

Министерство образования РФ
Московский государственный открытый университет

Кафедра электропривода и автоматизации
промышленных установок

Соснин О.М.

Основы автоматизации технологических
процессов

Москва 2006

Содержание

Перечень принятых обозначений

Основные понятия и термины

Введение

Контрольные вопросы к введению

Глава 1. Автоматизированные системы управления технологическими процессами

1.1. АСУТП как основа автоматизации технологических процессов

1.2. Основные функции и структура АСУТП

1.3. Структура и основные функции УВМ

Контрольные вопросы к главе 1

Глава 2. Информационное обеспечение АСУТП

2.1. Энтропия как мера информации

2.2. Количественная оценка информации

2.3. Кодирование информации

2.4. Двоичные коды

2.4.1. Экономичность двоичного кодирования

2.4.2. Арифметические двоичные коды

2.4.3. Неарифметические двоичные коды

2.5. Передача информации по каналам связи

2.5.1. Промышленные информационные сети

2.5.2. Последовательные интерфейсы по стандартам RS232C и RS485

2.5.3. Защита информации от искажений

2.6. Организация обмена информацией в АСУТП

2.6.1. Информационная структура АСУТП

2.6.2. Информационные сети Ethernet

2.6.3. Структура физической среды Ethernet

2.6.4. Контроллерные и полевые сети

2.6.5. Диспетчеризация в рамках АСУТП

Контрольные вопросы к главе 2

Глава 3. Моделирование технологических объектов, управляемых АСУТП

3.1. Алгоритмы функционирования

3.2. Аналитические методы моделирования

3.3. Моделирование технологических циклов

3.4. Экспериментальные методы получения моделей

3.4.1. Одномерные модели

3.4.2. Многомерные модели

Контрольные вопросы к главе 3

Глава 4. Алгоритмы управления в АСУТП

4.1. Задачи управления в АСУТП

4.2. Алгоритмы стабилизации управляющих параметров

4.3. Алгоритмы автоматической оптимизации

4.3.1. Статическая и динамическая оптимизация

- 4.3.2. Симплексный метод линейного программирования
- 4.4. Градиентные методы автоматической оптимизации
- 4.4.1. Поиск экстремума целевой функции
- 4.4.2. Автоматическая оптимизация электрохимической обработки
- 4.4.3. Поиск предельно допустимого оптимального режима
- 4.5. Применение методов нечеткой логики в АСУТП
- 4.5.1. Понятия и операции нечеткой логики
- 4.5.2. Синтез нечеткого регулятора положения

Контрольные вопросы к главе 4

Глава 5. Алгоритмы управления технологическим циклом

- 5.1. Задачи управления технологическим циклом
- 5.2. Синтез алгоритмов комбинационных схем управления
- 5.3. Схемная реализация релейно-контактных комбинационных схем
- 5.4. Схемная реализация комбинационных схем на логических элементах
- 5.5. Синтез алгоритмов последовательностных автоматов
- 5.5.1. Общая структура последовательностного автомата
- 5.5.2. Составление схемы простейшего автомата
- 5.6. Реализация алгоритмов управления последовательностных автоматов
- 5.6.1. Виды запоминающих устройств
- 5.6.2. Триггеры
- 5.6.3. Регистры
- 5.6.4. Преобразователи кодов и арифметические устройства
- 5.7. Обобщенные алгоритмы управления технологическим циклом

Контрольные вопросы к главе 5

Глава 6. Особенности проектирования АСУТП

- 6.1. Основные задачи и принципы проектирования
- 6.2. Этапы разработки и внедрения АСУТП

Контрольные вопросы к главе 6

Глава 7. Системы программного управления производственными установками

- 7.1. Назначение и общая структура
- 7.2. Локальные системы программного управления
- 7.3. Программируемые контролеры
- 7.3.1. Структура ПЛК
- 7.3.2. Языки программирования ПЛК
- 7.3.3. Язык программирования IL
- 7.4. Устройства числового программного управления
- 7.5. Программирование УЧПУ
- 7.6. Исполнительные устройства УЧПУ

Контрольные вопросы к главе 7

Литература

Приложения

Приложение 1. Таблицы кодов обмена информацией в АСУТП
Приложение 2. Базовые понятия теории вероятностей
Приложение 3. Элементы булевой алгебры формальной логики
Приложение 4. Символы и функции стандартного кода ISO-7 для ЧПУ
(ГОСТ 20999-83)

Перечень принятых сокращений

АЛУ – арифметико-логическое устройство;

АРМ – автоматизированное рабочее место;
 АСТПП – автоматизированная система технологической подготовки производства;
 АСУП – автоматизированные системы управления производством (по отраслям);
 АСУТП – автоматизированная система управления технологическим процессом;
 АЦП – аналого-цифровой преобразователь;
 БИС – большие интегральные микросхемы (высокой степени интеграции);
 ВЗУ – внешнее запоминающее устройство;
 ВУ – внешнее устройство;
 ГПМ – гибкий производственный модуль;
 ГПС – гибкая производственная система;
 ЗУ – запоминающее устройство;
 ЛУВМ – локальная УВМ;
 МОТИ – модули обработки текущей информации;
 МП – микропроцессор;
 ОЗУ – оперативное запоминающее устройство;
 ПБ – процессорный (системный) блок;
 ПЗУ – постоянное запоминающее устройство;
 ПИД – пропорционально-интегрально-дифференциальный (регулятор);
 ПК – промышленный компьютер;
 ПЛК – программируемый логический контроллер;
 ПО – программное обеспечение;
 ПУ – пульт управления;
 САПР – система автоматизированного проектирования;
 СПУ – система программного управления;
 СУЭП – система управления электроприводом;
 ТО – технологический объект;
 ТП, или техпроцесс, – технологический процесс;
 УВВ – устройства ввода-вывода информации (параллельный или последовательный интерфейсы);
 УВМ – управляющая вычислительная машина;
 УП – управляющая программа;
 УСО – устройство связи с объектом;
 УФС – устройство формирования состояний (управляющее устройство);
 УЧПУ – устройство ЧПУ;
 ЦАП – цифро-аналоговый преобразователь;
 ЦУВМ – центральная УВМ;
 ЧПУ – числовое программное управление;

ЭВМ – электронная вычислительная машина;
ASCII – American Standard Code for Information Interchange
(американский стандартный код для обмена информацией);
CAD – Computer Aided Design (САПР);
CAM – Computer Aided Manufacturing (АСУТП);
Ethernet – наиболее распространенная стандартная сетевая
технология для создания локальной информационной сети (Этернет);
IEC – International Electrotechnical Commission (МЭК,
Международная электротехническая комиссия);
IEEE – Institute of Electrical and Electronics Engineers (Институт
инженеров по электротехнике и электронике), США;
ISO – International Standardization Organization (Международная
организация стандартизации).

Основные понятия и термины

Автоматизация технологических процессов состоит в том, что функции контроля и управления, ранее выполнявшиеся человеком, передаются автоматическим управляющим устройствам и контрольно-измерительной аппаратуре. Управляющие устройства, получая информацию по каналам обратной связи о ходе техпроцесса, формируют управляющие сигналы, обеспечивающие функционирование технологического объекта (ТО) в оптимальном рабочем режиме.

1.

АСУТП – это совокупность аппаратных средств и их программного обеспечения, предназначенных для управления сложным ТО. АСУТП обеспечивает оптимальный уровень автоматизации сбора, накопления и переработки информации о техпроцессе и формирование таких управляющих воздействий на исполнительные устройства, что работа ТО происходит в оптимальном режиме.

УВМ – это совокупность управляющих устройств АСУТП, выполненных на базе микропроцессорной техники и составляющих конструктивное целое.

Иерархическая структура АСУТП обусловлена тем, что АСУТП реализует централизованное управление с помощью центральной УВМ (ЦУВМ). Однако ЦУВМ управляет исполнительными устройствами не непосредственно, а через локальные УВМ (ЛУВМ), каждая из которых отвечает за определенный участок техпроцесса (см рис. 1,1).

УСО – устройства связи с объектом, входящие в состав УВМ, через которые управляющие сигналы УВМ поступают на управляемый ТО. Иногда в состав УСО включают также устройства обратной связи, через которые информация о параметрах ТО передается на УВМ. Наличие УСО является главным отличием УВМ от обычных микро-ЭВМ, решающих чисто вычислительные задачи.

2.

Информация – это процесс формирования, пересылки и регистрации сообщений, а также зарегистрированная совокупность сообщений.

Бит – двоичная единица информации, эталоном которой является информация, получаемая из опыта с двумя равновероятными исходами. Битом называют также минимальный объем записи информации на электронных или магнитных носителях, соответствующий двоичной единице или двоичному нулю.

Двоичный код – любой код, в котором используются только два символа, обозначаемые обычно 0 и 1.

Арифметический двоичный код – это позиционный весовой код, лежащий в основе двоичной системы исчисления. Вес разряда в нем равен 2^n , где n – номер разряда.

Коды Грея – двоичные коды, служащие для обозначения чисел, причем обозначения соседних чисел различаются лишь одним кодовым символом.

Параллельная передача информации – обмен информацией в виде двоичных кодовых комбинаций фиксированной длины, называемых **словами**.

Последовательная передача информации – обмен информацией побитно, когда биты слова пересылаются один за другим, начиная, например, с младшего бита.

Интерфейс – это совокупность правил обмена информацией между двумя соседними уровнями управления, а также совокупность проводов и иных технических средств, обеспечивающих такой обмен информацией. В настоящее время в АСУТП применяют почти исключительно *стандартные интерфейсы*.

Совокупность программ, устройств и проводов, которые обеспечивают обмен информацией в АСУТП в целом, называют *промышленной информационной сетью*.

Локальные сети АСУТП обычно построены по схеме ведущий – ведомый и употребляют *интерфейс RS485* (см. рис. 2.4).

3.

Алгоритм функционирования – это математически обоснованное описание зависимости между управляющими (входными) и управляемыми (выходными) параметрами ТО; если он реализован в виде компьютерной программы, то называется **моделью ТО**.

Декомпозиция алгоритма – это расчленение алгоритма функционирования на отдельные математически однородные элементы, называемые типовыми звеньями. Модель ТО строится в виде совокупности типовых звеньев.

Аналитические методы моделирования базируются на знании законов природы, определяющих функционирование ТО. Аналитическая модель представляет ТО в виде совокупности динамических звеньев.

Динамическое звено описывает элемент ТО, в котором происходит однократное преобразование энергии, дифференциальным уравнением первого или второго порядка.

Многосвязная (многомерная) модель описывает сложный ТО с несколькими входами и выходами и с перекрестными связями между отдельными каналами.

Одномерная модель (канал) соответствует элементу ТО с одним входом и одним выходом.

Технологический цикл отображает циклическую последовательность сменяющих друг друга технологических операций. Для удобства описания технологический цикл делят на **такты**.

Циклограмма строится в виде таблицы, в строках которой перечислены все командные и исполнительные элементы ТО, а столбцы соответствуют тактам его технологического цикла (см. рис. 3.1.)

Регрессия выходного параметра y на входной параметр x - это любая функция $f(x)$, приближенно представляющая вероятностную зависимость y от x .

Полнофакторный эксперимент представляет собой цикл из 2^m экспериментов, где m - количество факторов, каждый из которых принимает в эксперименте либо максимальное, либо минимальное значение в заданном интервале. Остальные факторы, влияющие на поведение ТО, при этом поддерживаются неизменными.

4.

Алгоритмы управления определяют порядок подачи управляющих воздействий на исполнительные устройства АСУТП. Различают алгоритмы:

- стабилизации параметров техпроцесса при различных возмущениях;
- программного управления технологическим циклом;
- автоматической оптимизации техпроцесса в ходе выполнения производственного задания.

Целевая функция – это строго сформулированная зависимость между параметрами техпроцесса и критерием его оптимальности.

Статическая оптимизация предполагает, что параметры оптимального функционирования техпроцесса определяются до его начала.

Динамическая оптимизация – это поиск оптимального режима во время выполнения технологическим объектом заданной производственной программы. При этом **шагом оптимизации** называют каждую очередную попытку изменения параметров в направлении оптимального режима.

Линейное программирование – это совокупность методов определения параметров оптимального режима в условиях, когда целевая функция и уравнения ограничений, действующих в управляемом техпроцессе, линейны.

Основная задача линейного программирования (см. §4.3.2):

- найти максимум функции $F_y = cx$
- при условиях $Ax \leq b, x \geq 0$
- или найти минимум функции $F_y = -cx$
- при условиях $-Ax \geq -b, x \geq 0$.

При решении основной задачи линейного программирования **симплексным методом** уравнения ограничений представляют в виде (4.7).

Процедура симплексного метода основана на следующих положениях:

- совокупность точек в n -мерном пространстве E_n , ограниченная вышеприведенными условиями, образует область решений поставленной задачи; каждую точку $\mathbf{x}=(x_1, x_2, \dots, x_n)$ из указанной области принято именовать допустимым планом; точка, в которой достигается искомый экстремум заданной целевой функции, называется **оптимальным планом**;
- в пространстве E_n область допустимых решений существует в виде многогранника решений, причем оптимальная точка (оптимальный план) является одной из вершин многогранника решений; любые m линейно независимых векторов $\mathbf{a}_j (j=1, 2, \dots, n, m \leq n)$, удовлетворяющих системе уравнений (4,7), порождают точку $\mathbf{x}=(b_1, b_2, \dots, b_m, 0, \dots, 0)$, являющуюся вершиной многогранника решений и называемую угловой точкой или **опорным планом**;
- оптимальная точка (оптимальный план) является одной из угловых, а потому при оптимизации симплексным методом переходят от одной угловой точки к другой таким образом, что при каждом переходе значение целевой функции обязательно увеличивается (при поиске максимума целевой функции) или обязательно уменьшается (при поиске минимума).

Градиентные методы применяются, если выражение целевой функции или (и) уравнения ограничений техпроцесса являются нелинейными. Сущность градиентных методов заключается в том, что на каждом шаге оптимизации приращения управляющих параметров техпроцесса выбираются пропорциональными частным производным целевой функции по этим параметрам:

$$x_{ij+1} = x_{ij} \pm k_j \frac{\partial F_{ц}}{\partial x_i} \Big|_j \quad i=1, 2, \dots, n,$$

где x_{ij+1} - значение i -го параметра на $(j+1)$ -м шаге оптимизации;

x_{ij} -значение того же параметра на j -том шаге оптимизации;

k_j -коэффициент, определяющий величину шага на j -том шаге оптимизации, называемый также просто **шагом**;

$\partial F_{ц} / \partial x_i$ частная производная целевой функции $F_{ц}$ по i -тому параметру x_i , соответствующая j -тому шагу оптимизации;

n - количество оптимизируемых параметров техпроцесса.

Признаком достижения оптимального режима является равенство нулю всех значений $\partial F_{ц} / \partial x_i$.

Удвоение шага - увеличение или уменьшение в 2 раза значения k_j без расчета новых значений $\partial F_{\text{ц}}/\partial x_i$ с целью ускорения поиска оптимального режима и избежания рысканья.

Предельно допустимый оптимальный режим (предельное регулирование) – это оптимальный режим, который реализуется в одной из граничных точек области допустимых значений параметров техпроцесса, когда нужный экстремум целевой функции не может быть достигнут из-за имеющихся технологических ограничений. Типичным примером предельного регулирования является линейное программирование.

5.

Технологический цикл представляет собой циклически повторяющуюся последовательность сменяющих друг друга технологических операций.

Последовательностный автомат – управляющее устройство, обеспечивающее управление технологическим циклом. Такие автоматы обеспечивают управление последовательностью тактов работы ТО, причем выполнение отдельных тактов обычно связано с использованием информации, полученной по каналам обратной связи при выполнении предыдущих тактов.

Внутреннее состояние автомата – состояние элементов ЗУ в его составе.

Комбинационные схемы управления (комбинационные автоматы) – отдельные узлы управляющего устройства, выходные сигналы которых зависят только от комбинации входных сигналов, поступивших в текущем такте управления.

Единичный сигнал – сигнал высокого уровня потенциала, или замыкание контакта, или намагниченный участок магнитного носителя информации.

Нулевой сигнал – сигнал низкого уровня потенциала, или размыкание контакта, или ненамагниченный участок магнитного носителя информации.

Таблица истинности – таблица соответствия входных и выходных сигналов комбинационной схемы управления. По таблицам истинности получают выражения логических функций схем управления

Логическая (булева) функция Y логических переменных $X_1, X_2, \dots, X_n$ – это выражение

$$Y = f(X_1, X_2, \dots, X_n),$$

полученное путем выполнения над логическими переменными операций инверсии, дизъюнкции и конъюнкции. Логические функции являются алгоритмами управления комбинационных и последовательностных автоматов.

Метод Квайна – Мак-Класки – это регулярная процедура упрощения выражений логических функций путем выполнения над ними операций склеивания и поглощения (см. приложение 3), применяемая для автоматизации составления схем управления.

Метод последовательно-параллельных соединений при составлении релейно-контактных схем управления состоит в следующем:

- логическая инверсия (функция НЕ) реализуется размыкающим контактом реле, а логическое повторение – замыкающим контактом;
- логическое умножение (функция И) реализуется последовательным соединением контактов реле;
- логическое сложение (функция ИЛИ) реализуется параллельным соединением контактов реле.

Для дополнительного упрощения релейно-контактных схем применяются *скобочные формы* логических функций и *мостиковые структуры* схемных решений.

Логические элементы – это микросхемы малой степени интеграции, реализующие простейшие логические функции.

Триггер – элемент электронного ЗУ, характеризующийся двумя устойчивыми состояниями. С точки зрения теории информации это двоичный элемент памяти, способный запомнить либо единичный, либо нулевой сигнал, поступивший на его вход. Триггер – это стандартная ячейка ОЗУ, предназначенная для хранения информации не более 1 бит.

Арифметико-логическое устройство (АЛУ) – это комбинационная часть последовательностного автомата, предназначенная для выполнения заданных арифметических и логических операций. Простейшие автоматы выполняют только логические операции.

Устройство формирования состояний (УФС) автомата задает операции, которые должно произвести АЛУ в текущем рабочем такте. Функционирование УФС зависит как от сигналов, формируемых в АЛУ (сигналов обратной связи), так и от содержимого кадра УП (см.рис.5.8). В простейших автоматах управление от внешней УП не предусмотрено.

Гонка импульсов – это эффект неодновременного формирования выходных сигналов в ответ на одновременное изменение входных сигналов. Гонка импульсов приводит к кратковременному появлению ложных сигналов на выходах управляющего устройства при асинхронном формировании управляющих сигналов.

Асинхронное управление – способ управления, при котором входные сигналы считываются, а выходные сигналы формируются в произвольные моменты времени по мере поступления входных сигналов.

Синхронное управление – способ управления, при котором считывание входных и выдача сформированных выходных сигналов производится в определенные моменты времени, называемые временем

обмена информацией. В остальное время, называемое временем счета (когда формируются новые выходные сигналы) считывание входных сигналов и выдача выходных сигналов не производится. Синхронный и асинхронный способы управления сочетаются с помощью операций *прерывания*, когда прерывается рутинное течение синхронного процесса управления.

Синхронный триггер отличается от асинхронного триггера наличием тактового входа *C* (ср. рис.5.10 и 5.11), так что изменение состояния синхронного триггера возможно только после поступления синхронизирующего импульса на вход *C*. Синхронными триггерами являются наиболее важные виды триггеров: *D*-триггер, *T*-триггер и *JK*-триггер (см. рис. 5.12. и 5.13).

Регистр – электронное устройство памяти, состоящее из группы триггеров, объединенных общим управлением, и предназначенное для записи, хранения и выдачи информации. По своему основному назначению регистры подразделяются на регистры сдвига, регистры хранения и счетные регистры (см. рис. 5.14.-5.16).

Счетчики (счетные регистры) отличаются тем, что помимо функций записи, хранения и выдачи информации выполняют функцию счета поступающих на них импульсов.

Преобразователь кода – комбинационная схема, преобразующая двоичный код входных сигналов в заданный код ее выходных сигналов. В качестве примера укажем на преобразователь кода Грея в двоичный арифметический код (см. §5.4). Преобразователь кода, преобразующий двоичный входной код в единичный код на его выходах, называется *дешифратором* (см. рис. 5.18. и 5.19).

Сумматор – комбинационная схема, предназначенная для сложения и вычитания (в дополнительном коде) двух чисел, представленных в двоичной системе исчисления (см. рис. 5.20. и 5.21).

Схема алгоритма управления технологическим циклом состоит из прямоугольников и ромбов. Прямоугольники служат для отображения арифметических операций и операций с памятью, а ромбы служат для задания логических условий. Графические элементы схемы алгоритма соединяют стрелками, указывающими точную последовательность выполняемых операций (см. рис. 5.22 и 5.23).

6.

Техническое, программное, и информационное обеспечение – основные задачи проектирования АСУТП.

Техническое обеспечение включает в себя выбор комплекса типовых технических средств, необходимых для реализации заданных функций АСУТП, и проектирование необходимого специального оборудования.

Программное обеспечение подразделяется на **системное** (операционные системы и системы управления базами данных), **инструментальное** (системы автоматизированного проектирования и разработки УП) и **прикладное** (совокупность УП).

Информационное обеспечение – это единая система технологической, технико-экономической и справочной информации, обеспечивающая нормальное функционирование АСУТП.

Этапы разработки АСУТП: разработка технического задания, выполнение технического проекта, создание рабочего проекта, процесс внедрения в производство.

7.

Система программного управления (СПУ) – это АСУТП, которая обеспечивает управление ТО по заранее подготовленной и введенной в УВМ на специальном программноносителе программе, включая программу управления технологическим циклом.

Программируемый логический контроллер (ПЛК) – выпускаемое серийно управляющее устройство, построенное на базе микро-ЭВМ и предназначенное преимущественно для управления технологическим циклом. Имеет модульную конструкцию.

УЧПУ – комплектная система числового программного управления (ЧПУ), построенная на базе компьютера. В УЧПУ величины перемещений рабочих органов и другие параметры техпроцесса задаются в числовой форме.

Гибкая производственная система (ГПС) – наиболее сложный и разветвленный вид СПУ, решающий, помимо задач АСУТП, задачи САПР, АСТПП и текущего планирования. Состоит из ЦУВМ и гибких производственных модулей (ГПМ).

Локальные СПУ – это ПК и УЧПУ, управляющие отдельными ГПМ.

Основные функции локальных СПУ: централизованный контроль, регулирование техпроцесса, включая следящее управление перемещениями, программно-логическое управление технологическим циклом, ввод и отображение информации, включая УП, обмен информацией, диагностика СПУ и основного оборудования.

Основные модули в составе ПЛК: микропроцессорный модуль, интерфейсы связи с ЦУВМ и ЛУВМ, дискретные и аналоговые модули ввода-вывода, программатор, модуль питания и др.

Языки символического кодирования ПЛК строятся в соответствии со стандартом IEC 61131-3 и включают в себя:

- язык «лестничных диаграмм» LD, или язык РКС;
- язык набора инструкций IL, или язык булевых логических формул;
- еще три языка кодирования, более сложных.

Линейная интерполяция – задание и отработка перемещения рабочих органов по прямой между заданными опорными точками.

Круговая интерполяция – задание и отработка перемещения рабочих органов по окружности. Помимо координат начала и конца программируемой дуги, при круговой интерполяции задают также координаты центра окружности, часть которой составляет данная дуга.

Эквидистанта – траектория перемещения инструмента, заданная с учетом габаритов последнего.

Программное обеспечение УЧПУ делится на системное, инструментальное и прикладное. Управляющие программы (УП) создаются с помощью инструментального программного обеспечения, входящего в комплект программного обеспечения УЧПУ, и составляют прикладное программное обеспечение.

ISO-7 – стандартизированный на международном уровне язык программирования УЧПУ. В коде ISO-7 программа строится в виде последовательности **кадров**, каждый кадр - из последовательности **слов**. Первый символ каждого слова – латинская буква, а остальные символы являются арабскими цифрами.

Исполнительные устройства УЧПУ подразделяются на устройства с *цикловым управлением*, обеспечивающим лишь включение и отключение исполнительных механизмов, с *позиционным управлением*, обеспечивающим позиционирование, т.е. вывод исполнительного механизма в заданную точку и фиксацию его в этой точке. Наиболее сложны исполнительные устройства с *контурным управлением*, обеспечивающим перемещение исполнительных механизмов по произвольным траекториям с заданной точностью отработки траектории.

Введение

В настоящее время происходит быстрое развитие производства и применения самодействующих машин и аппаратов, увеличение количества производственных процессов, ведущихся по типу безлюдной

технологии. Различные автоматические устройства проникают во все сферы человеческой деятельности, включая науку, производство и быт. Для инженера любой специальности стало необходимым ознакомление с теоретическими основами и практическими приложениями автоматизации применительно к его профессиональным интересам. Это особенно важно для инженеров-электриков, специализирующихся в области автоматизированного электропривода, поскольку большая часть их профессиональной деятельности состоит в создании оборудования для автоматизации различных технологических процессов, его наладке и эксплуатации в производственных условиях.

Обычный курс автоматизации технологических процессов строится на базе технологии определенного производства: машиностроительного, металлургического, химического, текстильного и др. В нем много места уделяется частностям, которые весьма существенны для специалистов соответствующих производств, но не так существенны для инженеров, работающих в сфере создания и эксплуатации систем управления технологическим оборудованием. Специалистам по автоматизированному электроприводу важно получить представление об общих задачах, решаемых автоматизацией в современном высокомеханизированном и автоматизированном производстве, о месте электропривода в системах автоматизации. Они должны изучить основы теории автоматизации технологических процессов и научиться решать простые технические задачи, связанные с проектированием, выбором аппаратной части автоматизированных систем, разработкой алгоритмов и программного обеспечения их функционирования в конкретных условиях эксплуатации.

