазе все другие узлы всегда могут принимать все данные (принцип издатель /подписчик)
  + Авторская фаза. Управляющий узел предоставляет право передавать конкретные данные одному конкретному капу. На этой фазе используются IP протоколы и адресация.

Важные, но все ещё критические с временной точки данные могуи передаваться в течение более длинных циклов, чем базовый цикл

# ЛК 10 16.05.2023

Языки программирования высокого и низкого уровня отличаются по абстракции и близости к машинному коду. Вот их определения, преимущества и недостатки:

1. Языки программирования высокого уровня:
   * Определение: Языки программирования высокого уровня разработаны для облегчения разработки программ, предоставляя абстракции и инструменты для более высокого уровня анализа и проектирования. Они удобны для разработчиков, так как позволяют писать код, который легче понять и поддерживать.
   * Преимущества:
     + Читаемость: Код на языках высокого уровня обычно более читабелен и понятен людям, поскольку используются более естественные конструкции и ключевые слова.
     + Переносимость: Языки высокого уровня обычно абстрагируются от аппаратных особенностей и операционных систем, что делает программы переносимыми на разные платформы.
     + Продуктивность: Благодаря высокоуровневым конструкциям и библиотекам, разработка программ на языках высокого уровня может быть более быстрой и удобной.
   * Недостатки:
     + Производительность: Поскольку языки высокого уровня абстрагируются от низкоуровневых деталей, они могут быть менее эффективными по производительности по сравнению с языками низкого уровня.
     + Ограничения: Некоторые специфические задачи могут быть сложнее реализовать на языках высокого уровня из-за ограничений абстракции и уровня детализации.
   * Примеры: Python, Java, C#, JavaScript, Ruby.
2. Языки программирования низкого уровня:
   * Определение: Языки программирования низкого уровня предоставляют прямой доступ к аппаратным ресурсам компьютера и машинному коду. Они обладают более непосредственным управлением аппаратурой и позволяют разработчику работать на более низком уровне абстракции.
   * Преимущества:
     + Производительность: Языки низкого уровня позволяют полностью использовать ресурсы компьютера и обеспечивают более эффективную работу, особенно для задач требовательных к производительности.
     + Управление ресурсами: Разработчик имеет более прямой контроль над памятью, регистрами и другими ресурсами компьютера, что может быть полезным для оптимизации и специфических сценариев.
   * Недостатки:
     + Сложность: Языки низкого уровня требуют от разработчика более глубокого понимания аппаратуры и более трудоемкой разработки, поскольку низкоуровневые детали должны быть учтены.
     + Портируемость: Код на языках низкого уровня обычно зависит от конкретной аппаратной платформы и операционной системы, что затрудняет переносимость.
   * Примеры: машинные языки, Assembler.

Каждый из этих языков имеет свои особенности и подходит для разных типов задач. Выбор языка программирования зависит от требований проекта, сроков, производительности, уровня абстракции и других факторов.