Термин *автоматизация* относится к весьма широкому классу производственных процессов и других систем организации трудовой и иной деятельности человека, в которых значительный объем операций, относящихся к процессам получения, преобразования, передачи и использования энергии, материалов и особенно информации, передается специализированным техническим устройствам, средствам механизации и управляющим машинам. Автоматизированные процессы, в том числе управление, регулирование и (частично) контроль над ними, протекают автономно, в соответствии с заранее подготовленной и введенной на специальном программоносителе программой, так что не возникает необходимости непосредственного участия человека в их нормальном функционировании. На долю обслуживающего персонала остаются лишь функции общего контроля, а в случае необходимости – ремонта и наладки. Механизация, заключающаяся в замене ручного труда, физических усилий человека машинными операциями, является непременным элементом автоматизации. В отличие от простой механизации автоматизация обязательно включает в себя передачу управляющим машинам операций по управлению и организации

автоматизируемого процесса в соответствии с заранее сформулированной и, возможно, уточняемой во время реализации процесса целью. Цели автоматизации многообразны. Они могут включать в себя решение задач повышения производительности и эффективности труда, улучшения качества продукции, оптимизации управления, обеспечения безопасности трудовой деятельности человека, охраны окружающей среды и др.

Цели автоматизации реализуются с помощью *автоматизированных систем управления (АСУ)*, АСУ – это совокупность математических методов, технических средств (основные из них – это компьютеры и др. микропроцессорные устройства), их программного обеспечения и организационных комплексов, обеспечивающих управление и контроль параметров автоматизируемых объектов в соответствии с поставленной целью их автономного функционирования. Среди объектов автоматизации выделяют:

- 1) технологические, энергетические, транспортные и другие производственные процессы;
- 2) проектирование различных агрегатов и машин, судов, зданий и иных сооружений, производственных комплексов;
- 3) организацию, планирование и управление в рамках цеха, предприятия, стройки, войсковой части и др.;
- 4) научные и технические исследования, медицинское диагностирование, учет и обработку статистических данных, программирование, бытовую технику, охранные системы и пр.

Из всего перечисленного многообразия автоматизируемых объектов мы будем рассматривать только технологические процессы промышленного производства. При автоматизации последних функции управления и контроля, ранее выполнявшихся человеком, передаются автоматическим управляющим устройствам и контрольно-измерительной аппаратуре. Одновременно совершенствуется механизация отдельных рабочих операций. Управляющие устройства, получая информацию по каналам обратной связи об изменении контролируемых параметров, таких как размеры обрабатываемых изделий, скорость обработки, температура, формируют, в соответствии с заданной программой обработки, управляющие сигналы, обеспечивающие выполнение программы обработки в оптимальном рабочем режиме.

В первой главе рассмотрены общие вопросы автоматизации технологических процессов, основные функции и структура АСУТП. Поскольку управление технологическим процессом становится возможным благодаря информационным процессам, которые формируются параллельно текущему техпроцессу, то во второй главе рассмотрены элементы теории информации применительно к формированию информационных процессов управления. Особое

внимание здесь уделено вопросам кодирования информации в двоичных кодах, так как эти коды являются базой функционирования всех современных управляющих устройств. Глава завершается рассмотрением способов организации обмена информацией посредством ее передачи по каналам связи в рамках АСУТП.

Построение АСУТП невозможно без достаточно точного и подробного описания свойств и характеристик управляемого технологического объекта (ТО). Поэтому третья глава посвящена изложению аналитических и экспериментальных методов создания модели ТО, отображающей указанные свойства и характеристики.

Центральное место в учебном пособии занимают главы 4 и 5, посвященные методам анализа и синтеза алгоритмов управления АСУТП. В алгоритмах управления отображаются намечаемые способы решения задач АСУТП по стабилизации и программному управлению параметрами и режимами ТО, обеспечивающее протекание техпроцессов в соответствии с заданным критерием оптимальности. Особое внимание уделено рассмотрению способов оптимизации режимов функционирования ТО с линейными и нелинейными характеристиками и созданию блок-схем алгоритмов управления. Последние являются основой создания управляющих программ в процессе программирования аппаратной части систем автоматизации.

В шестой главе освещены этапы проектирования АСУТП, начиная с выбора необходимых технических средств построения АСУТП, разработки технического задания и кончая рабочим проектированием. В заключение, в седьмой главе, рассмотрены в качестве примера вопросы построения систем автоматизации в машиностроении на базе УЧПУ и программируемых логических контроллеров.

Изучение курса следует проводить последовательно, раздел за разделом, проверяя степень усвоения материала ответами на контрольные вопросы. Работу над пособием следует сочетать с приобретением некоторых практических навыков путем выполнения контрольных заданий с использованием рекомендуемых компьютерных программ.

Контрольные вопросы к введению

1. Каково значение курса автоматизации в системе подготовки специалистов по автоматизированному электроприводу?
2. Дайте определение термину автоматизация и обрисуйте роль обслуживающего персонала в автоматизированных системах.
3. В чем отличие автоматизации от простой механизации?
4. Каковы цели автоматизации производственных процессов?
5. Дайте определение АСУ.
6. Перечислите основные объекты и области автоматизации.

7. В чем заключается автоматизация технологических процессов?
8. Какова структура данного учебного пособия?

Глава 1. Автоматизированные системы управления технологическими процессами

1.1. АСУТП как основа автоматизации технологических процессов

Общие вопросы автоматизации организации и управления производством решаются на уровне автоматизированных систем управления производством (АСУП) по отраслям. Технологическая подготовка производства по отраслям производится в рамках систем автоматизированного проектирования (САПР) и автоматизированных систем технологической подготовки производства (АСТПП), действующих в соответствии с подготовленной помощью САПР проектной документацией. Непосредственное управление технологическими процессами изготовления запланированной продукции возлагается на автоматизированные системы управления технологическими процессами (АСУТП).

Принято отделять проблемы автоматизации организации и управления производством, решаемые на уровне АСУП, от технико-технологических вопросов автоматизации, решаемых на уровне САПР и АСУТП. Последние образуют единый комплекс автоматизированных производственных систем, обычно называемых гибкими производственными системами (ГПС), которые на международном уровне принято именовать системами CAD-CAM, причем системы CAD объединяют функции САПР и АСТПП. В настоящее время создание ГПС является основной задачей автоматизации по отраслям народного хозяйства.

Системы САПР и АСТПП являются специфическими для каждой отрасли производства, а в структуре АСУТП имеется много общего. Эта общность обусловлена тем, что в АСУТП применяются преимущественно электромеханические исполнительные устройства. Прежде всего, укажем на технологические процессы, связанные с изменением формы исходного материала, такие как обработка резанием, прокатка, бумажное, текстильное производство и др., в которых основными рабочими движениями являются перемещения рабочих органов, осуществляемые с помощью электроприводов. Автоматизированные электромеханические комплексы применяются также при добыче сырья в горной, нефтяной и газовой отраслях, при первичной обработке, хранении и транспортировке зерна и другой продукции сельского хозяйства. Разнообразные автоматизированные транспортные системы, применяемые для перемещения твердых, жидких и газообразных продуктов, такие как электропогрузчики, грузовые подъемники, насосные станции и трубопроводные системы имеют в качестве исполнительных устройств силовые электроприводы, а сервоприводы широко применяются в системах управления. Но и в технологических процессах, связанных с изменением агрегатного и физико-химического состояния исходных материалов, подача сырья на обработку и транспортировка готовой продукции производится с помощью электромеханических или управляемых с помощью электромеханических агрегатов

транспортных систем. По перечисленным причинам основное внимание в данном курсе уделено изучению АСУТП с электромеханическими исполнительными устройствами.

Главной задачей АСУТП является повышение эффективности производства путем замены человека-оператора аппаратными средствами, устройствами автоматического управления. С этих позиций определим АСУТП как *совокупность аппаратных средств и их программного обеспечения, предназначенных для управления технологическими объектами, которая обеспечивает оптимальный уровень автоматизации сбора, накопления и переработки информации и формирование таких управляющих воздействий на исполнительные устройства, что работа управляемого объекта происходит в оптимальном режиме.*

Технологическим объектом (ТО) мы здесь называем совокупность технологического оборудования и реализованного на нем производственного процесса. Что касается критериев оптимальности (эффективности) функционирования технологических объектов, то кроме обычно решаемой задачи достижения наибольшего экономического эффекта укажем на всегда актуальные проблемы охраны здоровья работающих и сохранения окружающей среды. Наибольшая экономическая эффективность достигается оптимальным сочетанием средств САПР и АСУТП с учетом требований охраны труда и окружающей среды.

1.2. Основные функции и структура АСУТП

Основным назначением АСУТП является оптимизация технологических процессов в соответствии с заданным критерием эффективности. При этом предполагается, что автоматизация с помощью АСУТП приведет к повышению эффективности производства в заданном отношении. В противном случае АСУТП просто не применяется. Строго сформулированную зависимость между параметрами технологического процесса (техпроцесса) и критерием его оптимальности называют *целевой функцией*. Обычно полагают, что оптимальный режим достигнут, если выбранному сочетанию параметров управляемого техпроцесса соответствует минимум (а иногда максимум) целевой функции. Если из-за технологических ограничений заданный экстремум целевой функции не может быть достигнут, то оптимальный режим имеет место при некоторых граничных значениях контролируемых параметров. Во всяком случае, оптимизация техпроцесса с помощью АСУТП сводится к поддержанию оптимального соотношения его параметров. Если это соотношение стабильно и может быть рассчитано или подобрано заранее, то достаточно, чтобы АСУТП стабилизировала значение контролируемых параметров на заданном оптимальном

уровне. Обычно к тому же необходимо изменять значения заданных параметров по заранее составленной программе ведения техпроцесса. В наиболее сложных случаях оптимальный ход техпроцесса не может быть определен и задан заранее и тогда поиск оптимального режима работы производится автоматически, самой АСУТП, в течении всего времени техпроцесса. Методика поиска зависит от вида заданной целевой функции и ограничений, накладываемых на значения контролируемых параметров.

В зависимости от имеющихся возможностей достижения оптимального режима, в АСУТП применяются следующие приемы управления технологическими процессами:

- Стабилизация заданных значений параметров техпроцесса при различных возмущениях. Примером такой стабилизации может служить стабилизация линейной скорости шлифования при уменьшении диаметра шлифовального круга.

- Изменение параметров техпроцесса по заранее подготовленной на специальном программном носителе программе, включая управление технологическим циклом. Такие АСУТП относятся к классу систем программного управления (СПУ). Реализация программного управления, наряду со стабилизацией контролируемых параметров, присуща всем АСУТП, особенно в станках с ЧПУ, лифтовых подъемниках, роботах-манипуляторах и др.

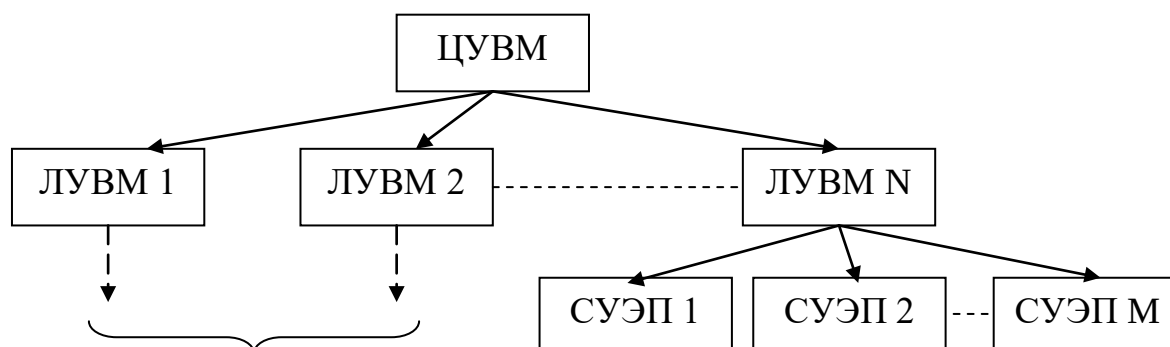
- Автоматическая оптимизация техпроцесса во время выполнения заданной производственной программы и в соответствии с заданным критерием эффективности (целевой функцией). Типичными процессами, нуждающимися в автоматической оптимизации, являются раскрой материалов из заготовок со значительным разбросом габаритов и составление оптимальных смесей из исходных веществ со значительным разбросом по составу.

Управляющие устройства АСУТП строятся на базе средств микропроцессорной вычислительной техники и являются по существу *управляющими вычислительными машинами* (УВМ).

Технологические объекты, управляемые АСУТП, - это совокупность сложных электромеханических систем, таких как станки с ЧПУ, робототехнические комплексы, прокатные станы, трубопроводные системы и прочее. Отдельные единицы оборудования, входящие в состав технологического объекта, управляемого АСУТП, имеют автономные системы управления, позволяющие как встраивать данное оборудование в технологический комплекс, так и использовать его автономно. По указанной причине управление в АСУТП организуется по *иерархическому принципу*. Иерархическое управление является одной из разновидностей централизованного управления. При управлении по иерархическому принципу система управления делится на отдельные *уровни*, или

ранги. Общее управление осуществляется *центральной УВМ* (ЦУВМ), которая считается УВМ высшего (первого) ранга. Однако ЦУВМ при иерархическом управлении управляет технологическим объектом не непосредственно, а только через промежуточные, *локальные УВМ* (ЛУВМ). Все ЛУВМ, управляемые непосредственно от ЦУВМ, называются УВМ второго ранга (второго уровня управления). Если имеются ЛУВМ, управляемые не от ЦУВМ, а от УВМ второго ранга, то такие ЛУВМ называются ЛУВМ третьего ранга. По отношению к ним соответствующая УВМ второго ранга оказывается центральной. В результате, при управлении по иерархическому принципу каждая УВМ управляет лишь управляющими устройствами ближайшего низшего ранга, а подчиняется лишь одной из УВМ ближайшего высшего ранга. Исключение составляет самый нижний уровень управления, на котором осуществляется непосредственное управление технологическим объектом, т.е. не управляющими, а исполнительными устройствами.

Для АСУТП типична трехуровневая иерархическая структура, показанная на рис. 1.1. Здесь ЦУВМ, которая является достаточно мощным промышленным компьютером, снабжает управляющими программами N ЛУВМ, а последние управляют исполнительными электроприводами (например, M электроприводов) через их системы управления (СУЭП) или другими исполнительными устройствами.



На системы управления исполнительными
устройствами

Рис. 1.1. Типичная иерархическая структура АСУ ТП

Локальные УВМ могут строиться на базе компьютеров, программируемых контроллеров и микроконтроллеров в зависимости от сложности решаемых задач управления. В качестве исполнительных устройств могут быть использованы не только электроприводы, но и нагревательные, электрофизические, электрохимические и др. установки. Системы управления

исполнительных устройств могут быть построены как на базе микропроцессорных комплектов, так и без них, могут быть как цифровыми, так и аналоговыми. На рис.1.1 показано прохождение лишь прямых управляющих сигналов, а описание каналов обратной связи от технологического объекта к АСУТП здесь опущено и будет дано ниже. Заметим только, что прохождение сигналов обратной связи строится также иерархическому принципу: от исполнительных устройств и агрегатов информация поступает сначала в ЛУВМ по результатам опросов датчиков, установленных на технологическом оборудовании, а ЦУВМ получает необходимую информацию о состоянии технологического объекта в порядке обмена информацией с ЛУВМ (см. §2.6).

1.3. Структура и основные функции УВМ

Управляющие устройства могут быть построены на дискретных элементах или выполнены в виде УВМ. Будем считать, что управляющее устройство выполнено на дискретных элементах, если в нем функции управления реализуются без применения процессоров. И напротив, если основные логические и арифметические операции, необходимые для осуществления процесса управления, реализуются с помощью микропроцессорных конструкций, то такое управляющее устройство является УВМ. Коротко говоря, *управляющая вычислительная машина (УВМ) – это управляющее устройство, построенное на базе микро-ЭВМ.*

Управляющие устройства на дискретных элементах, такие как магнитные пускатели и аналоговые системы управления электроприводами, применяются в АСУТП на нижних уровнях управления, а на верхних уровнях применяются исключительно УВМ (см. рис.1.1). Тем не менее, многие важные функции УВМ, связанные с вводом, выводом, отображением, преобразованием информации, реализуются на дискретных элементах, не входящих в микропроцессорные комплекты, таких как клавиатура, дисплей, магнитные запоминающие диски и дискеты, аналого-цифровые и цифро-аналоговые преобразователи и пр.

Структура УВМ в составе АСУТП показана на рис.1.2. Конструктивно УВМ выполняется в виде пульта управления ПУ и процессорного (системного) блока ПБ. На рис. 1.2 показано, что УВМ управляет технологическим объектом ТО с параметрами Y посредством управляющих сигналов X . Пульт управления ПУ является основой рабочего места оператора, осуществляющего контроль работы АСУТП. Через ПУ поступает исходная информация в виде управляющих программ (УП), считываемых с магнитных дисков и дискет внешнего запоминающего устройства (ВЗУ). С помощью клавиатуры ПУ оператор может составлять и

корректировать управляющие программы и контролировать ход управляемого технологического процесса, а дисплей ПУ представляет оператору визуальную информацию о ходе процесса и о содержании УП. Наконец с помощью принтера производится распечатка отчетно-справочной информации о выполнении производственных заданий. Обмен информацией в УВМ осуществляется через стандартные устройства ввода-вывода УВВ, состоящие из параллельного и последовательного интерфейсов (портов), причем для связи внутри ПБ обычно используется параллельный интерфейс. Через последовательный интерфейс реализуется связь с отдаленными корреспондентами и, прежде всего, с ЦУВМ, если она есть.

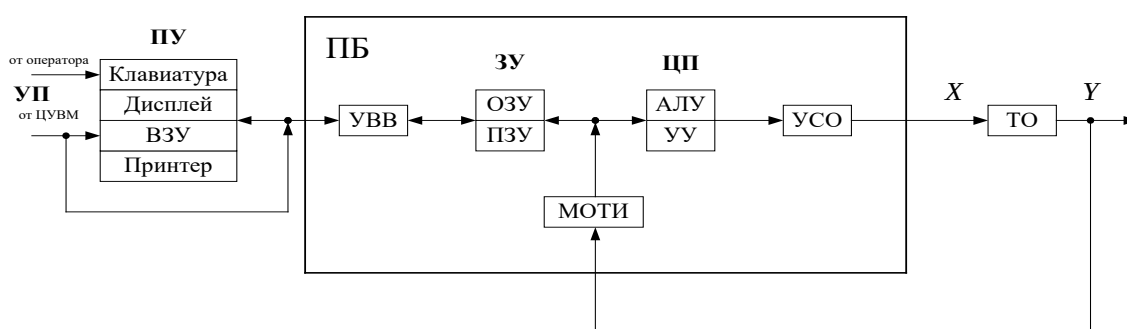


Рис. 1.2. Структура УВМ в составе АСУТП

Информация, поступающая в ПБ с пульта управления или непосредственно от ЦУВМ через УВВ, запоминается в устройствах памяти (ЗУ), состоящих из постоянного (ПЗУ) и оперативного (ОЗУ) запоминающих устройств. В ПЗУ содержится операционная система УВМ, инструментальное программное обеспечение для создания УП, сами УП и общие сведения об управляемом технологическом объекте. В ОЗУ хранятся управляющие программы, находящиеся в работе, и текущая информация о ходе реализуемого технологического процесса, о состоянии технологического оборудования и самой УВМ. Основным устройством, осуществляющим переработку поступающей информации в УВМ и выдачу управляющих сигналов, является центральный процессор (ЦП), состоящий из арифметико-логического (АЛУ) и управляющего (УУ) устройств. АЛУ осуществляет арифметическую и логическую обработку информации с выработкой управляющих сигналов, а УУ определяет, какие арифметико-логические операции и в каком порядке должно реализовать АЛУ в соответствии с заданной программой. Специфическими устройствами, отличающими УВМ от обычных ЭВМ, являются устройства связи с ТО (УСО) и модули обработки технологической информации (МОТИ).

УСО – это модули прямой связи управления. Они преобразуют приходящие с процессора управляющие сигналы, чтобы согласовать их с входными цепями ТО, в то время как МОТИ преобразуют приходящие с ТО сигналы обратной связи (сигналы Y) о параметрах ТО. Если рассматриваемая на рис.1.2 УВМ является для данного ТО центральной, то она управляет входящими в состав ТО локальными УВМ. В таком случае и УСО, и МОТИ, показанные на рис.1.2, состоят из стандартных УВВ, объединяющих все УВМ данной АСУТП в информационную и управляющую локальную сеть. Если же рассматривать УВМ, показанную на рис.1.2, как локальную, то УСО должны обеспечивать согласование управляющих сигналов УВМ со входными цепями различных дискретных цифровых и непрерывных (аналоговых) управляющих устройств нижнего уровня управления, на котором обычно производится управление электроприводами (см.рис.1.1).

Основные функции модулей УСО в составе ЛУВМ таковы:

- усиление управляющих сигналов с соответствующим преобразованием их по уровню и по виду;
- преобразование кодов цифровых управляющих сигналов (параллельного кода в последовательный и др.);
- цифро-аналоговое преобразование сигналов перед подачей их на аналоговые управляющие устройства;
- потенциальное разделение цепей управления.

Что касается МОТИ, то в ЛУВМ они должны совершить обратное преобразование сигналов обратной связи, идущих от управляющих устройств нижнего уровня управления и от технологического оборудования, к виду, приемлемому для системы сигналов, циркулирующих в УВМ. МОТИ производят:

- согласование уровней дискретных сигналов обратной связи и УВМ;
- аналого-цифровое преобразование аналоговых сигналов обратной связи, поступающих от аналоговых управляющих устройств и исполнительных механизмов;
- преобразование кодов цифровых сигналов обратной связи (последовательного в параллельный и др.);
- потенциальное разделение цепей управления.

В качестве примеров преобразования выходных сигналов УВМ приведем преобразование дискретного управляющего сигнала уровня до 5 В и до 5 мА тока (но предназначенного для включения контактора переменного тока) в переменное напряжение 110 В с током до 2 А, а также преобразование цифрового сигнала, предназначенного для управления электроприводом, в стандартное задающее напряжение в пределах 0 – 10 В. Примером преобразования сигналов обратной связи может служить преобразование сигналов

конечных выключателей, переключающих постоянное напряжение 24 В, в стандартный сигнал УВМ напряжением до 5 В. Таким же примером может быть счет импульсов датчиков перемещения, в результате которого величина перемещения фиксируется в УВМ в виде числа отсчитанных импульсов.

Устройства УСО и МОТИ выполняются в виде модульных конструкций, объединяющих в себе несколько каналов одностипных преобразователей, таких как преобразователи уровней, аналого-цифровые преобразователи и пр. При необходимости такие устройства строятся на базе микропроцессоров, как, например, модули управления сервоприводами.

Контрольные вопросы к главе 1

1. Каковы основные задачи, решаемые АСУТП? Дайте определение АСУТП.
2. Что такое целевая функция и какова ее роль в функционировании АСУТП?
3. Опишите реализацию основных функций АСУТП путем стабилизации параметров.
4. Опишите реализацию основных функций АСУТП путем программного управления.
5. Каковы особенности автоматической оптимизации техпроцесса в течение его реализации?
6. В чем состоит иерархический принцип управления?
7. Опишите типичную трехуровневую иерархическую структуру АСУТП.
8. Что такое УВМ? Дайте определение и опишите структуру УВМ в составе АСУТП.
9. Опишите основное назначение центрального процессора и запоминающих устройств в составе УВМ.
10. Что такое УСО? Опишите основные функции УСО и МОТИ в составе УВМ. Приведите примеры.

Глава 2. Информационное обеспечение АСУТП

2.1. Энтропия как мера информации

Технологический процесс, протекающий под управлением АСУТП, сопровождается информационными процессами, протекающими внутри АСУТП. Информационные процессы включают в себя:

- прием и запоминание управляющих программ (УП);
- прием и запоминание текущей информации о параметрах техпроцесса и параметрах АСУТП;
- формирование управляющих сигналов, воздействующих на управляемый техпроцесс, и передачу их по каналам УСО;
- формирование информации о ходе техпроцесса и о состоянии АСУТП для верхнего уровня управления с целью передачи ее по запросам ЦУВМ или оператора.

С точки зрения информационных процессов будем рассматривать техпроцесс и управляющие им сигналы как последовательность запланированных (предусмотренных) *событий*. События – это факты реализуемого производственного процесса. Реализованное событие будем называть *исходом*. Описание происходящих событий будем называть *сообщениями*. Все рассматриваемые сообщения будем считать дискретными величинами, начиная с сигналов на включение и отключение какого-либо оборудования и вплоть до показаний

стрелочных измерительных приборов, дискретность которых определяется ценой деления шкалы. Соответственно *информацией* будем называть процесс формирования, пересылки и регистрации (запоминания) сообщений, а также зарегистрированную совокупность сообщений.

Смысл сообщений, содержательная сторона информации, определяется известными закономерностями хода событий. Здесь, однако, нас будет интересовать не смысл, а количественная оценка информации, объем записей, которые необходимо сделать для регистрации сообщений. Последний будет существенно зависеть от совокупности символов, применяемых для записи информации. Такая совокупность обычно имеет вид основания системы исчисления или алфавита. Если в качестве основания используется совокупность n различных символов, а каждое сообщение состоит из H элементов, то с ее помощью можно сформировать не более чем n^H различных сообщений об интересующих нас событиях. Тогда величину H можно определить из соотношения

$$H = \log_n M, \quad (2.1)$$

где M – количество возможных вариантов события, сообщения о котором нам необходимо сформировать в зависимости от того, какой вариант состоится. Величину H , равную количеству элементов сообщения и характеризующую минимальный объем сообщения о том, что один из интересующих нас M вариантов изучаемого процесса действительно имел место, принято называть *энтропией*.

Если при планировании техпроцесса предполагается, что его результат однозначно предопределен, что исход будет однозначным, то в формуле (2.1) полагаем $M=1$ и получим $H=0$. Однако однозначный результат гарантировать невозможно. Исход техпроцесса и даже его простейших операций зависит от множества случайных факторов, действие которых оценивают методами теории вероятности. Так, если все предполагаемые события равновероятны и образуют полную группу (т.е. при каждом повторении изучаемого процесса происходит хотя бы одно из них), то вероятность каждого из этих событий равна

$$P = \frac{1}{M},$$

и соотношение (2.1) может быть записано в виде

$$H = -\log_n P. \quad (2.2)$$

Здесь величина H выступает как мера априорной неопределенности любого из исходов изучаемого процесса. Чем меньше вероятность P (чем больше вариантов равновероятных исходов), тем больше энтропия процесса. Таким образом, энтропия может быть использована для оценки степени случайности техпроцесса или какой-либо из технологических операций, как мера

неупорядоченности процесса. Обычно изучаемые события не равновероятны, характеризуются вероятностями P_j , и их энтропия рассчитывается по формуле Шеннона:

$$H = - \sum_{j=1}^M P_j \log_n P_j, \quad (2.3)$$

– причем необходимо чтобы изучаемые события составляли полную группу, когда $\sum_{j=1}^M P_j = 1$.

Величина H , рассчитанная по формуле (2.3), является *математическим ожиданием*, т.е. средним значением минимального объема записей; которые необходимо сделать для фиксирования факта реализации одного из M интересующих нас событий. Запишем P_j каждого из этих событий в виде соотношения

$$P_j = \frac{a_j}{M},$$

$$\text{где} \quad 0 \leq a_j \leq M, \quad \sum_{j=1}^M a_j = M, \quad M \geq 2.$$

Подставив данное соотношение в формулу (2.3), получим

$$H = \log_n M - \frac{1}{M} \sum_{j=1}^M a_j \log_n a_j. \quad (2.4)$$

Если все рассматриваемые события равновероятны, то все $a_j = 1$ и соотношения (2.3) и (2.4) сводятся к соотношению (2.1). В остальных случаях величина H будет уменьшена на величину

$$\frac{1}{M} \sum_{j=1}^M a_j \log_n a_j > 0,$$

т.е. максимальное значение энтропии H соответствует равной вероятности событий изучаемого процесса. Величина энтропии как минимального объема сообщения используется для оценки количества информации. Поскольку фактическое распределение вероятностей исходов чаще всего неизвестно, то при определении объема запоминающих устройств и пропускной способности каналов связи обычно руководствуются формулой (2.1).

2.2. Количественная оценка информации

При введении понятия энтропии мы исходили из количества событий, о которых нам необходимо получить информацию. Таким образом, энтропия характеризует количество информации, получаемой путем исследования интересующего нас процесса, если мы можем получить в свое распоряжение сведения о *любом* из M интересующих нас событий. Возможен, однако, такой исход исследования (или измерений), когда о некоторых событиях из рассматриваемых M событий сведения фактически не будут получены. Допустим, что до проведения исследований

предусматривалась реализация одного из M_1 событий, а после проведения исследований выяснилось, что в данном случае могут быть реализованы лишь M_2 из предусмотренных ранее M_1 событий, но осталось невыясненным, какое именно из этих M_2 событий реализовано в действительности. Энтропия исследуемого процесса до проведения исследований могла достигать значения

$$H_1 = \log_n M_1 ,$$

а после проведения исследований уменьшилась до значения, не превышающего

$$H_2 = \log_n M_2 .$$

В таком случае количество I полученной информации определяется разностью между исходной и конечной энтропиями:

$$I = H_1 - H_2 . \quad (2.5)$$

Формула (2.5) дает обобщенную количественную оценку информации и соответствует формулировке: *количество информации равно убыли энтропии*. Величина исчерпывающей информации соответствует $H_2=0$ и равна исходной энтропии.

В качестве примера рассмотрим оценку количества информации, получаемой от аналогового измерительного прибора с длиной рабочей части шкалы, равной L , работающего с погрешностью Δ , заданной в единицах длины прибора.

Полагаем также, что допустимая погрешность δ в заданной измерительной системе меньше той, какую может обеспечить используемый прибор: $\delta < \Delta$. Исходя из заданной точности измерений прибор должен гарантировать

$$M_1 = L / \delta$$

точек измерения в пределах рабочей части шкалы, но фактически он может гарантировать только

$$M_2 = L / \Delta$$

точек измерения. Начальная энтропия измерений составляет

$$H_1 = \log_n \frac{L}{\delta} ,$$

а энтропия после измерений –

$$H_2 = \log_n \frac{\Delta}{\delta} .$$

Количество информации, получаемой от этого прибора, равно

$$I = \log_n \frac{L}{\delta} - \log_n \frac{\Delta}{\delta} = \log_n \frac{L}{\Delta} , \quad (2.6)$$

т.е. определяется его точностью во всех случаях, когда имеет место $\delta < \Delta$. При $\delta = \Delta$ величина H_2 достигает значения $H_2 = 0$. Это говорит о том, что измерения производятся с заданной точностью. Заданная точность измерений поддерживается и при $\delta > \Delta$, т.е. сохраняется $H_2 = 0$. Хотя измерительный прибор при $\delta > \Delta$ может

обеспечить точность измерений больше заданной, это обстоятельство не является определяющим. Ведь величина исчерпывающей информации равна исходной энтропии, на которую сориентирована работа всего измерительного и управляющего комплекса, а точность первичного прибора является лишь одним из многих факторов, определяющих точность измерений. Иными словами, информация будет фиксироваться и преобразовываться в соответствии принятым объемом энтропии, а повышенная точность отдельных звеньев измерительного комплекса не будет учитываться.

Для практических расчетов количества информации по формулам (2.1-2.3) и по формуле (2.5) введем понятие единицы измерения информации. Примем во внимание, что в устройствах хранения, передачи или переработки информации обычно применяются элементы, в которых для записи информации используется только два символа, такие как высокий и низкий потенциалы (в электронных устройствах) или намагниченный и не намагниченный участки магнитных носителей информации. Эти символы обычно обозначают знаками 1 и 0. Использование только двух символов для записи информации соответствует $n=2$ в формулах (2.1-2.3), и в дальнейших рассуждениях мы так и будем полагать. Тогда окажется, что при исследовании процесса с двумя равновероятными исходами мы получим, в соответствии с формулой (2.1), $H_1=1(M_1=2)$ и $H_2=0(M_2=1)$, что дает, в соответствии с формулой (2.5), информацию $I=1$.

Информацию, которую получают из опыта с двумя равновероятными исходами, используют в качестве эталона (единицы) количества информации в устройствах, в которых запись информации производится с помощью только двух символов. Эту информацию называют *битом*, или *двоичной единицей информации*. Согласно формуле (2.3), количество информации, равное одному биту, можно получить и иным путем, из опыта со многими, но не равновероятными исходами. Опыт с двумя равновероятными исходами, (к которому приближается, например, подбрасывание монеты) является лишь *эталон*ом бита. Из определения бита следует, что максимальное количество информации, которая может храниться в электронном триггере или на элементарном намагниченном участке магнитной дискеты, равно одному биту. По тем же причинам *номинальная емкость запоминающих устройств (ЗУ), указанная в битах, есть максимальное количество информации, которое может храниться в этих ЗУ при условии, что все события, сообщения о которых зафиксированы в ЗУ, являются равновероятными*. Обычно, в ЗУ зафиксированы сообщения о событиях, которые не равновероятны. Ярким примером такого рода сообщений являются сообщения об отсутствии аварийной ситуации. Их вероятность при

нормальном ходе технологического процесса значительно выше вероятности поступления сигнала аварии. По указанным причинам фактический объем информации, хранящейся в ЗУ, значительно меньше номинального. Широко используются производные от бита единицы информации: 1 байт = 8 бит, 1 Кбайт = 2^{10} байт, 1 Мбайт = 2^{20} байт и др.

Особой единицей информации является *слово*. Слово – это количество информации, обрабатываемой в данном управляющем устройстве за один цикл работы устройства. Величина слова обычно кратна по отношению к байту: 1 байт, 2 байт, 4 байт. Если необходимо оценить лишь объем записи информации на электронных или магнитных носителях, то битом называют минимальный объем записи, соответствующий двоичной единице или двоичному нулю (элементарная ячейка памяти).

2.3. Кодирование информации

Кодированием мы будем называть процесс формирования сообщений о событиях, составляющих управляемый техпроцесс. Процесс кодирования состоит из двух этапов:

- представление информации в дискретной форме (аналогово-цифровое преобразование);
- представление дискретных сигналов в наиболее подходящем для решения задач управления техпроцессом виде (преобразование кодов).

Конкретный способ представления информации как системы соответствия между элементами сообщений и сигналами, их отображающими, называется *кодом*. Информация всегда представлена в каком-либо коде.

Первоначальный код, в котором информация поступает в управляющее устройство, определяется свойствами источника информации. Последний может быть либо датчиком какого-либо технологического параметра, либо некоторым ЗУ. В ЗУ информация уже представлена в дискретной форме, а сигналы датчика преобразуются в дискретную форму первичным преобразователем, свойства которого могут быть не согласованы с требованиями, предъявляемыми к информации, поступающей в управляющее устройство. Необходимое согласование производится соответствующим преобразователем кода, установленным на выходе первичного преобразователя.

По характеру взаимодействия с первичным преобразователем сигналы датчиков делятся на *дискретные и непрерывные* (аналоговые). Если первичный преобразователь выделяет в сигнале датчика только два (реже три) уровня, то такой сигнал считается *дискретным*. Характерными дискретными сигналами являются

сигналы о включении или выключении технологического оборудования. Если первичный преобразователь выделяет в сигнале датчика много (три и более) уровней, то такой сигнал считается непрерывным. Характерными непрерывными сигналами являются сигналы, регистрируемые стрелочными или самопишущими измерительными приборами. Если сигнал датчика классифицируется в качестве дискретного, то первичный преобразователь должен произвести лишь необходимое усиление сигнала и его гальваническую развязку с управляющим устройством. Если же сигнал датчика воспринимается в качестве непрерывного, то первичный преобразователь, осуществляющий аналого-цифровое преобразование, должен, помимо всего прочего, обеспечить различение необходимого количества уровней сигнала, т.е. должен иметь достаточную *разрешающую способность*, соответствующую величине энтропии данного элемента технологического объекта. Так, при использовании стрелочных измерительных приборов разрешающая способность обеспечивается выбором подходящей цены деления шкалы, а при использовании импульсных датчиков перемещения разрешающая способность обеспечивается выбором соответствующей величины перемещения контролируемого механизма между двумя соседними импульсами (цены импульса). Разрешающую способность называют также *шагом квантования по уровню* (см. рис. 2.1а).

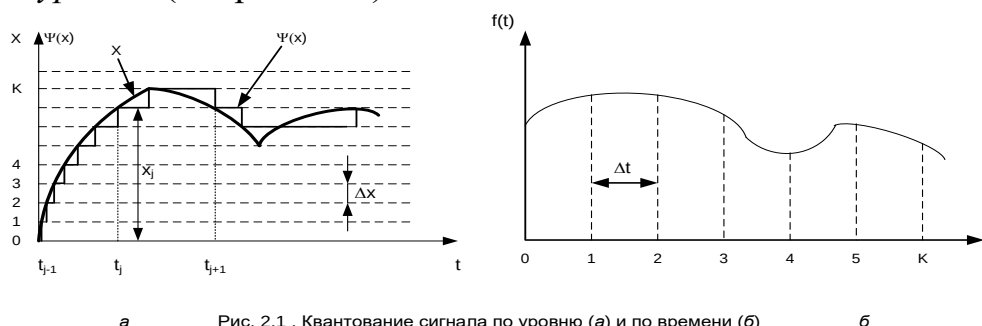


Рис. 2.1 . Квантование сигнала по уровню (а) и по времени (б)

При квантовании по уровню непрерывный сигнал $x(t)$ заменяется ступенчатой функцией $\Psi(x)$. Значение последней (рис.2.1а) изменяется только после изменения значения x на величину, не меньшую Δx , которая является шагом квантования и определяет разрешающую способность. При заданной допустимой относительной погрешности δ величина шага квантования Δx определяется из соотношения

$$\Delta x \leq \delta (x_{\max} - x_{\min}), \quad (2.7)$$

где $x_{\max}$ и $x_{\min}$ - максимальное и минимальное значения сигнала в заданном диапазоне его изменения.

Точность измерений зависит не только от разрешающей способности первичного преобразователя, но и от его быстродействия, которое у аналого-цифровых преобразователей (АЦП) определяется частотой отсчета измеряемой величины, т.е. *шагом квантования по времени* Δt (см. рис. 2.1б). Если принять шаг Δt , равным $t_j - t_{j-1}$ (рис. 2.1а), то в промежутке время между t_{j-1} и t_j измеренное значение x будет равно Δx , а в момент времени $t = t_j$ оно скачком увеличится до величины $x = x_j$. Видно, что большой шаг квантования по времени привел к значительному увеличению погрешности. При уменьшении шага квантования по времени точность измерения увеличивается. Чтобы точность измерений соответствовала шагу квантования по уровню Δx , величина шага квантования по времени Δt должна быть выбрана исходя из неравенства

$$\Delta t \left| \frac{dx}{dt} \right|_{\max} \leq \Delta x. \quad (2.8)$$

Соотношения (2.7) и (2.8) определяют требования к разрешающей способности и к быстродействию применяемых АЦП.

Отсчет и запоминание измеренной величины x обычно производятся либо в десятичной, либо в двоичной системе исчисления. Тип используемой системы исчисления зависит от примененной *системы кодирования*. Каждая система кодирования базируется на совокупности четко определенных, специфических сигналов одной физической природы, называемых *символами*. Полная совокупность символов, на которой базируется система кодирования, составляет ее *алфавит*. Конечная совокупность символов, входящих в принятую систему кодирования, называется *кодовой комбинацией*, *словом* или *числом* в зависимости от характера формируемого сообщения. Совокупности слов, правил их формирования и сочетания образуют разнообразные *языки кодирования*. В системах кодирования АСУТП широко применяется такие совокупности символов, как латинский и русский алфавиты, арабские цифры, знаки препинания, знаки математических операций и другие математические символы.

Для представления чисел обычно используются арифметические коды, в которых любое действительное число может быть представлено в виде

$$Q = \sum_{j=m}^l a_j n^j, \quad (2.9)$$

где: $n > 1$ - целое число, основание системы исчисления;
 m и l – произвольные числа, причем $m \geq l$;
 a_j - целое число, причем $0 \leq a_j \leq n - 1$.

Произведение $a_j n^j$ называют разрядом закодированного числа, причем n^j называют весом разряда, a_j - цифрой или цифровым значением разряда, а j – номером разряда. Если $m=l$, то закодированное число одноразрядное, состоит из одного разряда, а если все a_j равны нулю, то $Q=0$. Все разряды, у которых $j < 0$ входят в дробную часть закодированного числа, а остальные разряды составляют его целую часть. При записи чисел обычно опускают веса разрядов, а также нули перед первой значащей цифрой целой части числа и после последней значащей цифры дробной части числа, причем дробную часть числа отделяют от целой части запятой или точкой. После этого запись числа принимают более компактный вид:

$$Q = a_m a_{m-1} \dots a_1 a_0, a_{-1} \dots a_l. \quad (2.10)$$

В качестве примера рассмотрим запись десятичного числа ($n=10$), у которого $l=-5$, $m=-4$, $a_{-5}=9$, $a_{-4}=3$. По формуле (2.9) получим:

$$Q = 3 \cdot 10^{-4} + 9 \cdot 10^{-5},$$

что в обычной записи составит $Q=0,00039$.

В управляющих устройствах систем автоматизации наиболее распространена двоичная система исчисления и соответственно для записи числовой информации применяется двоичный арифметический код ($n=2$). При записи числа в двоичном арифметическом коде используют только две цифры: 0 и 1, так что закодированное число приобретает вид набора нулей и единиц, в котором дробная часть числа отделяется от целой части запятой. В качестве примера рассмотрим запись двоичного числа, у которого $m=2$, $l=-2$, $a_2 = 1, a_1 = a_0 = a_{-1} = 0, a_{-2} = 1$. по формуле (2.9) получим:

$$Q = 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 + 0 \cdot 2^{-1} + 1 \cdot 2^{-2} = 4 \frac{1}{4}.$$

Опустив обозначения весов разрядов, получим, в соответствии с формулой (2.10), запись данного числа в традиционном виде: $Q=100,01$.

Перевод целой части числа из десятичной в двоичную форму записи удобно производить методом деления. По этому методу исходное десятичное число последовательно делится на 2. Остатки от деления образуют двоичное число, причем старший разряд соответствует последнему остатку, а младший – первому. В качестве примера рассмотрим преобразование десятичного числа 25 в соответствующее двоичное число:

$$\begin{aligned} 25/2 &= 12 \text{ (остаток 1)} - \text{младший разряд} \\ 12/2 &= 6 \text{ (остаток 0)} \\ 6/2 &= 3 \text{ (остаток 0)} \\ 3/2 &= 1 \text{ (остаток 1)} \\ 1/2 &= 0 \text{ (остаток 1)} - \text{старший разряд} \end{aligned}$$

Полученное двоичное число-
 $Q = 11001 = 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0$.

Аналогично в двоичную форму переводится методом *умножения* дробная часть десятичного числа. В случае необходимости различения двоичных и десятичных (Decimal) чисел в конце двоичного числа ставят букву *B* (Binary).

Широкое применение при проектировании микропроцессорных управляющих устройств нашел шестнадцатеричный (Hexadecimal) арифметический код ($n=16$, HEXCODE). Для записи числовой информации в этом коде необходимо иметь 16 цифр. Для обозначения первых 10 из них используются арабские цифры, а остальные 6 цифр имеют следующие обозначения: 10-A, 11-B, 12-C, 13-D, 14-E и 15-F. Запись чисел в 16-ричном коде производится, как и в системах исчисления с другими основаниями, в соответствии с формулами (2.9) и (2.10). В качестве примера рассмотрим 16-ричное число, у которого $m=2$, $l=0$, $a_2 = 4$, $a_1 = E$ и $a_0 = A$. По формуле (2.9) получим:

$$Q = 4 \cdot 16^2 + 14 \cdot 16^1 + 10 \cdot 16^0 = 1024 + 224 + 10 = 1258.$$

Запись в 16-ричной форме дает, согласно формуле (2.10), $Q=4EA$.

От 16-ричного кода нетрудно перейти к двоичному коду. Для этого каждую цифру 16-ричного числа заменяют кодовой комбинацией из 4 двоичных символов, записанной в двоичном арифметическом коде и численно равной заменяемой цифре. Из цифр 16-ричного кода предыдущего примера получим: $4=0100$, $E=14=1110$, $A=10=1010$. Число Q в целом, записанное в двоичном коде, с учетом отбрасывания нуля, стоящего перед первой значащей цифрой, будет выглядеть как:

$$Q=10011101010.$$

Для перехода от двоичной формы записи числа к 16-ричной достаточно разбить двоичное число на группы по 4 символа в каждой, начиная с младшего разряда двоичного числа, с последующей заменой каждых четырех двоичных символов на один 16-ричный, и дополняя при необходимости нулями группу старших разрядов. Так, разбивка предыдущего числа, с последующей заменой четырехсимвольных групп 16-ричными цифрами, дает: $Q=0100\ 1110\ 1010=4EA$.

Простота перехода от двоичной к 16-ричной форме записи чисел и обратно обусловила применение 16-ричного кода для компактной записи двоичных кодовых комбинаций, применяемых в микропроцессорных системах управления.

2.4. Двоичные коды

2.4.1. Экономичность двоичного кодирования

Информация, поступающая в УВМ, должна быть преобразована и представлена в двоичном формате, т.е. в виде того или иного двоичного кода.

Двоичным кодом называется любая система кодирования, в которой используются только два символа: 0 и 1. Обработка информации процессором УВМ производится только в двоичных кодах. Это объясняется экономичностью построения управляющих устройств на элементах, имеющих при нормальной эксплуатации только два различных состояния (да-нет, открыто-закрыто, включено-выключено, намагничено - не намагничено, высокий потенциал - низкий потенциал и т.п.). В обобщенном виде одно из указанных состояний обозначается цифрой 0, а другое – цифрой 1. Обоснование данного положения проведем исходя из того, что при обработке числовой информации сложность запоминающих устройств и объем пересылаемых сообщений определяются произведением основания системы исчисления на количество разрядов максимального записываемого или пересылаемого числа. Например, для обеспечения записи в десятичной форме любого из чисел величиной не более 1000 требуется не менее трех десятипозиционных устройств, имеющих в сумме 30 рабочих контактов. В то же время для записи каждого из тех же чисел в двоичной форме требуется не более 10 двухпозиционных устройств ($2^{10}=1024$) с общим количеством рабочих контактов, равным 20. Преимущество двухпозиционных устройств памяти над десятипозиционными в данном случае несомненно. В общем случае максимальное число различных сообщений, каждое из которых можно зафиксировать с помощью m n -позиционных запоминающих устройств (n -основание системы исчисления), равно:

$$M = n^m. \quad (2.11)$$

Отсюда становится ясным, что количество разрядов m запоминаемого числа не может превышать (при основании системы исчисления, равном n) величины

$$m = \frac{\ln M}{\ln n},$$

а сложность запоминающего устройства, пропорциональная произведению nm , определится из соотношения:

$$nm = n \frac{\ln M}{\ln n}. \quad (2.12)$$

Продифференцировав правую часть соотношения (2.12) по n и приравняв полученную производную нулю, получим уравнение

$$\ln n_{opt} - 1 = 0,$$

из которого определится оптимальное значение $n_{opt} = e \approx 2.72$, которому соответствует наименьшая сложность запоминающих устройств. Из ближайших к оптимальному значений $n=2$ и $n=3$

лучшим является $n=2$, так как двоичные элементы имеют относительно большую (в расчете на один рабочий контакт) простоту технической реализации по сравнению с трехпозиционными элементами. Проиллюстрируем данное положение на примере схем двухпозиционного (RS -триггер) и трехпозиционного запоминающих устройств, приведенных на рис. 2.2.

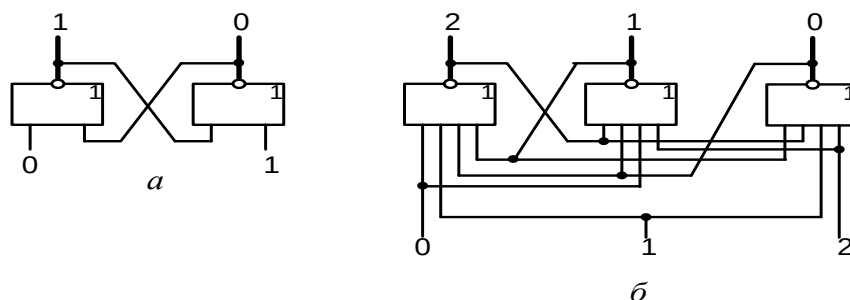


Рис. 2.2.

Схемы двухпозиционного (a) и трехпозиционного ($б$) запоминающих устройств

Схемы рис.2.2 построены на элементах ИЛИ-НЕ, выходной сигнал элемента ИЛИ-НЕ равен единице только при наличии нулевых сигналов на всех его входах. Информационные сигналы 0,1 и 2 поступают на входы схем a и $б$ (рис.2.2) в виде единичных импульсов. При поступлении единичного импульса на вход 0 выход 0 устанавливается в единичное состояние, а при поступлении единичного импульса на вход 1 выход 1 устанавливается в единичное состояние. Остальные выходы соответственно переходят в нулевое состояние. Состояние выходов сохраняется до тех пор, пока не поступит единичный импульс на вход, номер которого не совпадает с номером выхода, находящегося к тому времени в единичном состоянии. Одновременная подача единичных сигналов на два или три входа не допускается. На рис. 2.2 видно, что количество связей, необходимых для реализации трехпозиционного устройства (в расчете на один рабочий контакт) более чем в 1.5 раза больше, чем у двухпозиционного.

2.4.2. Арифметические двоичные коды

В предыдущем параграфе был уже рассмотрен двоичный арифметический код, применяющийся для хранения чисел и в вычислениях. Его называют также *прямым кодом*. При использовании прямого кода знак числа кодируется с помощью специального знакового разряда, принимающего значение 0 для положительных и 1-для

отрицательных чисел. Применение прямого кода неудобно тем, что при сложении двух чисел, имеющих разные знаки, операция сложения должна быть заменена операцией вычитания меньшей величины из большей с присвоением результату знака большей величины. Более удобен в вычислительных операциях двоичный *дополнительный код*, который, также являясь арифметическим, позволяет заменить вычитание на обычное сложение. При формировании дополнительного кода исходят из понятия *дополнения*. Дополнением целого числа Q называют разность:

$$M = 2^m - Q, \quad (2.13)$$

где m – максимальное количество разрядов двоичного числа в данном вычислительном устройстве, включая знаковый разряд. При записи числа Q в дополнительном коде согласно формуле (2.13) значение кодовой комбинации M , отображающий величину Q , если ее интерпретировать как арифметический код без знака, может изменяться от 0 до $2^m - 1$. При этом значения M от 0 до $2^{m-1} - 1$ рассматривают как положительные числа (в знаковом разряде находится 0), а значения от $2^{m-1} - 1$ до $2^m - 1$ – как отрицательные числа, модуль которых составляет от 0 до $2^{m-1} - 1$ (в знаковом разряде находится единица). Правомерность данной интерпретации доказывается тем, что сумма значения Q и его дополнения M , как следует из соотношения (2.13), всегда равна 2^m , т.е. единице старшего разряда. Последний не отображается, так как находится за пределами принятой разрядной сетки, а во всех m разрядах нашего кода при суммировании Q и M будут записаны нули. Следовательно, число Q и его дополнение M отображают числа, равные по модулю и противоположные по знаку.

Представив соотношение (2.13) в виде

$$M = ((2^m - 1) - Q) + 1, \quad (2.13')$$

получим простую процедуру формирования дополнения числа Q . Предварительно заметим, что

$(2^m - 1) - Q$ – это *инверсия (обратный код)* числа Q , которую получают заменой единиц нулями, а нулей единицами. Следовательно, для получения дополнения необходимо исходное число инвертировать и прибавить к полученной инверсии единицу. Следствия:

- если число Q представлено в дополнительном коде, то его дополнение тоже оказывается представленным в дополнительном коде;
- положительные числа записываются в прямом и в дополнительном кодах одинаково;
- для изменения знака числа, записанного в дополнительном коде, необходимо это число инвертировать, а затем прибавить к полученной инверсии единицу;

- для перевода отрицательного числа, записанного в прямом коде, в дополнительный код и обратно, необходимо инвертировать лишь значащую часть числа и прибавить к полученному результату единицу.

- Сложение отрицательных чисел в дополнительном коде производится по тем же правилам, что и сложение положительных чисел. Вычитание производится путем сложения уменьшаемого и дополнительного кода вычитаемого.

Пример 2.1. Сложить числа $-(2^{14}+1)$ и 17 при $m=16$, а затем из полученного результата вычесть 17.

Решение:

1) Запишем $-(2^{14}+1)$ и 17 в прямом коде при $m=16$;

$-(2^{14}+1)=1100\ 0000\ 0000\ 0001$;

$17=0000\ 0000\ 0001\ 0001$.

2) Перевод заданных чисел в дополнительный код:

$-(2^{14}+1)=1011\ 1111\ 1111\ 1110+1=1011\ 1111\ 1111\ 1111$.

Дополнительный код положительного числа 17 совпадает с его прямым кодом.

3) Сложение $-(2^{14}+1)$ и 17:

1011 1111 1111 1111

+

0000 0000 0001 0001

1100 0000 0001 0000

4) Перевод результата в прямой код:

$1011\ 1111\ 1110\ 1111+1=1011\ 1111\ 1111\ 0000$.

5) Изменение знака числа 17:

$-17=1111\ 1111\ 1110\ 1110+1=1111\ 1111\ 1110\ 1111$.

6) Заданное вычитание числа 17:

1100 0000 0001 0000

+

1111 1110 1110 1111

1 1011 1111 1111 1111

Поскольку переполнения нет, то перенос из старшего разряда не учитывается. В остальном результат точно совпадает с дополнительным кодом числа $-(2^{14}+1)$.

2.4.3. Неарифметические двоичные коды

Наряду с арифметическими кодами в системах автоматизации применяются и неарифметические двоичные коды. В частности, в устройствах индикации и связи с оператором применяются *двоично-десятичные коды*, предназначенные для отображения арабских цифр. Обычно для отображения цифр используются кодовые комбинации, состоящие из четырех двоичных символов. Наибольшее применение нашел двоично-десятичный код 8–4–2–1, в котором каждая

десятичная цифра записывается в арифметическом двоичном коде таким образом, что численное значение записанной четырехсимвольной комбинации равно отображаемой цифре, а именно:

0=0000, 1=0001, 2=0010, 3=0011, 4=0100, 5=0101, 6=0110, 7=0111, 8=1000, 9=1001.

Запись чисел в двоично-десятичном коде производится в десятичной системе исчисления, но вместо обычных цифр используются вышеприведенные комбинации нулей и единиц.

Из других неарифметических двоичных кодов, применяющихся для записи чисел, наиболее интересны циклические коды Грея, нашедшие применение в кодовых датчиках положения.

Кодами Грея называются двоичные коды, у которых соседние кодовые комбинации отличаются лишь одним кодовым символом. Соседними кодовыми комбинациями мы здесь называем кодовые комбинации, предназначенные для отображения чисел, которые, будучи записанными в каком-либо арифметическом коде, отличаются друг от друга на единицу младшего разряда, принятую в данной системе кодирования. Чтобы код Грея был циклическим, кодовая комбинация, отображающая максимальное число данного кода, выполняется соседней с кодовой комбинацией, отображающей минимальное число. Понятие соседней кодовой комбинации становится совершенно ясным, если мы представим круговой датчик положения, позиции которого последовательно отображены в каком-либо коде Грея. Тогда соседние кодовые комбинации будут отображать соседние позиции датчика положения, причем последняя позиция окажется соседней с первой. Количество двоичных символов m , отображающих код Грея, определяется количеством положений Q проектируемого датчика положения из соотношения

$$Q \leq 2^m.$$

Очевидно, что если, например, $Q=1000$, то $m \geq 10$.

В таблице 2.1 приведен пример четырехсимвольного кода Грея, способного отобразить не более 16 позиций кодового датчика, и одновременно приведен четырехразрядный арифметический двоичный код, отображающий те же позиции. Видно, что в коде Грея при переходе от позиции к позиции изменяется только один символ, а в арифметическом коде в некоторых позициях изменяются сразу несколько символов, причем при переходе от позиции 7 к позиции 8 и от последней позиции к первой изменяются все символы сразу.

Таблица 2.1

Позиции датчика	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Код Грея	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0
	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0
	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0
	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	1
Арифметический код	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Неточности в нанесении кодовой маски и установке считывающих элементов приводят, при использовании арифметического кода, к появлению ложных сигналов на границах между соседними позициями, если при переходе от позиции к позиции меняются сразу несколько символов. Так, если при переходе от позиции 7 к позиции 8 первым произойдет переход от нуля семерки к единице восьмерки, возникнет кратковременный сигнал о нахождении датчика в позиции 15. При применении кода Грея появление ложных сигналов такого рода невозможно. Использование кодов Грея повышает надежность аналого-цифрового преобразования в датчиках положения.

Важнейшую роль в организации взаимодействия между человеком-оператором и УВМ играют двоичные коды обмена информацией (см. приложение 1). Из них наиболее распространенным является код ASCII (American Standard Code for Information Interchange, табл. П1.1), играющий роль международного стандарта. Это семибитовый код, т.е. код, в котором каждая кодовая комбинация состоит из семи двоичных символов, что позволяет получить всего 128 (2^7) различных кодовых комбинаций. С помощью кода ASCII (произносится *аски*) кодируются 26 букв (заглавных и строчных) современного латинского (английского) алфавита, 10 арабских цифр, знаки математических операций, знаки препинания, различные специальные символы (например, %) и управляющие символы (табл. П1.3).

Обработка кодовых комбинаций ASCII производится байтами. Благодаря этому в составе каждой кодовой комбинации ASCII появляется дополнительный бит, который может быть использован либо для расширения кода (“расширенный ASCII”), либо для контроля правильности кодовых комбинаций. В байте кода ASCII дополнительный бит занимает место старшего бита. При расширении кода ASCII его основная часть, т.е. символы, отображенные в таблице П.1.1, кодируется нулевым значением старшего бита ($b_7=0$), а дополнительные символы, полученные при расширении ASCII, кодируются единичным значением старшего бита. Именно таким путем сформированы символы отечественного кода обмена информацией КОИ-8 (см. табл. П1.2), применение которого нормировано по ГОСТ Р34.303-92, построенным в соответствии с международным стандартом ISO4873-86. В коде КОИ-8

представлены, кроме символов базового ASCII, буквы русского алфавита и дополнительные управляющие символы. Соответствие русских управляющих символов КОИ-8 английским управляющим символам ASCII можно проследить по таблице П1.3. Заметим, что в таблицах П1.2 и П1.3 приведены только 16-ричные коды, но их нетрудно перевести в двоичную форму, если воспользоваться рекомендациями, приведенными в §2.3. А десятичные цифры в коде ASCII (табл. П1.1) закодированы по принципу двоично-десятичного кодирования, так как четыре младших бита ($b_0 - b_3$) кодов десятичных цифр соответствуют двоично-десятичному коду 8-4-2-1.

Символы кодов обмена информацией составляют основу алфавитов всех современных систем кодирования. По этой причине большинство символов кода ASCII отображено на клавишах стандартной клавиатуры любого компьютера. Типичным примером применения символов ASCII являются символы кода ISO-7 (см. приложение 4), предназначенного для программного управления оборудованием. В этом коде, которому соответствует отечественный стандарт ГОСТ 20999-83, используются исключительно символы ASCII. Код ISO-7 является 8-битным, причем свободный старший бит в каждом байте кода ISO-7 используется для контроля кода по четности (см. §2.5). Заметим, что код ISO-7 не может считаться двоичным кодом, так как его слова состоят не из двоичных символов (не из наборов нулей и единиц), а из символов ASCII. Аналогично строятся *языки ассемблеров*, которые позволяют при программировании в машинных кодах пользоваться алфавитно-цифровыми мнемоническими обозначениями кодов операций и команд процессоров УВМ, присваивать символические имена их рабочим регистрам и формировать удобные схемы адресации. Тем более не отображаются непосредственно в двоичных кодах алгоритмические языки высокого уровня.

2.5. Передача информации по каналам связи

2.5.1. Промышленные информационные сети

Передача информации по каналам связи решает следующие задачи (см.рис.1.2):

- обмен информацией между пультом управления и процессорным блоком УВМ;
- обмен информацией между УВМ и технологическим объектом управления (ТО);
- обмен информацией между локальной и центральной УВМ.

Обмен информации между ПУ и ПБ включает в себя организацию связи между человеком-оператором и АСУТП с помощью клавиатуры, дисплея, принтера и различных средств индикации. Кроме того обычно реализуется связь с ВЗУ,

выполненными на магнитных дисках и дискетах. Связь с технологическим объектом обеспечивается, прежде всего, через модули выдачи и приема дискретных сигналов, а также с помощью специализированных модулей. В частности, выдача и прием непрерывных сигналов обеспечивается с помощью модулей ЦАП и АЦП.

Обмен информацией в УВМ осуществляется при помощи двоичных кодовых комбинаций фиксированной величины, называемых словами. Величина слов, обрабатываемых ЛУВМ, обычно равна 1-2 байт. Передача дискретной информации между различными уровнями управления АСУТП производится словами или объединениями слов, называемыми кадрами. Различают два основных способа передачи информации:

- параллельная передача слова информации внутри УВМ (все биты слова передаются одновременно, параллельно);
- последовательная передача слова или кадра информации между УВМ и исполнительным устройством, между ЛУВМ и ЦУВМ, когда биты информации, передаются один за другим в заданной последовательности.

Совокупность правил обмена информацией между двумя соседними уровнями управления, а также совокупность проводов и иных технических средств, обеспечивающих такой обмен информацией, называется *интерфейсом*. В настоящее время в АСУТП применяют почти исключительно *стандартные интерфейсы*, т.е. интерфейсы, которые обеспечивают заданный международным или иным стандартом порядок обмена информацией независимо от технической базы, на которой реализованы конкретные модификации того или иного интерфейса. При параллельной передаче слов информации используется *параллельный интерфейс*. В нем число проводов, по которым передается информация, должно быть не меньше числа битов передаваемого слова. Так, для параллельной передачи слова в два байта необходимо иметь шину связи, с учетом общего провода, из не менее чем 17 проводов. В то же время для последовательной передачи информации посредством *последовательного интерфейса* достаточно иметь только два связанных провода независимо от величины передаваемых слов. Многопроводную шину трудно уберечь от помех, ее стоимость много выше стоимости двухпроводной связи. Поэтому параллельную передачу информации применяют при расстоянии между объектами связи, не превышающем 1-2 метра, преимущественно внутри УВМ.

Обмен информацией как с помощью параллельного интерфейса, так и с помощью последовательного может быть двух видов: синхронный и асинхронный. При *синхронном* обмене информацией время и темп выдачи слов управления и записи сигналов обратной

связи определяется исключительно тактовыми импульсами УВМ. При *асинхронном* обмене информацией время и темп выдачи и приема информации определяется также и сообщениями, поступающими от технологического объекта, оператора или от ЦУВМ. Обмен информацией внутри УВМ строится по синхронному принципу, но для связи с внешними устройствами (ВУ) широко применяется асинхронный обмен.

Пропускная способность канала связи определяется соотношением

$$C=S/T, \quad (2.14)$$

где S – величина слова (кадра) информации, бит;

T – время цикла передачи, с.

Передача слова информации сопровождается управляющими сигналами, один из которых инициирует начало передачи слова, а второй информирует о конце передачи (приема) слова. Для управления передачей слова информации в параллельном интерфейсе должно быть не менее двух дополнительных связанных проводов, а в последовательном интерфейсе должна быть предусмотрена передача не менее двух (стартового и стопового) дополнительных битов информации.

Отдельные производственные модули (ПМ) автоматизируемого объекта обычно разбросаны на достаточно большой производственной территории. Каждый из них, имея собственную систему управления, объединяется для согласованной работы с другими ПМ посредством ЦУВМ. Большие расстояния отдельных узлов автоматизируемых объектов друг от друга делают целесообразным применение исключительно последовательных интерфейсов как для связи между ЦУВМ и ЛУВМ, так и для обмена информацией между ЛУВМ, которые реализуют управление ПМ, и исполнительными устройствами. АСУТП, объединенная устройствами последовательной связи, принимает вид *распределенной системы управления*. Совокупность программ, устройств и проводов, которые обеспечивают обмен информацией в АСУТП, называют *промышленной информационной сетью*. Способ соединения отдельных устройств в такой сети составляет её топологию. Управляющие устройства, входящие в состав сети, называются её *узлами*, а совокупности проводов, соединяющих узлы, называют разделяемым (между составляющими узлами) физическим каналом или *разделяемой средой передачи данных (shared media)*. Наиболее распространенными сетевыми топологиями являются звезда, кольцо и шина.

О соединении в *звезду*(star) говорят в тех случаях, когда порядок обмена информацией определяется одним управляющим устройством, которое в таком случае называется *центральный узлом*,

или *master* (ведущий). Остальные устройства (узлы), участвующие в обмене информацией, называются *slave* (ведомый). Такой порядок характерен для обмена информацией между ПЛК и исполнительными устройствами, где отказ центрального узла ПЛК не скажется на функционировании АСУТП в целом.

Кольцевой структурой, или *кольцом* (ring), называют такой порядок обмена информацией, когда роль ведущего узла (*master*) играют по очереди все управляющие устройства, участвующие в обмене информацией. При этом каждое устройство получает возможность инициировать необходимый ему обмен информацией только в момент получения специального сообщения, называемого *маркером* (token), и на ограниченное время. Одна из стандартных сетевых технологий, реализующих при обмене информацией принцип передачи маркера по кольцу, так и называется: Token Ring («маркерное кольцо»).

Шинная топология соответствует объединению всех участвующих в обмене информацией устройств посредством общей линии связи, называемой *шиной* (bus), или *магистралью*. В условиях распределенной системы управления, каковой является АСУТП, общая шина является малопроводной, предназначенной для одновременного обмена информацией лишь между какими-либо двумя устройствами, причем в последовательном коде. Физическое соединение через общую шину может соответствовать логическому соединению в звезду в том случае, когда один из участников обмена по общей шине является *ведущим* (*master*), т.е. определяющим порядок доступа к ней, порядок «захвата» шины. С другой стороны, если доступ к шине определяется передачей маркера устройству, которое путем предварительного распределения шинных ресурсов признано «достойным» играть роль ведущего в текущем цикле обмена информацией, то имеет место логическое соединение участников обмена информацией в кольцо.

Наиболее распространенным стандартным способом распределения шинных ресурсов в собственно шинной топологии является метод коллективного доступа с наблюдением за несущей и обнаружением коллизий, имеющий обозначение CSMA/CD (Carrier Sense Multiple Access with Collision Detection). Этим методом организует доступ к шинным ресурсам наиболее популярная сетевая технология Ethernet. Согласно этому методу любое устройство из подключенных к общей шине может начать передачу данных, если в течение условленного промежутка времени передачу данных не начало никакое другое устройство. О том, что шина свободна, свидетельствует отсутствие на ней напряжения несущей частоты. При появлении несущей частоты все устройства, подключенные к общей шине, осуществляют прием начавшейся передачи. Пока

устройство, начавшее передачу, не закончит её, остальные устройства, подключенные к общей шине, не могут войти в режим передачи. Время передачи не может превышать заданного предела, чтобы не было чрезмерной задержки передачи данных для всех устройств, претендующих на режим передачи. Если случайным образом окажется, что одновременно начали передачу сразу два устройства, то такой режим является ненормальным и называется коллизией. Коллизия обнаруживается по повышенному напряжению несущей. В случае коллизии все устройства, начавшие передачу, прекращают её на некоторое время. Величина выдержки времени после коллизии выбирается для каждого устройства различной, с помощью генератора случайных величин. Благодаря этому по окончании выдержки времени коллизия между устройствами, ранее бывшими в состоянии коллизии, вновь не возникает.

Метод CSMA/CD не гарантирует доступа к общей шине при большой загруженности информационной сети из-за большого количества коллизий, но он предоставляет возможность быстрого доступа при малой загруженности сети. Метод с передачей маркера, напротив, гарантирует каждому узлу доступ к общей шине, но лишь при наступлении очереди на передачу информации после обхода кольца. Поэтому сетевые технологии с применением метода CSMA/CD более эффективны при малой загрузке сети (менее 30 %), а при большой загрузке целесообразно использовать системы с передачей маркера.

Технология обмена информацией в АСУТП имеет стандартизованную трехуровневую структуру, построенную в соответствии со стандартной семиуровневой моделью OSI (Open Systems Interconnection) взаимодействия открытых систем, предложенной Международной организацией по стандартизации (ISO). Обычно информационные сети АСУТП поддерживают 1-й, 2-й и 7-й уровни модели OSI, т.е. физический, канальный и прикладной уровни.

На *физическом уровне* (Physical Layer) информация передается побитно. Биты информации передаются последовательно, один за другим по конкретным физическим каналам связи (коаксиальный кабель, витая пара, оптоволоконный кабель и др.). При этом определяются характеристики сигналов, тип кодирования битовых последовательностей, параметры сред передачи данных, стандартизируются разъемы и назначение контактов.

На *канальном уровне* (Data Link Layer) биты группируются в наборы, называемые *кадрами* (frames). Канальный уровень обеспечивает корректную передачу кадра путем добавления специальных битов, обеспечивающих опознавание кадра и контроль правильности его передачи. Обмен информацией на канальном

уровне зависит от принятой сетевой технологии: Modbus, Profibus, Ethernet, Token Ring, FDDI (оптоволоконная технология) и др.

На *прикладном уровне* (Application Layer) обеспечивается связь прикладной программы управления техпроцессом с управляемыми исполнительными устройствами через нижележащие уровни обмена информацией.

Описание взаимодействия одноименных уровней обмена информацией, находящихся в разных узлах, принято называть *протоколом*. Интерфейсом же, при построении системы обмена информацией в соответствии с моделью OSI, называют стандарт взаимодействия между соседними уровнями обработки информации в одном узле. Соответственно стандарт обмена информацией между канальным и физическим уровнями OSI, называют *физическим интерфейсом*.

2.5.2. Последовательные интерфейсы по стандартам RS232C и RS485

В качестве примера рассмотрим применение весьма распространенных в контроллерном программном управлении и в УЧПУ физических последовательных интерфейсов по стандартам RS232C и RS485 (RS – Recommended Standard). Эти стандарты были разработаны американской ассоциацией EIA (Electronics Industries Alliance). Основное назначение интерфейса RS232C – обеспечение связи УВМ с управляемым объектом по методу PtP (Point-to-Point – точка-точка). По стандарту RS232C функционирует COM – порт (Communication Port) персонального компьютера. Поскольку RS232C является радиальным интерфейсом типа PtP, то понятие адреса в нем отсутствует. Об основных функциях системы передачи по стандарту RS232C дает представление описание сигналов данного интерфейса, приведенное в таблице 2.2.

Таблица 2.2. Назначение основных сигналов интерфейса RS232C

Обозначение	Назначение
SG(Signal Ground)	Сигнальная земля
TD(Transmitted Data)	Данные, передаваемые в оконечное устройство
RD(Received Data)	Данные, принимаемые от оконечного устройства
DTR(Data Terminal Ready)	Готовность контроллера к обмену данными
DSR(Data Set Ready)	Готовность оконечного устройства к обмену данными
RTS(Request To Send)	Запрос передачи данных
CTS(Clear To Send)	Готовность регистра

Согласно таблице 2.2 информация передается через интерфейс RS232C в оконечное устройство в виде импульсов напряжения между проводами TD и SG, а в обратном направлении – по проводу RD относительно того же SG. Таким образом, обеспечивается возможность реализации дуплексного режима связи, т.е. одновременной передачи информации в обоих направлениях. Уровень логического нуля передающего устройства TD стандартизируется в пределах от +5В до +12В, а уровень логической единицы – от -12В до -5В. Соответственно уровни потенциала приемного сигнала RD должны находиться в пределах $\pm(3\div 12)$ В. Повышенные уровни напряжений сигналов RS232C (а также RS485) способствуют повышенной помехозащищенности цеховых систем автоматизации, использующих эти интерфейсы.

Обмен информацией в интерфейсе RS232C начинается с установление сигнала DTR=1, означающем готовность контроллера интерфейса к обмену данными. Если в ответ оконечное устройство также объявит о готовности к обмену данными путем установки DSR=1, то передача данных в оконечное устройство будет осуществляться при наличии сигналов RTS=1 и CTS=1 одновременно. Процесс передачи информации проходит либо в асинхронном, либо в синхронном режиме. В асинхронном режиме обмен информацией производится побайтно. Для синхронизации передающего узла интерфейса с приемной частью, в поток информации вставляют стартовый бит перед передачей каждого байта информации, а в конце каждого байта вставляют стоповый бит. Стартовый бит соответствует уровню логического нуля, а стоповый бит – уровню логической единицы. В течение всего времени паузы пока не начнется передача нового байта информации, на линии передачи поддерживается состояние логической единицы. В синхронном режиме оформление кадра информации (побайтно или в иной форме) и его адресация производятся на канальном уровне обмена информацией.

Максимальное быстродействие RS232C соответствует скорости передачи 115,2 кбит/с при длине линии связи до 15 м, поскольку его линии связи несимметричны. Если необходимо вести обмен информацией способом последовательной передачи данных по одной и той же линии между несколькими устройствами (многоточечный режим), то время использования этой линии для передачи данных должно быть поделено между всеми устройствами, ведущими передачу. Для сохранения той же средней скорости передачи за цикл обмена данными между всеми устройствами, что и при работе в одноточечном режиме, скорость обмена данными при работе в многоточечном режиме должна быть повышена пропорционально

количеству устройств, подключенных к линии передачи. Это может быть достигнуто путем перехода от однопроводной несимметричной линии к двухпроводной симметричной линии связи.

Задача обеспечения многоточечного режима обмена данными и повышение скорости передачи решается с помощью интерфейса RS485, включаемого на выходе интерфейса RS232C через согласующее устройство (адаптер). Типичная двухпроводная информационная сеть, построенная на базе интерфейса RS485, приведена на рис. 2.3. Основой каждого интерфейсного узла RS485 является пара электронных преобразовательных устройств: G и R. Одно из них (G) преобразует несимметричное входное напряжение в симметричное линейное напряжение между точками A и B на его выходе. Устройство G является передатчиком сигналов TD, которые поступают на вход G с выхода интерфейса RS232C. Единичному логическому сигналу на выходе АВ соответствует отрицательное напряжение в точке А по отношению к точке В (U_{AB}), а нулевому сигналу – положительное напряжение U_{AB} . Таким образом, передача информации осуществляется разнополярными импульсами напряжения одной и той же амплитуды, что облегчает реализацию гальванической развязки. Устройство R является приемником, оно преобразует симметричный дифференциальный сигнал U_{AB} , поступающий с линии связи, в несимметричный выходной сигнал RD. В целом симметричная дифференциальная система связи, на которой построена информационная сеть рис. 2.3, реализуется с помощью трех проводов: связанных проводов А и В и общего провода SG (сигнальная земля). Такая трехпроводная система связи позволяет создать помехоустойчивую скоростную магистраль со скоростью передачи до 10 Мбит/с при длине линии связи до 15 м и не менее 90кбит/с при длине линии связи до 1200 м. На такой линии связи может быть подключено до 32 приемо-передающих узлов.

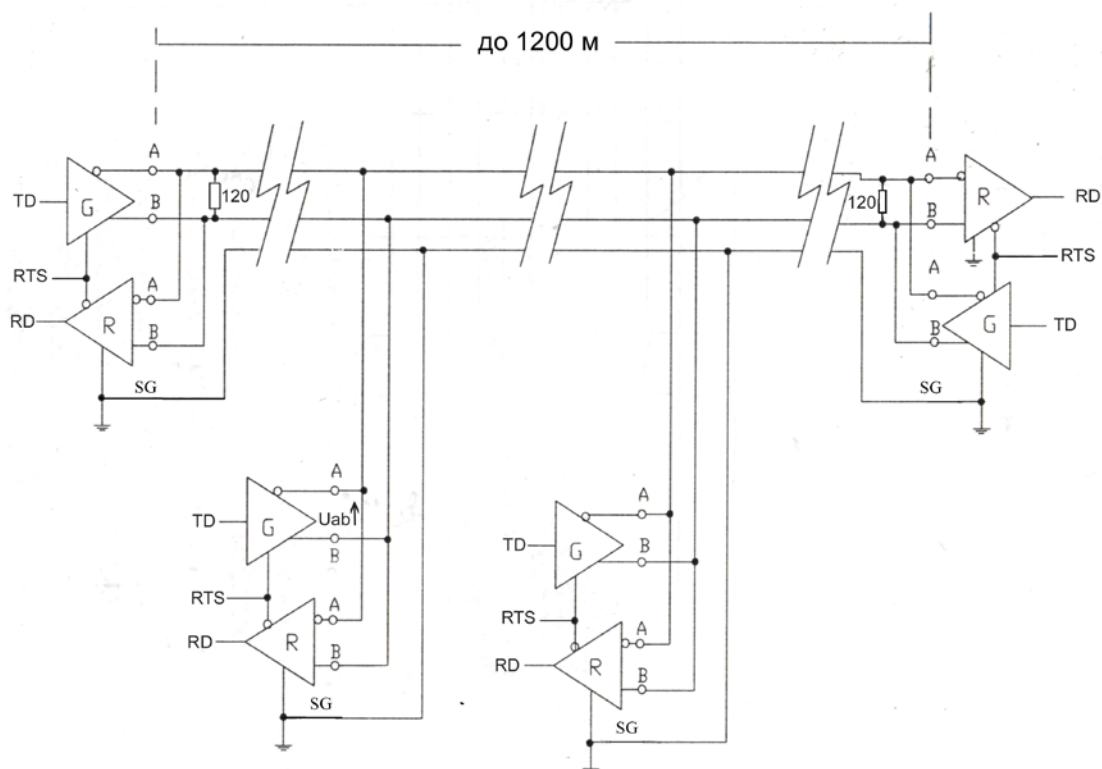


Рис.2.3. Типичная двухпроводная многоточечная информационная сеть на базе интерфейса RS485

У обоих концов линии связи должны быть подключены согласующие резисторы величиной 120 Ом. Подключение таких резисторов в промежуточных точках подсоединения адаптеров RS485 не требуется.

Все передатчики информационной сети рис. 2.3 передают информацию в одну и ту же информационную магистраль, так что когда один передатчик передает данные, остальные должны быть отключены. Чтобы это стало возможным, все передатчики G (как и приемные устройства R) реализованы в виде схем с тремя возможными состояниями выходов: логическая единица, логический ноль и отключенное состояние. Каждое из устройств G включено при сигнале запроса передачи $RTS=1$, а работающее с ним в паре в составе интерфейса RS485 приемное устройство R при этом отключено (см. рис.2.3). При $RTS=0$ передатчик G интерфейса RS485 оказывается отключенным, а приемник R соответственно включенным. Поскольку в двухпроводной информационной системе, построенной на интерфейсе RS485, включенным может быть только один передатчик, все остальные узлы информационной сети будут работать в режиме приема. Те из них, кому адресованы передаваемые данные, будут их принимать через приемные устройства R. Последние преобразуют данные, пришедшие в виде симметричных импульсов напряжения U_{AB} , в несимметричные сигналы RD, которые через адаптер попадают на интерфейс RS 232C или в приемный

регистр исполнительного устройства, управляемого от УВМ. Любое исполнительное устройство может передавать через передатчик G запрошенные данные в УВМ. Возможен вариант четырехпроводной сети, но она может работать только в режиме ведущий(master) – ведомый (slave). При этом все приемники ведомых узлов подсоединяются только к передатчику ведущего узла, а все передатчики ведомых узлов подсоединены только к приемнику ведущего узла. Такая схема повышает надежность информационной сети, построенной по принципу ведущий – ведомый, но и удорожает её.

Локальные сети АСУТП обычно построены по схеме ведущий – ведомый и употребляют интерфейс RS485. В качестве примера рассмотрим двухпроводную информационную сеть, состоящую из ведущей УВМ и ведомых устройств. УВМ через адаптер RS232/RS485 связана с ведомыми оконечными устройствами, количество которых превышает 32 (рис.2.4). В этом случае увеличение количества подключаемых оконечных устройств типа УСО и интеллектуальных датчиков достигается за счет использования повторителя. Повторитель (repeater) системы RS485 является двухпроводным двухсторонним устройством, которое передает данные, генерируемые в информационной сети, в обе стороны с одновременным восстановлением исходной амплитуды импульсов передаваемых данных. С его вторичной стороны может быть подключено еще 31 оконечное устройство.

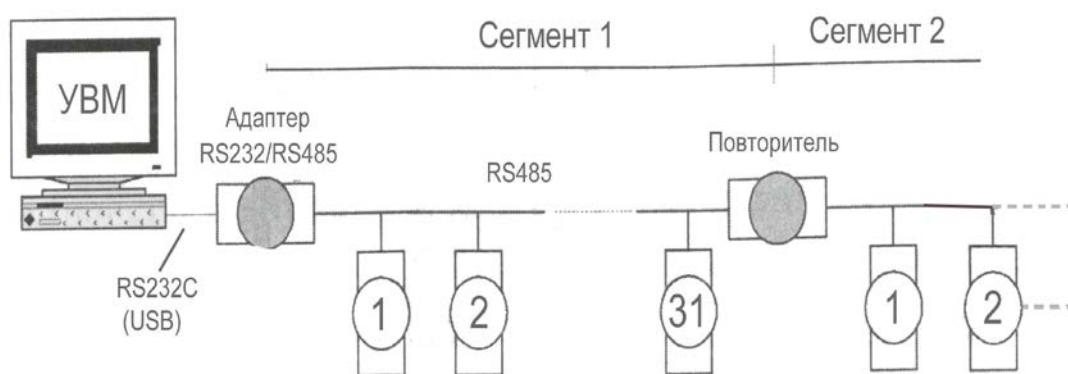


Рис. 2.4. Информационная сеть RS485 по схеме ведущий-ведомый

Локальная сеть, построенная на базе интерфейса RS485, аналогично приведенной на рис.2.4, может быть присоединена к УВМ и через интерфейс USB, но с адаптером USB/RS485. Интерфейс USB отличается повышенными скоростями передачи (до 12Мбит/с при длине связного сегмента до 5 м), а также малогабаритными надежными разъемами.

2.5.3. Защита информации от искажений

Дополнительное усложнение устройств обмена информацией может быть вызвано необходимостью защиты от помех. Основными источниками помех являются дефекты системы синхронизации УВМ с оконечными устройствами, электромагнитные поля, создаваемые работающим электрооборудованием, в том числе самой УВМ, и фоновое радиоактивное излучение. Помехи вызывают искажение хранимой и передаваемой информации: вместо единицы в отдельных битах может оказаться записанным ноль и наоборот, что ведет к сбоям в работе УВМ. Наиболее распространенным способом борьбы с искажениями, возникающими при передаче информации, является *контроль четности*. При контроле четности исходят из того, что все правильно запрограммированные слова информации должны содержать четное число единиц. Так, при программировании устройств ЧПУ в коде ISO-7, для достижения четности в старший бит записывается 1, если исходная кодовая комбинация данного байта содержит нечетное количество единиц. В противном случае в старший бит записывается 0. Если затем при проверке на четность окажется, что проверяемый байт содержит нечетное число единиц, то это будет означать, что при записи или пересылке данного байта произошел сбой. Нетрудно заметить, что двойной сбой в проверяемом слове информации не может быть обнаружен посредством контроля четности, но вероятность двойного сбоя много меньше вероятности единичного сбоя.

Если при передаче слова информации произошел сбой, то передача данного слова должна быть произведена повторно. Автоматическая повторная передача имеет смысл, если есть уверенность, что передаваемая информация не была искажена еще до передачи. Такое искажение весьма вероятно при длительном хранении информации (например, управляющих программ) в электронных ЗУ локальных УВМ, таких как программируемые контроллеры. Полезно перед пересылкой длительно хранимой информации по каналу связи проверить ее правильность и исправить обнаруженные ошибки.

Защита информации от искажений производится путем использования при ее кодировании кодов с обнаружением и исправлением ошибок. Построение таких кодов базируется на том, что часть кодовых комбинаций слова информации употребляется не для записи информации, а для борьбы со случайно возникающими ошибками, ведущими к сбоям в работе АСУТП. Возможности кода с обнаружением или исправлением ошибок определяются *кодовым расстоянием d* , характерным для данного кода. Кодовое расстояние – это минимально возможное число несовпадающих разрядов в

рассматриваемом блоке (наборе) слов информации. Если оно равно единице, то обнаружение ошибок в записи информации невозможно. О возможности обнаружения и исправления ошибок можно судить по следующему соотношению:

$$d=r+s+1, \quad (2.15)$$

где r -число обнаруживаемых ошибок;

s -число исправляемых ошибок из общего числа r обнаруженных ошибок, т.е. обязательно соблюдается соотношение $r \geq s$.

Нетрудно подсчитать, что при кодировании с обеспечением контроля четности в коде ISO-7 имеет место $d=2$. Таким образом, контроль данного типа обеспечивает обнаружение одиночной ошибки, но не исправление ее. В качестве примера кода, обеспечивающего также и исправление одиночных ошибок, рассмотрим один из $(n-1, k)$ -кодов Хемминга (R. W. Hamming), пригодный для исправления ошибок, возникающих при хранении и пересылке программ ЧПУ, записанных в коде ISO-7.

Предположим, что в рассматриваемом устройстве ЧПУ информация обрабатывается словами $n=16$ бит (2 байт). Для кодирования в коде Хемминга все биты слова нумеруются слева направо (см. рис.2.5), причем биты, номера которых равны степени числа 2 (в данном случае биты 1,2,4 и 8) играют роль битов четности, а все оставшиеся биты, кроме нулевого, используются как информационные.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
*	0	1	0	0	0	0	1	1	0	1	1	1	0	0	0

Рис. 2.5. Код Хемминга (15,11) с записью цифры 8 в коде ISO-7

Таким образом, данный код Хемминга состоит из 15 битов и контролирует блок информации величиной в 11 бит, т.е. является (15,11) – кодом Хемминга.

Каждый из битов четности контролирует в данном коде Хемминга свою особую группу из 8 битов, включая самого себя. Их номера указаны в таблице 2.3.

Таблица 2.3

Биты четности	контролируемые биты							
1	1	3	5	7	9	11	13	15
2	2	3	6	7	10	11	14	15
4	4	5	6	7	12	13	14	15
8	8	9	10	11	12	13	14	15

Значение бита четности устанавливается таким, чтобы сумма единиц, входящих в его группу, была четной. Это можно проследить по рис. 2.5. Так, в группу бита четности 8 входят три единичных бита: 10, 11 и 12, что обусловило единичное значение самого бита 8.

Запись байта кода ISO-7 в слово кода Хемминга производится обычным порядком: номера разрядов возрастают справа налево, символы кода ISO-7 соответствуют коду ASCII (см.табл. П1.1), старший бит кода ISO-7 занимает место бита 7. Значения информационных битов с хемминговыми номерами меньше 7 при записи кода ISO-7 могут быть выбраны произвольно и для определенности сделаны равными нулю. В составе слова кода Хемминга бит четности кода ISO-7 не играет роли бита четности, является обычным информационным битом. При записи программы ЧПУ в электронную память ЛУВМ каждый байт программы, написанной в коде ISO-7, программно преобразуется в слово кода Хемминга, а при вызове программы из ОЗУ ЛУВМ для пересылки по каналам связи или на исполнение, производится обратное преобразование в код ISO-7 с исправлением возможных ошибок. Исправление кодом Хемминга сбоев ОЗУ производится в следующем порядке:

- производится проверка всех битов четности слова;
- если их значения (0 или 1) правильны, то сбоя не было;
- вычисляется сумма номеров всех неправильных битов четности, если они есть;
- инвертируется значение бита, номер которого равен сумме номеров неправильных битов четности;
- исправленное слово кода Хемминга записывается в ОЗУ вместо дефектного слова.

Допустим, что в слове, представленном на рис. 2.5, в результате сбоя в бите 12 оказался 0 вместо 1. Тогда суммы битов, контролируемых битами четности 4 и 8 (см. табл. 2.3), окажутся нечетными. Следовательно, сбой мог произойти только в одном из битов, контролируемых битами 4 и 8 совместно. Это биты 12, 13, 14, и 15. но биты 13, 14 и 15 контролируются также битами 1 и 2, контрольные суммы которых четны, т.е. правильны. Тогда неправильным может быть только бит 12, номер которого определяется суммой номеров битов 4 и 8. Тем самым подтверждается вышеизложенное правило исправления дефектных битов.

Для организации обмена информацией с ВУ в составе пульта управления УВМ (например, с клавиатурой или накопителями на магнитных дисках) используются специализированные программируемые микроконтроллеры, являющимися, по сути дела, специализированными микро-ЭВМ, которые наряду с

микроконтроллерами в составе систем управления исполнительными электроприводами превращают УВМ в ведущий узел локальной информационной сети.

2.6. Организация обмена информацией в АСУТП

2.6.1. Информационная структура АСУТП

Информационная структура АСУТП с учетом взаимодействия с оператором и АСУП приведена на рис.2.6 . На верхнем уровне (уровень 1) находятся автоматизированные рабочие места (АРМ) технолога, ведущего технологическую подготовку производства по системе АСТПП в рамках ГПС, и оператора АСУТП. Основой аппаратной части АРМ является промышленный компьютер ПК, который играет роль ЦУВМ. На втором уровне управления в качестве ЛУВМ, управляющих локальными ТО, выступают программируемые логические контроллеры ПЛК и устройства ЧПУ (УЧПУ). Контроллеры ПЛК применяются, когда главным назначением ЛУВМ является управление технологическим циклом, которое состоит, в основном, во включении и отключении различных технологических агрегатов. Операции включения и отключения, а также прием сигналов обратной связи от технологического оборудования производятся с помощью специализированных модулей УСО. От ПЛК могут быть также реализованы несложные операции по регулированию автоматизированных электроприводов (АЭП). Более сложные функции ЧПУ реализуются УЧПУ, которые помимо управления технологическим циклом обеспечивают управление многими следящими АЭП. Системы управления АЭП и другими исполнительными устройствами, а также автоматизированные системы сбора первичной технологической информации составляют третий, самый нижний уровень управления в АСУТП.

Устройства получения, пересылки и хранения информации на каждом уровне АСУТП образуют информационную сеть. Структура такой сети чаще всего имеет вид совокупности магистралей (линий), к которым подсоединяются отдельные источники, преобразователи и приемники информации. Промышленные компьютеры ПК верхнего уровня АСУТП объединяются в промышленную информационную сеть, как показано на рис. 2.6, с помощью сетевой технологии Ethernet (произносится *этернет*). Сеть Ethernet объединяет ПК в рамках АСУП. На контроллерном уровне (уровень 2) также применяется технология Ethernet, но здесь с ней конкурируют локальные сети Profibus, Canbus и др.

Ethernet

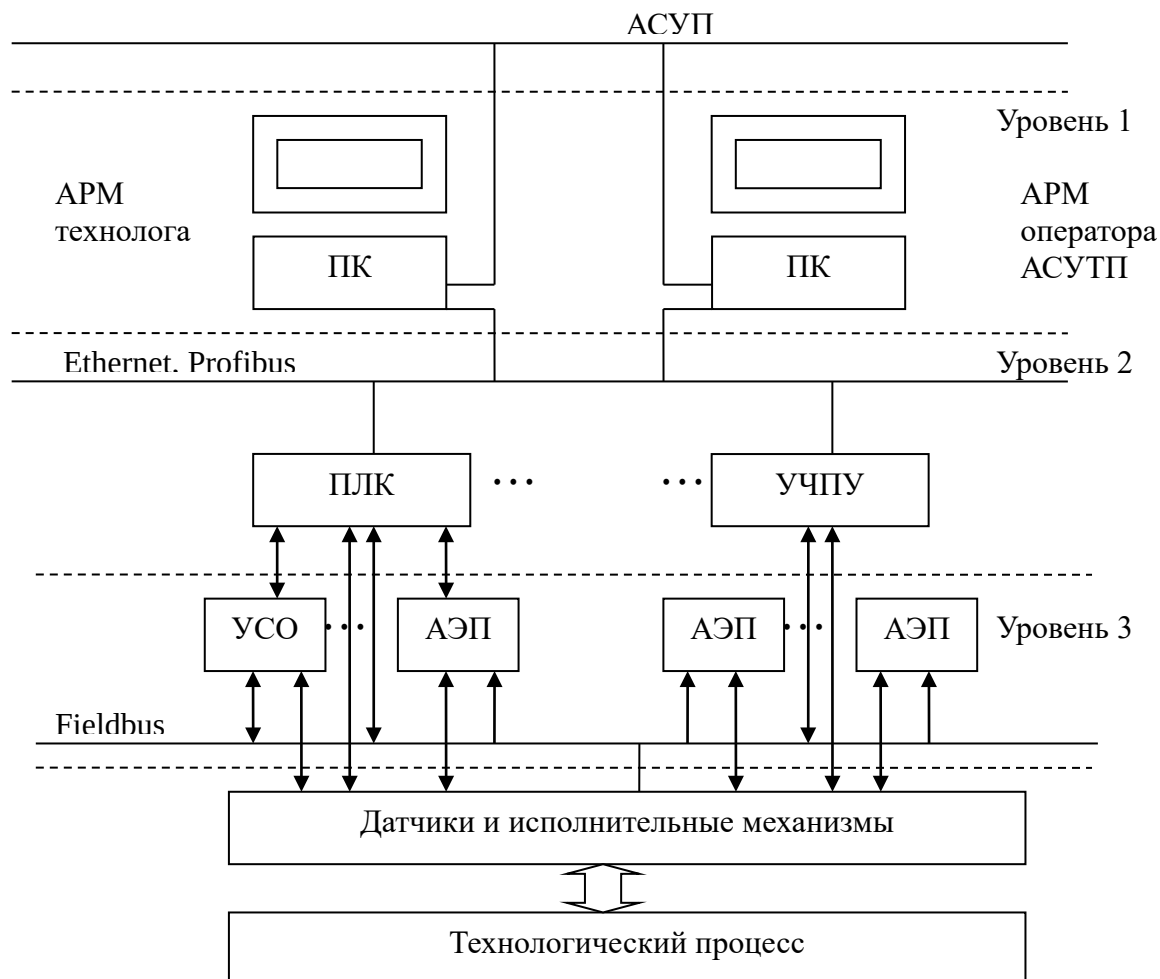


Рис. 2.6. Информационная структура АСУТП

На нижнем уровне АСУТП функции объединения датчиков, систем управления исполнительными механизмами и контроллеров в единую цифровую сеть осуществляется системой Fieldbus (полевая сеть). Она заменяет аналоговые интерфейсы 4-20 мА и 0-10 В цифровой коммуникационной технологией. Благодаря этому большое количество отдельных линий связи, идущих от датчиков и исполнительных устройств к модулям ввода-вывода ПЛК и УЧПУ, удастся заменить на один информационный малопроводный кабель, по которому также поступает электропитание датчиков и маломощных исполнительных устройств. Системы управления нижнего уровня становятся «интеллектуальными», могут выполнять функции самонастройки и самодиагностики, так что снижается информационная нагрузка на управляющие устройства верхних уровней. Наиболее распространены варианты системы Fieldbus на базе стандартизированных сетевых технологий Profibus фирмы SIEMENS (ФРГ) и Modbus фирмы MODICON (Франция).

2.6.2. Информационные сети Ethernet

Сетевая технология Ethernet (Этернет) – распространенный метод построения локальных информационных, в том числе

промышленных сетей. Сети Ethernet строятся на базе американского стандарта IEEE 802.3 и международного стандарта ISO 8802.3. В зависимости от типа физической среды стандарт предусматривает ряд модификаций, из которых наибольший практический интерес представляют:

10 Base-T¹ -- для двух неэкранированных витых пар (UTP—Unshielded Twisted Pair) категории 3 (телефонный кабель);

10 Base-F² – для многомодового оптоволоконного кабеля, использующего два оптоволоконка;

100 Base-TX – для двухпарного кабеля из двух неэкранированных витых пар UTP категории 5;

100 Base-FX – для многомодового оптоволоконного кабеля, использующего два оптоволоконка.

Первые две модификации соответствуют пропускной способности сети $C=10\text{Мбит/с}$ (обычный Ethernet), а две последние – пропускной способности $C=100\text{Мбит/с}$ (Fast Ethernet, *быстродействующий* Ethernet). Все отличия Fast Ethernet от обычного Ethernet получены за счет изменений на физическом уровне, а на канальном уровне их характеристики одинаковы. Так, метод доступа к среде передачи данных (к линиям передачи) один и тот же. Имеется в виду метод коллективного доступа с наблюдением за несущей и с обнаружением коллизий CSMA/CD.

Согласно этому методу (см. §2.5) два узла могут случайно начать одновременную передачу данных, если сигналы узла, начавшего передачу первым, не дойдут до узла, готового начать передачу вторым, раньше, чем этот второй тоже начнет передачу. Такая ситуация может случиться, прежде всего, с узлами, максимально удаленными друг от друга. Если второй узел начнет передачу как раз в тот момент, когда до него достигнет передающий сигнал первого узла, то последний получит информацию об этом событии только после того, как до него дойдет передающий сигнал второго узла. Таким образом, максимальное время, которое должно быть затрачено на обнаружение наступившей коллизии, есть *время двойного оборота* сигнала между максимально удаленными друг от друга узлами информационной сети. Это время не должно превышать величины промежутка времени, необходимого для передачи самого короткого кадра, чтобы первый передающий узел и все принимающие передаваемую информацию узлы успели забраковать передаваемый кадр, поскольку он испорчен сигналами второго передающего узла. Минимальная длина поля данных кадра в сети Ethernet составляет 46 байт, а с учетом 18 байт служебной информации (см. табл.2.4) и 8 байт преамбулы полная длина кадра Ethernet составляет минимум 72

¹ T=Twisted (скрученный)

² F=Fiber (волокно)

байт или 576 бит. По этой причине время двойного оборота не должно превышать 575 битовых интервалов, что при 10-мегабитном стандарте Ethernet (1 бит за 0,1 мкс) составит 57,5 мкс. В технологии Fast Ethernet учет количества битов служебной информации производится без преамбулы, так что величина времени прохождения кадра минимальной длины оценивается в 512 битовых интервалов, т.е. в 5,12 мкс. Исходя из полученных оценок допустимого времени двойного оборота сигналов, определяется максимальное допустимое расстояние между узлами (станциями) различных модификаций сети Ethernet.

Типовую структуру кадра информации, передаваемой по сети Ethernet, можно проследить по табл.2.4.

Таблица 2.4

DA	SA	L	Поле данных, байт	FCS
6	6	2	46—1500	4

Основное содержание кадра Ethernet составляют:

DA – *адрес назначения* (Destination Address), т.е. адрес узла, в который пересылается информация, помещенная в поле данных. Адрес назначения представляет собой уникальный, ни в одной сети типа Ethernet не повторяющийся, локальный шестибайтовый адрес, называемый MAC³-адресом. Он обычно записывается в виде шести пар 16 – ричных цифр, разделенных дефисами, например: 40-A0-17-B5-DC-31.

SA – *адрес источника* (Source Address) информации, также являющийся шестибайтовым MAC-адресом.

L – *длина* (Length) поля данных в кадре, записывается двумя байтами. Знание длины кадра позволяет сократить время его передачи.

FCS – *поле контрольной суммы* (Frame Check Sequence), занимающее последние 4 байта кадра. Оно содержит подсчитанную перед пересылкой кадра контрольную сумму. После получения кадра по заданному адресу контрольная сумма вычисляется заново в узле назначения. Если присланное и вновь вычисленное значение контрольной суммы совпадают, то полученный кадр считается неискаженным.

Поле данных (Data) содержит пересылаемую по указанному адресу DA информацию, количество которой может составлять от 0 до 1500 байт. В табл.2.4 предполагается, как и положено для сети Ethernet, что минимальный объем пересылаемой информации составляет 46 байт. Это необходимо, чтобы сигнал коллизии успел

³ MAC=Media Access Control – управление доступом к среде

дойти от самого дальнего узла сети до передающего узла ранее окончания передачи текущего кадра. Если фактический объем информации, которую необходимо передать, не достигает 46 байт, то используется дополнительное поле, *поле заполнения* (Padding), не показанное в табл.2.4. Поле заполнения дополняет поле данных до минимально допустимого значения в 46 байт пустыми символами. Каждый кадр, передающийся по сети Ethernet, начинается с двух следующих полей, не показанных в табл.2.4:

P – *поле преамбулы* (Preamble), которое состоит из 7 синхронизирующих байтов 10101010.

SFD – определитель начала кадра (Start of Frame Delimiter). Он состоит из одного байта 10101011, появление которого завершает преамбулу и указывает на то, что следующий байт – это начало адреса назначения DA.

По окончании передачи кадра все узлы сети Ethernet обязаны выдержать паузу величиной в 9,6мкс (соответственно – 0,96мкс в сети Fast Ethernet), называемую межкадровым интервалом (Inter Packet Gap). После паузы любой узел имеет право сделать попытку начать передачу.

2.6.3. Структура физической среды Ethernet

Информационная сеть 10Base-T, использует в качестве среды две неэкранированные витые пары UTP телефонного кабеля. Сеть имеет звездообразную структуру благодаря тому, что узлы сети соединяются друг с другом через посредство специального устройства, называемого *концентратором* или *хабом* (hub). Концентратор, являясь многопортовым повторителем, усиливает полученный им сигнал и распределяет его по всем узлам, подключенным к концентратору, кроме узла, на который данный сигнал поступил. Если сигналы поступают сразу на два входа концентратора, то он фиксирует коллизию и формирует усиленный сигнал глушения передачи, состоящий из 32 бит (jam-последовательность). Благодаря подключению узлов сети через концентратор (см. рис.2.7), образуется единая среда передачи данных, эквивалентная общей шине (логическая общая шина). Как видно на рис.2.7, передатчик Tx каждого сетевого адаптера соединяется с приемником Rx соответствующего входа концентратора, а передатчик каждого выхода концентратора соединяется с входом приемника соответствующего сетевого адаптера. При скорости передачи 10 Мбит/с длина кабеля на витой паре между сетевым адаптером узла сети и концентратором не может превышать 100 м, что определяется полосой пропускания и затухания сигнала в витой паре.

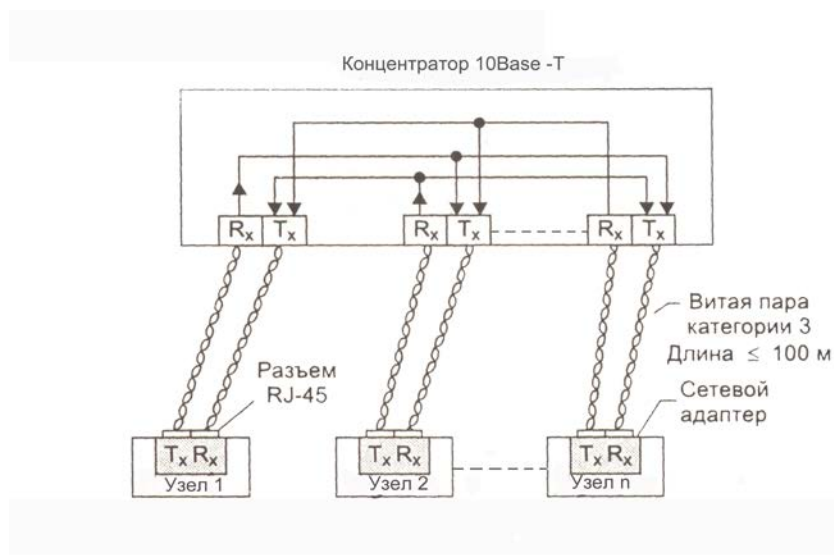


Рис. 2.7. Структура сети 10Base-T: T_x – передатчик; R_x – приемник

Концентраторы можно соединять друг с другом тем же способом, что и с сетевыми адаптерами. Надежное распознавание возникающих в сети коллизий при минимальной длине пересылаемого кадра в 72 байт и уверенная синхронизация обменивающихся информацией узлов сети обеспечиваются при максимальном числе концентраторов между двумя любыми узлами не более четырех (правило четырех хабов). Общее число узлов в сети 10Base-T не должно превышать заданного стандартом предела в 1024 узла. Чтобы создать информационную сеть, насчитывающую сотни узлов, достаточно использовать двухуровневую иерархическую структуру, в которой корневой концентратор с несколькими десятками портов T_x – R_x обменивается информацией через каждый свой порт с ведомыми концентраторами, каждый из которых, в свою очередь, обеспечивает обмен информацией с несколькими десятками конечных узлов.

Максимальный диаметр сети 10Base-T при условии установки четырех хабов может, очевидно, достичь 500 м: по 100 м между хабами и еще по 100 м с каждого конца сети. Если требуется сеть большей длины, необходимо переходить на оптоволоконный стандарт 10Base-F, но тогда информационная сеть обойдется дороже.

Применение концентраторов приводит к разделению общего кабеля сети на отдельные кабельные отрезки, каждый из которых связан одним конечным узлом и подключен к центральному коммуникационному устройству. Наличие концентратора позволяет контролировать отдельные отрезки сети, отключая их в случае неисправности кабеля или сетевого адаптера. При этом сохраняется работоспособность остальной части сети.

Сигналы, передаваемые по линии связи, должны обеспечивать самосинхронизацию и отсутствие постоянной составляющей.

Самосинхронизация состоит в определении приемником момента считывания данных по фронту приходящего информационного сигнала, а отсутствие постоянной составляющей необходимо для обеспечения возможности реализации трансформаторной гальванической развязки между приемником и передатчиком. Стандарт 10Base-T, как и другие сети Ethernet, обеспечивает выполнение указанных требований путем применения при передаче информации *манчестерского кода*. При манчестерском кодировании каждый такт, соответствующий передаче одного бита информации, организуется так, что значение бита, определяется перепадом потенциала *в середине такта*. Единица кодируется перепадом от отрицательного уровня к положительному, а ноль – обратным перепадом. При необходимости в начале такта производится служебный перепад потенциала, который не является информационным, но подготавливает перепад нужного характера в середине такта. Если C —битовая скорость передачи данных, то основная гармоника передаваемых сигналов имеет частоту C Гц при передаче последовательности либо одних нулей, либо одних единиц, но её частота равна $C/2$ Гц при передаче перемежающихся нулей и единиц.

Оптоволоконная сеть Ethernet, построенная в соответствии со стандартом 10Base-F, имеет такую же структуру, что и сеть стандарта 10Base-T, показанная на рис.2.7. Только вместо витой пары здесь применяется многомодовое оптическое волокно с полосой пропускания не менее 500 МГц. Благодаря использованию оптоволокна максимальное расстояния между узлом и концентратором увеличивается до 2000 м при общей длине сети до 2500 м. Из-за повышенной стоимости оптоволоконного кабеля и связанного оборудования оптоволоконные линии употребляют по преимуществу для связи между концентраторами, причем правило четырех хабов сохраняется.

Сети Fast Ethernet применяются, если необходимо многократно увеличить пропускную способность. Структура сетей Fast Ethernet такая же, как и у сетей 10Base-T/10Base-F: древовидная, построенная на концентраторах (см. рис.2.7). Основным отличием является сокращение диаметра сети до 200 м для сетей 100Base-TX и примерно до 300 м для сетей 100Base-FX, что объясняется уменьшением времени передачи кадра минимальной длины в 10 раз за счет увеличения скорости передачи в 10 раз. Соответственно уменьшается запас времени на ликвидацию последствий коллизий. Увеличение диаметра сетей за указанные пределы возможно за счет применения *коммутаторов*. Коммутаторы выполняют те же функции, что и концентраторы, но одновременно они разделяют сети Ethernet на автономные сегменты, так что коллизии, возникающие этих

сегментах, через коммутатор не передаются. Ограничение коллизий пределами отдельного сегмента позволяет наращивать длину сети путем последовательного соединения сегментов.

2.6.4. Контроллерные и полевые сети

На контроллерном уровне (уровень 2, см. рис.2.6), помимо сетевой технологии Ethernet используются и многие другие типовые решения. Из них отметим сеть Profibus (PROcess Field BUS), разработанную немецкими компаниями SIEMENS, BOSCH и KLOCKNER—MÖLLER. Собственно, на контроллерном уровне работает сеть Profibus-FMS (Field Bus Message Specification), но в комплекте оборудования для систем автоматизации имеется также вариант сети Profibus-DP (Distributed Periphery), обслуживающей цепи низовой автоматики, исполнительные и измерительные устройства. Сеть Profibus использует стандарты физического и канального уровней OSI-модели, а также обеспечивает интерфейс между канальным и прикладным уровнями. Управление связью на контроллерном уровне обеспечивается передачей маркера по контроллерному кольцу. Контроллеры ПЛК в составе информационного кольца являются ведущими (master) узлами для нижнего (полевого) уровня управления, состоящего из исполнительных и измерительных устройств (ИУ). Последние по отношению к ведущим (контроллерным) узлам являются ведомыми (slave). Сеть может состоять из 122 узлов, часть которых могут быть ведущими узлами. Сообщение-маркер передается от одного ведущего узла к другому. Каждый контроллер, получивший маркер, имеет право на доступ к общей шине Profibus на строго регламентированное время. В это время он обменивается информацией со своими ведомыми узлами и с ЦУВМ. Протокол Profibus предусматривает следующие команды обмена информацией:

SDN (послать данные без подтверждения);

SDA (послать данные с подтверждением);

SRD (послать и запросить данные);

CSRD (циклическая посылка и запрос данных).

Обнаружение и исправление ошибок в процессе передачи данных производится с помощью кодов Хемминга с кодовым расстоянием 4, т.е. в пересылаемом кадре будут обнаружены 2 ошибочных бита и один из них может быть восстановлен.

На нижнем уровне управления для высокоскоростного обмена информацией с ИУ применяется сеть Profibus-DP. В ней интерфейс физического уровня базируется на стандарте RS485. Максимальная скорость обмена информацией достигает 12 Мбит/с при длине связного кабеля не более 100 м, а при длине в 1200 м она

уменьшается до 100 кбит/с. Заметим, что DP-протокол является нижним уровнем протокола Profibus-FMS.

Наиболее распространенным протоколом нижнего (полевого) уровня управления, обеспечивающего управление исполнительными устройствами и сбором технологической информации, является Modbus. Этот протокол разработан фирмой MODICON (Франция). Физический интерфейс сети Modbus чаще всего базируется на стандартах RS232C, RS422 и RS485. Сеть строится по принципу ведущий – ведомый (master – slave). Предполагается, что один ведущий узел может обеспечить обмен информацией с группой до 247 ведомых узлов. Ведущий узел инициирует обмен информацией двух типов:

- 1) запрос – ответ, когда ведущий узел ведет обмен информацией только с одним ведомым узлом;
- 2) широковещательная передача (broadcast), когда ведущий узел обращается ко всем ведомым узлам сети одновременно, путем выставления адреса 0.

Для кодирования передаваемых данных используется код ASCII. При обмене информацией запрос со стороны ведущего узла включает в себя код команды, адрес ведомого узла, к которому обращен запрос, поле данных фиксированного размера, сами данные и контрольная кодовая комбинация (CRC-код). Команды, предусмотренные протоколом Modbus, позволяют реализовать следующие функции:

- чтение или запись битов;
- чтение или запись регистров;
- функции диагностики;
- программные и другие функции.

Данный протокол отличается простотой логики и независимостью от типа интерфейса.

2.6.5. Диспетчеризация в рамках АСУТП

Промышленные компьютеры верхнего уровня АСУТП обеспечивают управление техпроцессом в режиме диспетчеризации. Диспетчерское управление является основным и наиболее перспективным методом автоматизированного управления сложными динамическими системами, каковыми являются системы АЭП в современных технологических комплексах. Именно на принципах диспетчерского управления строятся крупнейшие АСУТП в промышленности, энергетике, на транспорте и в других областях их применения.

Проблема диспетчерского управления стала особенно актуальной за последние 20 лет, когда резко возросло количество находящихся в

эксплуатации высокоэффективных и высоконадежных систем автоматизации. Их создание не было бы возможным без организации высокоэффективного обмена информацией между человеком-оператором и автоматизированным комплексом путем построения соответствующего человеко-машинного интерфейса. Таким интерфейсом явилась система SCADA (Supervisory Control and Automated Data Acquisition) – автоматизированная система сбора данных и оперативного диспетчерского управления.

Система SCADA предусматривает инструментальное и оперативное программное обеспечение (ПО). Ее инструментальный комплекс обеспечивает разработку конкретного ПО, обслуживающего АРМ технолога и оператора (диспетчера). Разработанные оперативные программы реализуются на рабочих местах в определенной операционной среде.

Система SCADA организует процесс сбора информации в режиме *реального времени*, т.е. практически одновременно с событиями контролируемого техпроцесса. Обработанная информация поступает к оператору (диспетчеру), контролирующему ход процесса, для анализа и принятия, при необходимости, управленческих решений. Оператор несет, как правило, ответственность, за управление автоматизированным процессом, который при нормальных условиях требует лишь некоторой подстройки параметров для обеспечения оптимальности контролируемого процесса. Поэтому активное участие оператора в процессе управления требуется нечасто и в непредсказуемые моменты времени, обычно – для устранения неполадок и ликвидации последствий нештатных ситуаций. Основные функции оператора таковы:

- планирование ближайших действий в течение рабочей смены;
- обучение (программирование) системы управления на запланированные действия;
- отслеживание автоматического функционирования управляемого ТО;
- вмешательство в течение техпроцесса в случае наступления критических событий или при необходимости подстройки параметров.

В соответствии с перечисленными функциями оператора на SCADA-системы возлагаются следующие функции по созданию инструментального комплекса и обеспечению работы пульта управления в режиме реального времени:

сбор, первичная обработка и накопление информации о параметрах техпроцесса, полученной от контроллеров второго уровня управления и других цифровых устройств, связанных с пультом оператора непосредственно или через информационную сеть;

архивирование информации и ее дальнейшая вычислительная и логическая обработка;

отображение информации на дисплее операторской станции (АРМ) в виде различных мнемосхем, анимационных изображений, таблиц, графиков, выделение аварийных ситуаций и пр.;

ввод и передача команд оператора в управляющие устройства автоматизированного комплекса;

составление и ввод в действие прикладных управляющих программ, обеспечивающих функционирование АРМ;

обеспечение информационных связей с серверами, другими АРМ, с верхним уровнем управления (АСУП) через различные сетевые структуры;

печать отчетов и протоколов в заданные моменты времени, регистрация аварийных ситуаций.

SCADA-системы выпускаются под различными фирменными названиями: Trace Mode (AdAstra, Россия), Genesis (ICONICS, США), Bridge VIEW (National Instruments, США), WinCC (SIEMENS, Германия) и др. Их общими чертами является работа в операционной среде Windows и применение общих стандартов в процессе обмена информацией. Поэтому они не ограничивают выбора ЛУВМ (контроллеров, УЧПУ) нижнего уровня управления, предоставляют большой набор драйверов ввода – вывода и имеют хорошо развитые средства создания собственных программных модулей драйверов для новых устройств нижнего уровня. Для связи драйверов ввода-вывода применяют два стандартных механизма протокола: DDE (Dynamic Data Exchange) и OLE (Object Linking and Embedding) для PC, или OPC (Ole for Process Control) – система обмена информацией, ориентированная на обмен информацией в режиме реального времени и потому предпочтительная для АСУТП.

В качестве примера рассмотрим графический человеко-машинный интерфейс (кадр управления), созданный для управления насосно-перекачивающей станцией (НПС) в рамках SCADA-системы (рис.2.8). Непосредственное управление НПС обеспечивается контроллером второго уровня. Кадр управления, приведенный на рис.2.8., позволяет управлять НПС с пульта управления АРМ оператора.

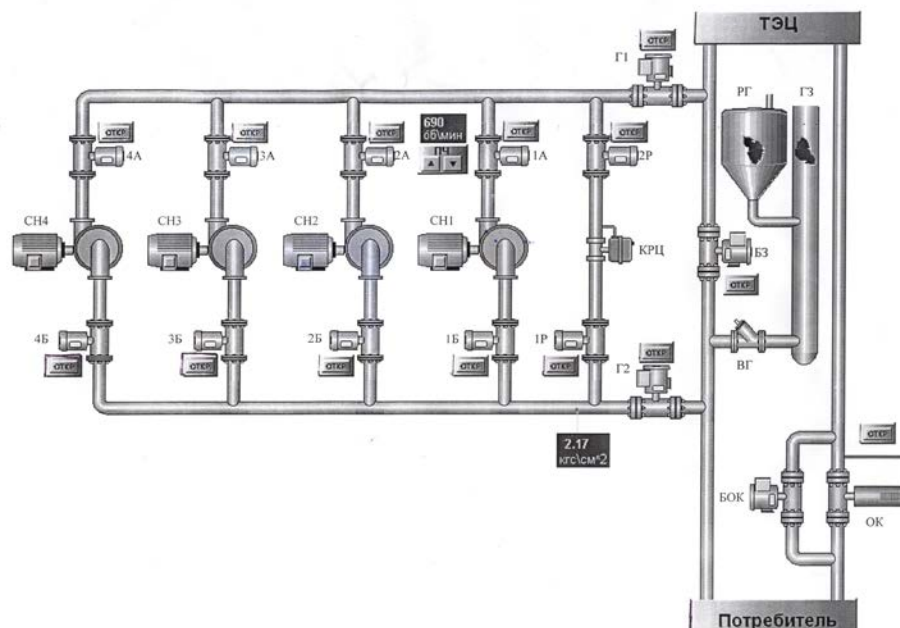


Рис. 2.8. Графический кадр управления насосно-перекачивающей станцией

Графический интерфейс состоит из следующих элементов:

СН1, СН2, СН3, СН4 – сетевые насосные агрегаты;

ПЧ – преобразователь частоты регулируемого электропривода агрегата СН1;

1А, 2А, 3А, 4А – задвижки с запорной арматурой на напоре НПС;

1Б, 2Б, 3Б, 4Б – задвижки с запорной арматурой на всасе НПС;

Г1, Г2 – головные задвижки с запорной арматурой;

1Р, 2Р – задвижки линии рециркуляции;

КРЦ – клапан рециркуляции;

БЗ – байпасная задвижка НПС;

ГЗ – гидрозатвор;

РГ – расширитель гидрозатвора;

ВГ – вентиль гидрозатвора;

ОК – отсечной клапан;

БОК – байпас отсечного клапана.

Для создания графического кадра использовалась библиотека готовых символов, включающая в себя графические изображения трубопроводов, запорной арматуры, а также элементов электрооборудования. При программировании графического кадра необходимые элементы библиотеки символов переносятся мышью компьютера и помещаются на предназначенное им место в рабочей области экрана, обеспечивая соединение с другими элементами необходимым образом. Для отображения цифровой и текстовой

информации формируются текстовые окна. Затем производится установка кнопок управления аналогично установке других графических элементов. К каждой кнопке привязывается соответствующая управляющая переменная.

Оператор управляет НПС, нажимая на кнопки кадра управления с помощью мыши. Для наглядности управления и привлечения внимания к происходящим событиям производится анимация кадра управления изменением цвета элементов и отображением движения жидкости по трубопроводам.

Контрольные вопросы к главе 2

1. Сформулируйте понятие информации на базе понятия сообщения.
2. Что такое энтропия и как ее вычислить?
3. Дайте оценку величины информации через убыль энтропии.
4. Что такое бит как единица информации?
5. Каковы основные единицы информации? Как с их помощью определяется емкость запоминающих устройств?
6. Что такое кодирование и в чем состоит процесс кодирования?
7. Что такое дискретный и аналоговый сигналы?
8. Каково должно быть соотношение между шагом квантования по уровню и шагом квантования по времени?
9. Что такое арифметический код с основанием n ? Как рассчитать числовое значение кодовой комбинации, записанной в арифметическом коде?
10. Что такое двоичный код? Какими элементами реализуются двоичные символы в технических устройствах?
11. Что такое двоичный арифметический код? Перечислите разновидности двоичного арифметического кода.
12. Что такое 16-ричный код (HEXCODE) и для чего он применяется?
13. Каковы преимущества двоичных систем кодирования?
14. Что такое двоично-десятичный код и для чего он применяется?
15. Что такое коды Грея и как они применяются в датчиках положения?
16. Что такое коды обмена информацией? Какова структура и принцип построения кода ASCII?
17. В чем состоит параллельная и последовательная передача слов информации по каналам связи?
18. Дайте определение интерфейса и протокола информационной сети.
19. Каковы основные типы сетевых топологий информационных сетей АСУТП и методы доступа к среде передачи данных?
20. Как обеспечивается многоточечный режим обмена данными с помощью интерфейса RS485?
21. Приведите пример информационной сети УВМ, устроенной по схеме ведущий-ведомый.

22. Как и для чего организуется контроль четности при передаче информации?
23. Как организуется исправление ошибок при передаче информации с помощью кода Хемминга?
24. Как сказывается на обмене информацией удаленность ЦУВМ от ЛУВМ в АСУТП?
25. Какова типичная информационная структура АСУТП? Приведите ее блок-схему.
26. Как организуется информационная сеть Ethernet для обмена информацией между ЦУВМ и ЛУВМ?
27. Какова структура и содержание кадра информации в сети Ethernet.
28. Как строится физическая среда Ethernet с помощью концентраторов?
29. Как организуется информационная сеть Modbus для обмена информацией между ЛУВМ и исполнительными устройствами?
30. Каковы основные функции оператора при диспетчерском управлении в АСУТП?
31. Какова структура и основные функции диспетчерской системы SCADA в составе АСУТП?
32. Какова структура и принцип построения человеко-машинного интерфейса на примере графического кадра управления НПС?

Глава 3. Моделирование технологических объектов, управляемых АСУТП

3.1. Алгоритмы функционирования

Что бы решать задачи управления технологическим объектом (ТО), следует достаточно точно и полно знать его свойства и характеристики, обеспечивающие нормальное течение технологического процесса. Характеристики объекта, обеспечивающие управление им, будем называть *управляющими* (входными) параметрами, а интересующие нас параметры техпроцесса будем называть *выходными* параметрами. Основной задачей управления будем считать изменение управляющих параметров по закону, обеспечивающему работу технологического объекта в соответствии заданным критерием оптимальности. Для решения этой задачи необходимо знать зависимость между управляющими и выходными параметрами техпроцесса в условиях действия различных возмущений. Достаточно точное и математически обоснованное описание зависимости между управляющими (входными) и управляемыми (выходными) параметрами ТО называется *алгоритмом функционирования*. Если алгоритм функционирования реализован в виде компьютерной программы или на моделирующей установке иного типа, то его называют *моделью технологического объекта*. Алгоритм функционирования, оформленный в виде совокупности математических выражений, называют *математической моделью*. При построении модели ТО алгоритм функционирования исследуемого ТО обычно расчленяют на отдельные математически однородные элементы, называемые типовыми звеньями. Этот процесс носит название *декомпозиции алгоритма*.

После этого модель ТО строится в виде совокупности типовых звеньев, объединенных характерными для данного ТО связями. Далее проводится исследование функционирования ТО путем подачи на модель типовых для данного ТО возмущений. Пригодность полученной модели для описания исследуемого ТО может быть оценена путем вычисления величины I суммарного квадратичного отклонения данных расчета на модели Y_p от результатов экспериментального исследования ТО Y_o :

$$I = \sum_{j=1}^n (Y_{jo} - Y_{jp})^2. \quad (3.1)$$

Чем меньше величина I , тем удачнее модель.

3.2. Аналитические методы моделирования

Многообразие технологических процессов обусловило многообразие моделей ТО и методов их получения. Последние подразделяются на *аналитические* и *экспериментальные*. Аналитические методы базируются на знании физических, химических и прочих законов, определяющих функционирование технологического объекта. Если теоретических знаний недостаточно, то применяются экспериментальные методы, когда параметры моделей определяются опытным путем. При составлении алгоритма функционирования аналитическим путем технологический объект представляют в виде совокупности типовых звеньев, в каждом из которых происходит однократное преобразование энергии. Звенья такого рода принято называть динамическими звеньями. Они описываются дифференциальными уравнениями не выше второго порядка.

Математическое описание звеньев данного типа производится в следующем порядке.

1) Выходной параметр рассматриваемого звена назовем преобразуемым продуктом Q , а управляющий (входной) параметр – разностью потенциалов U , измеряемой количеством энергии, затрачиваемой на единицу преобразуемого продукта.

2) Описываемое звено представим в виде контура, в котором сумма разностей потенциалов должна быть равна нулю:

$$\sum_{j=1}^m U_j = 0. \quad (3.2)$$

3) Связи между отдельными звеньями, отображающими ТО, формируются исходя из условия равенства нулю суммы расхода продукта в любом разветвлении

$$\sum_{j=1}^n \frac{dQ_j}{dt} = 0. \quad (3.3)$$

В качестве примера рассмотрим уравнения характерных для систем электромеханического преобразования энергии электрического и механического звеньев.

1) Уравнение цепи якоря двигателя постоянного тока

$$u - e_{\text{я}} = L_{\text{я}} \frac{di_{\text{я}}}{dt} + R_{\text{я}} i_{\text{я}}, \quad i_{\text{я}} = \frac{dq}{dt},$$

где u – напряжение, приложенное к цепи якоря;

$e_{\text{я}}$ – ЭДС якоря, причем предполагается, что положительное направление ЭДС якоря противоположно положительному направлению напряжения;

$L_{\text{я}}$ и $R_{\text{я}}$ – индуктивность и активное сопротивление в цепи якоря;
 q – переносимый электрический заряд.

2) Уравнение механического звена двигателя, работающего в условиях наличия сухого (M_c) и вязкого трения с коэффициентом вязкости β_c

$$M - M_c = J \frac{d\omega}{dt} + \beta_c \omega, \quad \omega = \frac{d\varphi}{dt},$$

где φ – угол поворота вала двигателя, а положительное направление момента M_c выбирается противоположным положительному направлению момента двигателя M .

В первом примере имеем: Q_j – это переносимый заряд q , U_j – это ЭДС и падение напряжения в рассматриваемом контуре; во втором примере Q_j – это угол поворота φ , U_j – это момент двигателя и моменты сопротивления в рассматриваемом механическом звене.

Полный алгоритм функционирования ТО получают путем установления связей между входящими в его состав динамическими звеньями. В результате получают многоконтурную модель, которая будет представлена в виде односвязной или многосвязной системы. Односвязная (одномерная) система получается тогда, когда модель ТО имеет один вход и один выход. Однодвигательные электроприводы, особенно электроприводы постоянного тока, чаще всего отображаются односвязными моделями. Однако электроприводы и другие ТО, входящие в состав АСУТП, обычно объединены различными связями в один комплекс. Их совместную работу невозможно рассматривать отдельно, без учета их взаимодействия и взаимовлияния. Поэтому алгоритм функционирования ТО, управляемого от АСУТП, обычно представляется в виде многосвязной (многомерной) системы, в которой учтено взаимодействие входных параметров, их совместное воздействие на выходные параметры и перекрестные обратные связи от выходных параметров, действующие в ТО. Для анализа подобных систем строят математические модели в матричной форме.

Рассмотрим, например систему уравнений, отображающую некий ТО в виде трехсвязной (трехмерной) линейной системы:

$$\left. \begin{aligned} a_{11}y_1 + a_{12}y_2 + a_{13}y_3 &= b_{11}x_1 + b_{12}x_2 + b_{13}x_3 \\ a_{21}y_1 + a_{22}y_2 + a_{23}y_3 &= b_{21}x_1 + b_{22}x_2 + b_{23}x_3 \\ a_{31}y_1 + a_{32}y_2 + a_{33}y_3 &= b_{31}x_1 + b_{32}x_2 + b_{33}x_3 \end{aligned} \right\}, \quad (3.4)$$

где x_1, x_2, x_3 – входные переменные, y_1, y_2, y_3 – выходные переменные, а a_{ij} и b_{ij} – коэффициенты (некоторые из них могут быть равны нулю). Слагаемые с x_2 и с x_3 , а также с y_2 и с y_3 , имеющиеся в первом уравнении системы (3.4), отображают объективно существующие в данном ТО перекрестные связи между его параметрами. Аналогичные связи отображены и в остальных уравнениях системы (3.4). Так, слагаемое $b_{32}x_2$ в третьем уравнении отображает, возможно нежелательное, но объективно существующее перекрестное влияние управляющего воздействия x_2 , предназначенного для управления параметром y_2 , на параметр y_3 , а

слагаемое $a_{23}y_3$ во втором уравнении отображает перекрестную обратную связь с выхода y_3 на параметр y_2 . Если коэффициенты b_{32} и a_{23} равны нулю, то это означает, что указанные перекрестные связи отсутствуют.

С помощью системы уравнений типа (3.4) можно анализировать как статические, так и динамические режимы, если при анализе динамики заменить дифференциальные уравнения их операторными отображениями. В матричной форме данная система отображается так:

$$\mathbf{AY}=\mathbf{BX}, \quad (3.5)$$

где $\mathbf{A}$, $\mathbf{B}$, $\mathbf{X}$, и $\mathbf{Y}$ – записанные в табличной форме (в виде матриц) коэффициенты и переменные системы (3.4):

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \mathbf{Y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$$

Решение уравнений, записанных в матричной форме, облегчается наличием специализированного программного обеспечения операций с матрицами, предназначенного для исследования систем автоматизации.

Во многих случаях модели ТО существенно нелинейны. В качестве примера рассмотрим трехмерную систему уравнений, описывающих движение механизма по прямой линии со скоростью v из начала координат в точку с координатами x, y и z :

$$\left. \begin{aligned} \frac{dx}{dt} &= \frac{vx}{\sqrt{x^2 + y^2 + z^2}} \\ \frac{dy}{dt} &= \frac{vy}{\sqrt{x^2 + y^2 + z^2}} \\ \frac{dz}{dt} &= \frac{vz}{\sqrt{x^2 + y^2 + z^2}} \end{aligned} \right\} \quad (3.6)$$

Уравнения (3.6) задают проекции скорости v перемещения рабочего органа в пространстве на координаты x, y и z в пределах текущего кадра программы. Уравнения такого типа используются в системах ЧПУ при вычислениях промежуточных точек заданной траектории движения, когда движение между заданными точками должно производиться по прямой линии (линейная интерполяция). Система уравнений (3.6) соответствует случаю, когда начало координат совмещается с начальной точкой очередного линейного участка траектории (с начальной точкой кадра программы), а координаты x, y и z соответствуют конечной точке в данном кадре, что соответствует заданию перемещений в приращениях. Общий вид траектории движения определяется законом изменения координат x, y

и z от кадра к кадру. Чтобы пользоваться типовыми программами решения уравнений, записанных в матричной форме, нелинейные уравнения линеаризуются в окрестности интересных для исследования поведения технологического объекта точек.

3.3. Моделирование технологических циклов

Технологические процессы имеют циклический характер и обычно представляют собой последовательность сменяющих друг друга технологических операций. Описание технологических циклов является неотъемлемой частью моделирования технологических объектов и основой формирования управляющих программ АСУТП.

Для удобства описания технологический цикл делится на такты, или состояния. Каждый такт охватывает промежуток времени, в течении которого управляющие параметры, характеризующие состояние командных (кнопки, переключатели, контакторы) и исполнительных (приводы, муфты) органов, остаются неизменными. Далее производится анализ переходов от одного такта к другому как при нормальном ходе техпроцесса, так и при различного рода отклонениях и в аварийных ситуациях. По результатам анализа составляется формализованное описание (модель) технологического цикла с применением таблиц, циклограмм, граф-схем, логических формул и пр. Различают две разновидности моделей технологических циклов: комбинационные и последовательностные. Состояние выходов *комбинационных моделей* определяется исключительно текущей комбинацией входных сигналов и не зависит от такта, в котором наблюдается данная комбинация, а состояние выходов *последовательностных моделей* зависит также и от комбинаций входных и выходных сигналов, имевших место в предыдущих тактах технологического цикла.

В течение технологического цикла переход от такта к такту осуществляется путем включения и отключения отдельных узлов, модулей и элементов технологического объекта. Включенное состояние обычно обозначают единицей (1), а отключенное – нулем (0). В качестве комбинационных моделей чаще всего используют таблицы истинности (см. приложение 3, §П3.4), в которых отмечают значения выходных параметров, соответствующие всем возможным комбинациям входных параметров технологического объекта. Если имеется n входных (управляющих) параметров, каждый из которых может принимать значение либо 0, либо 1, то они могут составить 2^n различных комбинаций, некоторые из которых на практике не могут быть реализованы при нормальной работе оборудования. Например, не могут быть сформированы сигналы о том, что кабина лифта находится сразу на нескольких этажах. В таблице истинности должны быть представлены только реализуемые комбинации входных

сигналов, соответствующие предусмотренным режимам работы технологического объекта. Остальные комбинации рассматриваются на этапе проектирования системы управления технологическим объектом.

В качестве примера рассмотрим таблицу истинности датчика положения, работающего в циклическом коде Грея, приведенном в табл.2.1. Такой датчик может фиксировать до 16 положений, которые в табл.2.1 имеют номера от 0 до 15. Если каждому положению соответствуют какие-либо операции системы управления, управляющей механизмом, которому принадлежит данный датчик, то можно считать, что алгоритм функционирования механизма состоит из 16 тактов. В таком случае таблица истинности рассматриваемого датчика охватывала бы все 16 позиций табл.2.1 в ее верхней части, задающей код Грея. Переход от такта к такту здесь задается последовательной сменой комбинаций сигналов датчика, т.е. потенциалов на его выходных клеммах, по мере перемещения механизма. В другом случае, когда нужно фиксировать только шесть каких-либо положений, можно было бы применить другой датчик, работающий в таком же коде Грея, но с использованием только первых шести кодовых комбинаций (0-5). Этот код тоже был бы циклическим, но охватывал бы шесть кодовых комбинаций из возможных шестнадцати. Остальные кодовые комбинации не вошли бы в состав таблицы истинности, но могли быть использованы при проектировании системы управления механизмом, положение которого контролируется рассматриваемым датчиком, например - для организации контроля четности (см. §2.5).

Итак, комбинационные модели отличаются тем, что каждой предусмотренной комбинации входных сигналов соответствует один и тот же выходной сигнал. У последовательностных же моделей одной и той же комбинации входных сигналов могут соответствовать различные выходные сигналы в зависимости от последовательности комбинаций входных и выходных сигналов, предшествовавшей данной комбинации. Предыстория текущего состояния технологического объекта фиксируется с помощью особых запоминающих устройств, которые при моделировании приобретают вид *промежуточных переменных*, часть которых может совпадать с выходными сигналами. Простые последовательностные модели составляются в виде *циклограмм* или *граф-схем*. Циклограмма строится в виде таблицы, в строках которой перечислены все командные и исполнительные элементы технологического объекта, а столбцы соответствуют тактам его рабочего цикла. Включенное состояние элемента показывают сплошной линией в соответствующей строке, а вертикальными линиями показывают команды на включение и отключение исполнительных устройств в

соответствующих тактах циклограммы. В качестве примера рассмотрим циклограмму работы грузового подъемника (рис. 3.1).

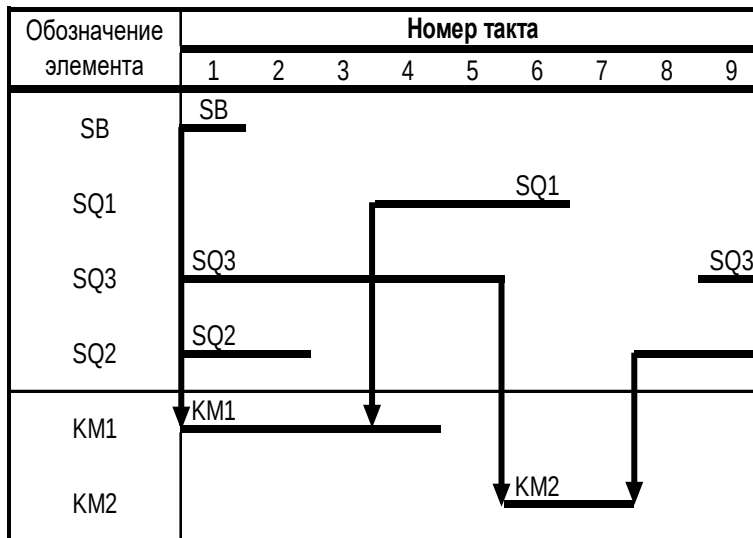


Рис. 3.1. Циклограмма работы грузового подъемника

Грузовой подъемник включается кнопкой *SB* при условии, что на нем помещен условленный груз (нажат конечный выключатель *SQ3*) и выключен контактор *KM2* (взаимная блокировка контакторов). Нажатие кнопки и отпускание кнопки *SB* составляет такт 1 циклограммы. Оно приводит к включению контактора *KM1*, а через него – к включению привода подъемника на подъем, что приводит к отключению нижнего конечного выключателя *SQ2* (такт 2). Последнее не приводит к отключению привода, подъем продолжается, пока подъемник не достигнет крайнего верхнего положения, в котором включается конечный выключатель *SQ1* (такт 3). Включение *SQ1* ведет к отключению контактора *KM1* и к остановке привода (такт 4). После снятия груза с подъемника освобождается конечный выключатель *SQ3*, что приводит к включению контактора *KM2* (такт 5). Последний включает привод подъемника на движение вниз. С началом движения вниз освобождается, конечный выключатель *SQ1*, фиксировавший крайнее верхнее положение подъемника (такт 6). Движение вниз продолжается до тех пор, пока не будет нажат нижний выключатель *SQ2* (такт 7), который отключает контактор *KM2*, а с ним и привод подъемника (такт 8). На подъемник устанавливается новый груз, что фиксируется нажатием конечного выключателя *SQ3* (такт 9). На этом цикл функционирования подъемника завершается.

Представление технологического цикла в виде *графа* начинается с перечисления всех возможных состояний технологического объекта, которые обозначаются кружками, в каждом из которых проставлен номер или код состояния. Указанные кружки являются вершинами конструируемого графа. Вершины соединяются стрелками, указывающими направление возможных переходов от одного

состояния к другому. Над стрелками надписывают значения управляющих сигналов, комбинации которых обеспечивают переходы, указанные стрелками. Рассмотрим составление графа (рис.3.2), отображающего работу подъемника, циклограмма которого представлена на рис. 3.1. Как видно на рис.3.2, граф имеет три вершины, обозначенные в двоичном коде:

- 00 – неподвижное состояние ($KM1=0$ и $KM2=0$);
- 01 – движение вверх ($KM1=1$, $KM2=0$);
- 10 – движение вниз ($KM1=0$, $KM2=1$).

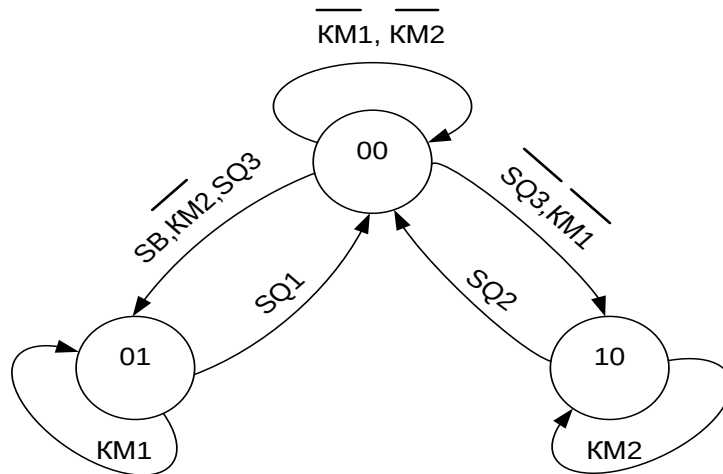


Рис. 3.2. Граф-схема функционирования грузового подъемника

Пуск вверх путем включения контактора $KM1$ начинается, если нажата кнопка SB , выключен контактор $KM2$ и нажат конечный выключатель $SQ3$, сигнализирующий наличие груза. Движение вверх продолжается независимо от сохранения перечисленных пусковых условий, но движение вниз при этом блокируется, как видно на граф-схеме ($KM1=0$), пока идет движение вверх. Последнее прекращается после нажатия $SQ1$, сигнализирующего о достижении подъемником заданного верхнего положения. Движение вниз начинается посредством включения контактора $KM2$, после отключения контактора $KM1$ и снятия груза с подъемника ($SQ3=0$). Оно прекращается только после достижения заданного нижнего положения по сигналу $SQ2=1$.

Более сложные последовательностные модели требуют составления *таблиц состояний*, которые будут рассмотрены ниже при изучении методов синтеза алгоритмов управления. Наиболее сложные модели получаются при учете действия случайных факторов, прежде всего – отказов оборудования (стохастические модели).

3.4. Экспериментальные методы получения моделей технологических объектов

3.4.1. Одномерные модели

Экспериментальное исследование технологических объектов и входящих в их состав звеньев обычно имеет целью подтверждение правильности моделей, составленных аналитическими методами. Однако возможны случаи, когда аналитическая модель отсутствует. Это возможно, если объект недостаточно изучен или если разработка аналитической модели до проведения экспериментальных исследований слишком трудоемка и экономически не оправдана. В таких случаях ставится задача получения моделей на основе обработки результатов эксперимента.

Процесс установления соответствия итогов экспериментальных исследований теоретическим представлениям об исследуемом объекте принято называть *идентификацией* (опознаванием) объекта. Этот термин введен в немалой степени потому, что результаты экспериментальных исследований зависят от ряда трудно учитываемых факторов, многие из которых имеют случайный характер. К таким факторам относятся:

- разброс параметров изучаемых объектов;
- изменение параметров объектов в процессе испытаний, в том числе и отказы оборудования;
- разброс параметров обрабатываемых заготовок и материалов;
- разброс показаний (погрешности) измерительной аппаратуры.

Действие указанных факторов приводит к тому, что при одних и тех же входных (управляющих) воздействиях в разных сериях испытаний значения выходных параметров испытуемого объекта получаются различными. При моделировании технологического объекта по результатам экспериментальных исследований возникает задача оценки разброса выходных параметров и определения их однозначной (детерминированной) зависимости от управляющих воздействий в условиях действия случайных факторов. Для получения надежной детерминированной модели технологического объекта в условиях разброса результатов экспериментальных исследований используют методы теории вероятностной (см. приложение 2), на которых основан *регрессионный анализ* результатов эксперимента.

Регрессией выходного параметра y на входной параметр x мы будем называть любую функцию $f(x)$, приближенно представляющую вероятностную зависимость y от x . В результате функция y представляется в виде суммы:

$$y = f(x) + h(x, y)$$

где $h(x, y)$ – поправочный член.

На первый план обычно выдвигается задача определить, как в *среднем* изменяется величина y при изменении управляющего воздействия x . Эта задача лучше всего решается с помощью *функции регрессии* $y=g(x)$, где

$$g(x) = \sum_{j=1}^m y_j P_{y/x}(y_j), \quad (3.7)$$

где m – количество различных значений y , полученных из опытов, произведенных при заданном значении x ;

y_j – значения y , полученные при заданном значении x ;

$P_{y/x}(y_j)$ – условная вероятность того, что $y=y_j$ при заданном значении x .

При практическом определении $y=g(x)$ исходят из соотношения;

$$y_i = \frac{1}{k} \sum_{j=1}^k y_j, \quad (3.8)$$

где k – количество произведенных измерений величины y при $x=x_i$.

При $k \rightarrow \infty$ имеет место $y_i \rightarrow g(x_i)$.

Функция регрессии $g(x)$ отличается тем, средний квадрат отклонения ее от искомой функции $y(x)$ меньше среднего квадрата отклонения y от любой другой функции $f(x)$, приближенно представляющей вероятностную зависимость $y(x)$. В общем случае функция регрессии имеет нелинейный характер, в связи с чем возникает задача ее линеаризации. Наилучшим линейным приближением вероятностной зависимости $y(x)$ является *линейная регрессия y на x* , которая может быть представлена в таком виде:

$$y = r \frac{S_y}{S_x} (x - x_0) + y_0, \quad (3.9)$$

где y_0 – среднее значение y в заданном диапазоне измерения x :

$$y_0 = \frac{1}{n} \sum_{i=1}^n y_i, \quad (3.10)$$

x_0 – среднее значение x в заданном диапазоне:

$$x_0 = \frac{1}{n} \sum_{i=1}^n x_i, \quad (3.10')$$

n – количество экспериментальных точек $y(x)$ на аппроксимируемом интервале;

S_y и S_x – несмещенные стандартные отклонения y и x от их средних значений:

$$S_y = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (y_i - y_0)^2}, \quad (3.11)$$

$$S_x = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (x_i - x_0)^2}, \quad (3.11')$$

r – эмпирический коэффициент корреляции:

$$r = \frac{1}{(n-1)S_y S_x} \sum_{i=1}^n (y_i - y_0)(x_i - x_0). \quad (3.12)$$

Зависимость (3.9) называют *эмпирической прямой регрессии*, причем коэффициент r , который принимает значения $-1 \leq r \leq 1$, показывает, насколько хорошо в среднем может быть представлена величина y в виде линейной функции от x . Если все экспериментальные точки $y(x)$ лежат на одной прямой, то $|r|=1$, а линейная зависимость $y(x)$ является абсолютно точной. При $r=0$ величины y и x являются *некоррелированными*. Отсюда следует, что при малых r связывать y и x линейной зависимостью не имеет смысла.

Пример 3.1. В таблице 3.1 приведена зависимость тока I , потребляемого нагревательным элементом, от напряжения U на его клеммах. Данные записывались один раз в сутки. Числа m показывают, сколько раз записывались одинаковые пары U и I . Считая зависимость $I(U)$ линейной, определить сопротивление R_n нагревательного элемента.

Таблица 3.1

$U, В$	23	24	24,5	24,5	25	25,5	26	26	26	26,5	26,5	27	27	28
$I, А$	0,22	0,24	0,23	0,24	0,25	0,26	0,23	0,25	0,27	0,24	0,26	0,28	0,26	0,27
m	2	4	3	2	1	1	2	1	2	1	1	2	1	3

Обозначаем $U=x$, а $I=y$ и по формулам 3.10-3.12 (учитывая, что общее количество точек $I(U)$; с учетом m , равно 26, т.е. $n=26$) определяем: $x_0=25,5$; $y_0=0,248$; $S_x=1,49$; $S_y=0,0185$; $r=0,793$.

Соответственно линейная регрессия y на x согласно (3.9), такова:

$$y = 0,793 \frac{0,0185}{1,49} (x - 25,5) + 0,248.$$

После упрощения и возврата к U и I получим $I \approx 0,0098 U$;

$$R_t = \frac{1}{0,0098} = 102 \text{ Ом.}$$

Если линейная регрессия неудовлетворительна (коэффициент корреляции r близок к нулю), то применяют более точные аппроксимации уравнения регрессии (3.7) с помощью полинома вида

$$y = \sum_{j=0}^m a_j x^j, \quad m > 1,$$

называемого параболической регрессией порядка m , или с помощью функции

$$y = a e^{bx},$$

называемой экспоненциальной регрессией, особенно полезной при идентификации поведения технологического объекта в динамике. Коэффициенты аппроксимирующих функций выбирают таким образом, чтобы было минимизировано суммарное квадратичное отклонение вида (3.1) избранной регрессии от экспериментальных данных.

3.4.2. Многомерные модели

До сих пор мы рассматривали идентификацию одномерных объектов, моделируемых функциями одного переменного. Такой подход применим, в основном, при моделировании отдельных звеньев технологических объектов. Последние же являются, как правило, многосвязными, многомерными системами. Модели таких объектов получают посредством *факторного анализа*, который называют также *методом планирования эксперимента*. Данный метод базируется на линейной аппроксимации функций исследуемых объектов. Ввиду разброса результатов экспериментальных исследований многомерная линейная модель формируется как линейная регрессия моделируемого выходного параметра y на управляющие (входные) параметры $x_1, x_2, \dots, x_m$.

Уравнение линейной регрессии y на $x_1, x_2, \dots, x_m$ записывается в таком виде:

$$y = y_0 + \sum_{i=1}^m a_i (x_i - x_{i0}), \quad a_i = -\frac{A_{yi}}{A_{ii}}, \quad (3.13)$$

где A_{yi} и A_{ji} – обобщенный корреляционный момент, и обобщенная дисперсия, выраженные в матричной форме;

y_0 и x_{i0} – средние значения y и x_i , рассчитанные по формулам (3.10) в заданном диапазоне изменения y и x_i .

Факторы $x_1, x_2, \dots, x_m$, существенно влияющие на выходной параметр y , обычно представляют в относительных величинах:

$$x_{i\Delta} = \frac{x_i - x_{i0}}{\Delta x_i}, \quad (3.14)$$

где $\Delta x_i = x_{imax} - x_{i0}$;

x_{imax} – максимальное запланированное значение x_i .

После подстановки $x_{i\Delta}$ в уравнение (3.13), оно будет представлено в виде

$$y = y_0 + \sum_{i=1}^m a_{i0} x_{i\Delta}, \quad (3.15)$$

где $a_{i0} = a_i \Delta x_i$.

Задачей факторного анализа является рациональное планирование экспериментальных исследований изучаемого объекта, позволяющее по их результатам наименее трудоемким путем определить коэффициенты $a_1, a_2, \dots, a_m$ уравнения линейной регрессии (3.13), моделирующего данный объект. Коэффициенты определяются таким образом, чтобы расчетные значения y , полученные по формуле (3.15), были равны экспериментальным значениям y , полученным при тех же значениях $x_{i\Delta}$. Если будут проведены n циклов экспериментального определения значений y при различных наборах значений входных параметров (факторов) x_i , то для определения параметров a_i можно будет составить n уравнений вида

$$y_j = y_0 + \sum_{i=1}^m a_{i0} x_{ij}, \quad j = 1, 2, \dots, n, \quad (3.16)$$

где y_j – значение y , полученное в j -том эксперименте;
 x_{ij} – относительное значение x_i в j -том эксперименте.

В матричной форме эта система уравнений может быть представлена в виде:

$$Y = X_0 A_0, \quad (3.16')$$

где Y – матрица-столбец экспериментальных значений y с числом элементов, равным n ;

X_0 – матрица входных воздействий размером $n \times m$;

A_0 – матрица-столбец коэффициентов a_{i0} с числом элементов, равным m .

Значения x_i при проведении экспериментальных исследований варьируют таким образом, чтобы решение системы уравнений (3.16) относительно коэффициентов a_{i0} можно было представить в виде:

$$a_{i0} = \frac{\sum_{j=1}^n y_j x_{ij}}{n}, \quad (3.17)$$

где n – количество опытов по определению значений y_i .

Для этого эксперименты проводят в следующем порядке:

- в каждом эксперименте полагают либо $x_i = x_{imax}$, либо $x_i = x_{imin}$, с тем чтобы получать всегда $x_{i0} = 1$, либо $x_{i0} = -1$, для всех варьируемых факторов (условие нормированности);
- сумма всех экспериментальных значений каждого фактора x_{i0} в серии из n экспериментов должна быть равна нулю, т.е. должно быть

$$\sum_{j=1}^n x_{ij} = 0, \quad i = 1, 2, \dots, n, \quad (\text{условие симметричности})$$

- сумма всех произведений $x_{i0} \times x_{(i-1)0}$ в серии из n экспериментов должна быть равна нулю (условие ортогональности).

Значения x_{imax} и x_{imin} выбираются в зависимости от целей линеаризации параметра y . Чем ближе значение x_{imin} к x_{imax} , тем точнее аппроксимация функции y в области значений x_i от x_{imax} до x_{imin} , но тем сложнее ограничивать разброс экспериментальных значений параметров в пределах, обеспечивающих заданную точность.

Если следовать рассмотренному порядку экспериментальных исследований, то окажется, что в ходе эксперимента каждый параметр (фактор) будет принимать только два значения: x_{imax} и x_{imin} . И если варьируется m факторов, то максимальное количество различных вариантов в одном цикле экспериментальных исследований составит 2^m . Если в экспериментальном цикле реализуются все 2^m вариантов, то такой эксперимент называют *полнофакторным*. Полнофакторный

эксперимент всегда удовлетворяет условиям нормированности, симметричности и ортогональности. Он также обеспечивает максимальную точность расчетов. Однако, если желательно уменьшить объем эксперимента, то можно применить *дробнофакторный эксперимент*, когда в цикл экспериментальных исследований входят не все возможные варианты сочетаний факторов, но условия нормированности, симметричности и ортогональности все же выполняются.

Пример 3.2. Аппроксимировать зависимость объемной производительности J токарного станка от глубины резания t_p и подачи на оборот s при постоянной скорости резания уравнением регрессии вида

$$J = a_0 + a_1 t_p + a_2 s + a_3 t_p s. \quad (3.18)$$

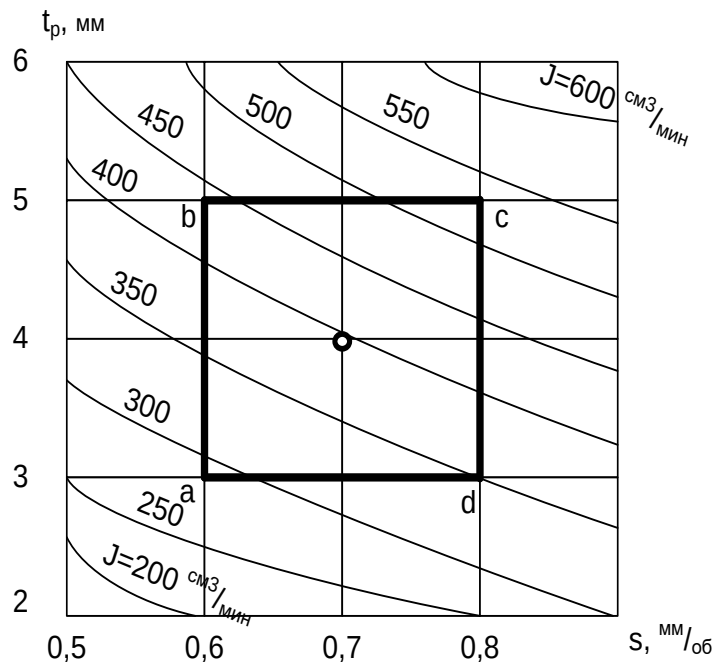


Рис.3.3. График производительности токарного станка

Экспериментальные усредненные зависимости t_p при различных значениях параметра J приведены на рис.3.3. Пределы изменения глубины резания полагаем от $t_{pmin}=3$ мм до $t_{pmax}=5$ мм, а подачи на оборот – от $s_{min}=0,6$ мм/об до $s_{max}=0,8$ мм/об. Область определения функции $J(t_p, s)$ показана на рис.3.3 в виде прямоугольника a, b, c, d .

Усредненность кривых $t_p(s)$ позволяет полагать:

$$t_{p0} = \frac{t_{pmax} + t_{pmin}}{2} = 4 \text{ мм}, \quad \Delta t_p = t_{pmax} - t_{p0} = 1 \text{ мм};$$

$$s_0 = \frac{s_{max} + s_{min}}{2} = 0,7 \text{ мм/об}, \quad \Delta s = s_{max} - s_0 = 0,1 \text{ мм/об}.$$

Кроме того, полагаем:

$$x_{1э} = \frac{t_p - t_{p0}}{\Delta t_p}; \quad x_{2э} = \frac{s - s_0}{\Delta s}; \quad x_{3э} = \frac{t_p - t_{p0}}{\Delta t_p} \cdot \frac{s - s_0}{\Delta s},$$

что позволяет искать выражение (3.18) в форме линейной регрессии:

$$J = J_0 + a_{10}x_{1э} + a_{20}x_{2э} + a_{30}x_{3э} \quad (3.19)$$

в соответствии с выражениями (3.15) и (3.16), где полагаем $y=J$. В рассматриваемом случае $m=3$, что ведет к полнофакторному эксперименту для определения значений a_{10} , a_{20} и a_{30} в объеме восьми опытов. С целью уменьшения объема эксперимента ограничимся четырьмя опытами ($n=4$), проведенными в порядке, указанном в таблице 3.2.

Таблица 3.2

№ опыта	$x_{1э}$	$x_{2э}$	$x_{3э}$	$Y_j=J_j$
1	-1	-1	1	285
2	1	-1	-1	435
3	-1	1	-1	350
4	1	1	1	530

Значения $x_{1э}$ и $x_{2э}$ равны +1 при максимальных значениях t_p и s и -1 - при минимальных значениях t_p и s . Значения $x_{3э}$ равны произведению значений $x_{1э}$ и $x_{2э}$. Экспериментальные значения J , соответствующие принятым значениям $x_{1э}$, $x_{2э}$ и $x_{3э}$, определены по точкам a , b , d и c на рис. 3.3. Как видно в таблице 3.2, дробнофакторный эксперимент, отображенный в ней, соответствует требованиям нормированности, симметричности и ортогональности. Поэтому, согласно (3.17), вычисляем:

$$a_{10} = \frac{-285 + 435 - 350 + 530}{4} = 82;$$

аналогично получим $a_{20}=35$ и $a_{30}=7,5$. Кроме того, согласно (3.10) получим:

$$J_0 = \frac{1}{n} \sum_{i=1}^n J_i = \frac{285 + 435 + 350 + 530}{4} = 400,$$

после чего выражение (3.19) будет представлено в виде:

$$J = 400 + 82,5 \frac{t_p - 4}{1} + 40 \frac{s - 0,7}{0,1} + 7,5 \frac{t_p - 4}{1} \frac{s - 0,7}{0,1}.$$

Последнее выражение приводится к виду (3.18) в такой форме:

$$J = 30t_p + 100s + 75t_p s,$$

так как $a_0=0$.

Получение аналитического выражения $J(t_p, s)$ было обеспечено наличием усредненных экспериментальных зависимостей $t_p(s)$, приведенных на рис. 3.3. Определение зависимостей такого рода в условиях разброса экспериментальных данных требует применения регрессионного анализа в соответствии с выражениями (3.7) и (3.8),

причем необходимо, чтобы объем и число повторений эксперимента обеспечивали нужный уровень достоверности модели.

Контрольные вопросы к главе 3

1. Что такое алгоритм функционирования, или модель ТО?
2. В чем состоит декомпозиция алгоритма ТО?
3. Как оценить эффективность построенной модели ТО?
4. В каком порядке производится построение аналитической модели ТО? Приведите пример типового звена ТО.
5. В чем отличие односвязных и многосвязных моделей ТО?
6. Приведите пример записи уравнений многосвязной линейной системы с перекрестными связями.
7. В каком порядке строится модель технологического цикла с помощью таблиц истинности?
8. Как реализуются последовательностные модели в виде циклограмм?
9. Как составляются граф-схемы технологического цикла?
10. В чем состоит экспериментальная идентификация ТО? Каковы ее трудности?
11. В чем состоит регрессионный анализ при экспериментальном исследовании параметров ТО?
12. Как используется уравнение линейной регрессии при экспериментальной идентификации ТО? Приведите пример.
13. Как записывается уравнение линейной регрессии многомерного (многосвязного) объекта?
14. В каком порядке производится факторный анализ многомерного объекта?
15. Как производится полнофакторный эксперимент? Приведите пример.
16. Разберите пример неполнофакторного эксперимента при определении зависимости объемной производительности токарного станка от глубины резания и подачи.

Глава 4. Алгоритмы управления АСУТП

4.1. Задачи управления в АСУТП

Для решения основной задачи АСУТП, заключающейся в оптимизации функционирования технологического объекта (ТО), необходимо обеспечить решение ряда частных задач, которые подразделяются на три типа:

- 1) стабилизация параметров техпроцесса при различных возмущениях;
- 2) программное управление технологическим циклом;
- 3) автоматическая оптимизация техпроцесса в ходе выполнения производственной программы.

Решение этих задач описывается алгоритмами, которые мы будем называть *алгоритмами управления*. Алгоритмы управления определяют порядок подачи управляющих воздействий на исполнительные устройства АСУТП. Они должны обеспечивать программирование работы АСУТП в соответствии с заданным критерием оптимальности (целевой функцией) функционирования управляемого ТО. В зависимости от характера функционирования ТО и его составных частей алгоритмы управления АСУТП делятся на три группы:

- 1) алгоритмы стабилизации значений управляющих параметров на уровнях, обеспечивающих оптимальный ход техпроцесса;
- 2) алгоритмы оптимального программно-следящего управления в рамках технологического цикла;
- 3) алгоритмы автоматической оптимизации техпроцесса в процессе его реализации.

Последние применяются в тех случаях, когда оптимальные параметры техпроцесса не могут быть определены и заданы заранее.

4.2. Алгоритмы стабилизации управляющих параметров

Алгоритмы стабилизации предназначены для поддержания значений параметров АСУТП на заданном уровне. Известны два основных приема стабилизации: по отклонению и по возмущению. При стабилизации по отклонению измеряется отклонение регулируемого параметра от заданного значения, а затем вырабатывается управляющее воздействие, обеспечивающее ликвидацию возникшего по той или иной причине отклонения. Если регулирующее воздействие достаточно действенно, то возникшее отклонение будет в конце концов ликвидировано, но предотвратить отклонение таким способом невозможно, так как регулирующее воздействие формируется только *после* возникновения отклонения.

При стабилизации по возмущению сигнал, пропорциональный возмущающему воздействию, подается в регулирующее устройство,

где преобразуется таким образом, чтобы в месте воздействия возмущения было сформировано управляющее воздействие, равное по величине и противоположное по знаку возмущающему воздействию. В таком случае возмущение оказывается полностью скомпенсированным и не сможет вызвать отклонения регулируемого параметра от заданного значения. Однако такое отклонение может иметь место в силу наличия возмущений какой-либо иной природы. Скажем, если скомпенсировано возмущение электропривода нагрузкой, то отклонение регулируемого параметра данного электропривода от заданного значения может быть вызвано, например, изменением температуры окружающей среды и многими другими факторами. Кроме того, стабилизация параметров АСУТП осложняется дополнительно тем обстоятельством, что АСУТП являются многосвязными системами, у которых изменение одного параметра приводит к изменению многих других взаимно влияющих друг на друга параметров.

Будем считать, что каждый управляемый параметр АСУТП задается особым задающим воздействием (уставкой). Совокупность звеньев АСУТП, ведущих от задающего воздействия к месту формирования параметра, для управления которым предназначено данное задающее воздействие, будем называть *каналом управления*. Управляемые координаты АСУТП зависят, как правило, не только от значения «своего» управляющего воздействия и внешних возмущений, но и от других управляющих воздействий, предназначенных для управления другими параметрами АСУТП. Управляющие воздействия других каналов управления являются для рассматриваемого канала внутренними (в рамках АСУТП) возмущающими воздействиями. Также и внешние возмущения могут воздействовать на различные каналы управления по-разному. Взаимное влияние каналов управления друг на друга принято учитывать в виде *перекрестных связей* этих каналов. Модели многоканальных объектов с перекрестными связями представляют собой, как показано в §3.2, многомерные системы, описываемые алгебраическими и дифференциальными уравнениями. Задача стабилизации параметров в многомерной системе решается как путем учета и компенсации возмущений, являющихся причиной отклонения параметров от заданных значений, так и путем ликвидации возникших отклонений.

Многоканальные управляющие системы, в которых с заданной точностью достигается компенсация действия возмущений, называются *инвариантными* по отношению к компенсируемым возмущениям. Рассмотрим пути достижения инвариантности стабилизируемых параметров АСУТП по отношению к внешним и

внутренним возмущениям, считая модель системы автоматизации линеаризованной.

В общем виде систему линейных уравнений, описывающих функционирование технологического объекта удобно (см. §3.2) представлять в матричном виде:

$$\mathbf{A}\mathbf{Y} = \mathbf{B}_0\mathbf{X} + \mathbf{D}_0\mathbf{F}, \quad (4.1)$$

где: $\mathbf{X}$ – совокупность задающих воздействий $x_1, x_2, \dots, x_n$, отображенных в виде матрицы-столбца;

$\mathbf{Y}$ – совокупность управляемых параметров ТО $y_1, y_2, \dots, y_n$, задаваемых уставками $x_1, x_2, \dots, x_n$ и отображенных также в виде матрицы-столбца;

$\mathbf{F}$ – матрица-столбец внешних возмущений $f_1, f_2, \dots, f_m$, действующих на ТО;

$\mathbf{A}$, $\mathbf{B}_0$ и $\mathbf{D}_0$ – матрицы исходных коэффициентов, полученных при записи системы уравнений ТО.

При записи системы уравнений (4.1) мы полагали, что каждый параметр y_i управляется посредством собственного задающего сигнала x_i , но одновременно испытывает возмущающие воздействия как от некоторых “соседних” задающих воздействий, так и от “соседних” управляемых координат и некоторых внешних возмущений. Следовательно, в рассматриваемом случае количество управляемых координат равно количеству задающих воздействий и исходные матрицы $\mathbf{A}$ и $\mathbf{B}_0$ являются квадратными матрицами ранга n , где n – количество управляемых координат (параметров). Что касается количества m внешних возмущений f_i , то оно не зависит от количества управляемых параметров и определяется лишь свойствами ТО. По этой причине исходная матрица $\mathbf{D}_0$ имеет размер $n \times m$.

Для решения матричного уравнения (4.1) относительно управляемых параметров $\mathbf{Y}$ домножим обе части его на обратную матрицу $\mathbf{A}^{-1}$ и после некоторых преобразований получим:

$$\mathbf{Y} = \mathbf{B}\mathbf{X} + \mathbf{D}\mathbf{F} \quad (4.2),$$

где $\mathbf{B} = \mathbf{A}^{-1}\mathbf{B}_0$, а $\mathbf{D} = \mathbf{A}^{-1}\mathbf{D}_0$ – матрицы того же размера, что и исходные матрицы $\mathbf{B}_0$ и $\mathbf{D}_0$.

Полная инвариантность параметров $\mathbf{Y}$, описываемых уравнением (4.2), к любому возмущению из учтенных нами с помощью матриц $\mathbf{B}$ и $\mathbf{D}$ или, иными словами, абсолютная стабилизация всех параметров $\mathbf{Y}$ будет достигнута, если каждый параметр y_i будет строго пропорционален управляющему им задающему воздействию x_i . Следовательно, для реализации полной инвариантности параметров $\mathbf{Y}$ необходимо соблюдение двух условий:

- 1) матрица $\mathbf{B}$ должна быть диагональной;
- 2) матрица $\mathbf{D}$ должна быть нулевой, т.е. все ее элементы должны быть равны нулю.

Матрица **B** есть совокупность коэффициентов, связывающих значения управляемых параметров, составляющих вектор **Y**, с задающими воздействиями, составляющими вектор **X**. Ее диагональность означает, что не равны нулю только коэффициенты, связывающие управляемые параметры с собственными задающими воздействиями. Действительно, если выполняются условия инвариантности, то уравнение (4.2) разлагается на следующие простейшие равенства:

$$y_1=b_{11}x_1, y_2=b_{22}x_2, \dots y_n=b_{nn}x_n, \quad (4.2')$$

свидетельствующие о независимости значений управляемых параметров как от сигналов, управляющих соседними каналами, так и от внешних возмущений. В этих равенствах коэффициенты $b_{11}, b_{22}, \dots b_{nn}$ – это элементы матрицы **B**, расположенные по ее диагонали начиная с элемента b_{11} , расположенного в начале первой строки и первого столбца, и до элемента b_{nn} , расположенного в конце последних строки и столбца матрицы **B**.

В реальных ТО условия инвариантности обычно не соблюдаются. Достижение в необходимых случаях инвариантности по отношению к тем или иным видам возмущений является одной из основных задач проектирования АСУТП. Принцип достижения инвариантности состоит в том, что для компенсации возмущения, действующего в каком-либо канале, формируется дополнительный управляющий канал, по которому проходит компенсирующее воздействие, равное по величине и противоположное по знаку компенсируемому возмущению в точке его приложения. Рассмотрим

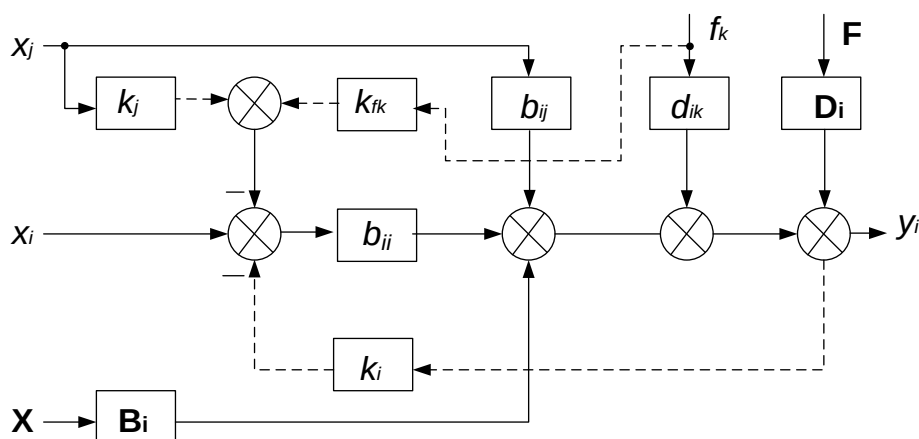


Рис.4.1 Схема реализации *i*-того канала управления с обратными связями по возмущению и по отклонению:

x_i – основное управляющее воздействие *i*-того канала;

x_j – основное управляющее воздействие *j*-того канала;

X – совокупность остальных управляющих воздействий;

b_{ii} – коэффициент передачи i -того канала по основному управляющему воздействию;

b_{ij} – коэффициент паразитной перекрестной связи от j -того на i -тый канал управления;

$\mathbf{B}_i$ – i -тая строка матрицы $\mathbf{B}$ кроме элементов b_{ii} и b_{ij} , см. соотношения (4.2) и (4.3);

f_k – k -тое внешнее возмущение, действующее на ТО;

$\mathbf{F}$ – совокупность остальных внешних возмущений;

d_{ik} – коэффициент передачи возмущения f_k на i -тый канал управления;

$\mathbf{D}_i$ – i -тая строка матрицы $\mathbf{D}$ кроме элемента d_{ik} , см. соотношения (4.2) и (4.3);

k_j – коэффициент компенсирующей перекрестной связи от j -того на i -тый канал управления;

k_{fk} – коэффициент компенсирующей обратной связи по внешнему возмущению f_k ;

k_i – коэффициент отрицательной обратной связи по регулируемому параметру y_i .

(рис. 4.1), каким образом можно организовать компенсацию возмущений, действующих в i -том канале АСУТП, обеспечивающей управление ТО, модель которого представлена матричным уравнением (4.2). В соответствии с (4.2) уравнение i -того канала управления может быть представлено в виде

$$y_i = b_{i1}x_1 + b_{i2}x_2 + \dots + b_{ii}x_i + \dots + b_{in}x_n + d_{i1}f_1 + d_{i2}f_2 + \dots + d_{im}f_m; \quad i=1, 2, \dots, n. \quad (4.3)$$

Любой из коэффициентов b_{ij} ($j \neq i$, $b_{ij} \neq 0$) уравнения (4.3) отображает влияние других каналов управления на i -тый канал, а коэффициенты $d_{ik} \neq 0$ ($k=1, 2, \dots, m$) отображают влияние на этот канал внешних возмущений. Допустим, что на i -тый канал управления наибольшее нежелательное влияние оказывают управляющее воздействие x_j (через j -тый канал управления) и внешнее возмущение f_k (см. рис. 4.1). Действие возмущений на i -тый канал показано на рис. 4.1 сплошными линиями, так же как и действие основного управляющего воздействия x_i через звено прямой связи b_{ii} . Компенсацию возмущения от j -того канала проведем посредством формирования компенсирующей перекрестной связи на i -тый канал через звено с коэффициентом передачи k_j , а компенсирующее воздействие от возмущения f_k подадим через дополнительный канал обратной связи с коэффициентом передачи k_{fk} . Компенсирующие связи, также как канал обратной связи по отклонению с коэффициентом обратной связи k_i , показаны на рис. 4.1 штриховыми линиями.

После введения компенсирующих связей по x_j и f_k , показанных на рис 4.1, уравнение (4.3) преобразуется к виду:

$$y_i = b_{i1}x_1 + b_{i2}x_2 + \dots + (b_{ij} - b_{ii}k_j)x_j + \dots + b_{in}x_n + d_{i1}f_1 + d_{i2}f_2 + \dots + (d_{ik} - b_{ii}k_{fk})f_k + \dots + d_{im}f_m$$

$$; i=1, 2, \dots, n; \quad d_{ik} = d_{i1}, d_{i2}, \dots, d_{im}. \quad (4.4)$$

Чтобы добиться инвариантности i -того канала к возмущениям x_j и f_k , необходимо обеспечить равенство возмущающих и компенсирующих сигналов, которое выражается, согласно (4.4), соотношениями:

$$k_j = \frac{b_{ij}}{b_{ii}} \quad \text{и} \quad k_{fk} = \frac{d_{ik}}{b_{ii}}, \quad (4.4')$$

где: k_j – коэффициент передачи компенсирующей перекрестной связи от j -того на i -тый канал;

k_{fk} – коэффициент передачи компенсирующей обратной связи по k -тому внешнему возмущению, действующему в i -том канале связи.

Поскольку не было наложено никаких ограничений на номер k внешнего возмущения и на номер j соседнего канала управления (кроме $j \neq i$), то соотношения (4.4') пригодны для организации компенсации любых возмущений, воздействующих, согласно уравнению (4.3), на i -тый канал управления. Равным образом, поскольку не были наложены ограничения на номер i рассматриваемого канала управления, то соотношения (4.3) и (4.4) пригодны для организации инвариантного управления любым каналом в рамках АСУТП, управляющей любым ТО, описываемым системой линейных уравнений (4.2).

Общей трудностью при построении инвариантной системы управления является нелинейность как внутренних, так и внешних связей, приводящих к нежелательным отклонениям параметров ТО от заданных значений. Проектирование каналов управления, инвариантных к внешним возмущающим воздействиям, осложняется кроме того трудностями выделения и измерения воздействий такого рода. В качестве примера можно привести трудности учета влияния температуры окружающей среды в различных точках ТО, где она может быть различной и зависеть от различных факторов.

Иногда причины, вызывающие отклонения параметров от заданных значений, обусловлены случайными факторами, влияние которых трудно учесть заранее. Практически невозможно добиться полной инвариантности управляемого техпроцесса по отношению ко всем внешним возмущениям. Поэтому обычно ограничиваются компенсацией основных возмущений, а для нейтрализации остальных возмущающих воздействий вводят отрицательную обратную связь по отклонению управляемого (регулируемого) параметра от заданного значения.

На рис. 4.1 отрицательная обратная связь по управляемому параметру y_i реализована посредством звена с коэффициентом передачи k_i . Будем считать, что возмущающие воздействия x_i и f_k скомпенсированы, как показано на рис. 4.1, а остальные возмущения нейтрализуются отрицательной обратной связью. Внешние возмущения представлены на рис.4.1 в виде столбца **F**, действующего через строку **D_i** матрицы **D** в

соответствии с соотношением (4.3). Аналогично действует столбец $\mathbf{X}$ через строку $\mathbf{B}_i$ матрицы $\mathbf{B}$.

Действие отрицательной обратной связи по параметру y_i соответствует добавлению члена $-b_{ii}k_i y_i$ в правую часть уравнения (4.3). После перенесения этого члена в левую часть уравнения (4.3) получим с учетом введенных обозначений $\mathbf{B}_i$ и $\mathbf{D}_i$:

$$y_i(1 + b_{ii}k_i) = b_{ii}x_i + \mathbf{B}_i\mathbf{X} + \mathbf{D}_i\mathbf{F},$$

откуда получим

$$y_i = \frac{b_{ii}}{1 + b_{ii}k_i} x_i + \frac{\mathbf{B}_i\mathbf{X} + \mathbf{D}_i\mathbf{F}}{1 + b_{ii}k_i}. \quad (4.5)$$

Таким образом, благодаря отрицательной обратной связи по отклонению действие возмущений в любом канале управления может быть уменьшено в $1 + b_{ii}k_i$ раз. При этом нужное значение управляемого параметра y_i может быть достигнуто соответствующим увеличением управляющего воздействия x_i .

Подход к компенсации внутренних возмущающих воздействий, обусловленных прохождением управляющих сигналов по различным каналам управления, зачастую отличается от подхода к внешним возмущениям. Внешние возмущения являются большей частью неконтролируемыми и всегда нежелательными. Поэтому типичным средством их нейтрализации является отрицательная обратная связь по управляемому параметру. Кроме того, при решении задачи инвариантности к наиболее опасным возмущениям, организуется их измерение и компенсация, как показано на рис. 4.1. Показанная там же аналогичная компенсация соседнего управляющего воздействия x_j производится только тогда, когда воздействие соседнего канала на рассматриваемый канал x_i является нежелательным, паразитным. Если же это воздействие связано с общей стратегией управления ТО, то оно не компенсируется, а лишь согласуется с другими управляющими воздействиями, формируемыми АСУТП. Примером такого согласования действия каналов управления являются уравнения линейной интерполяции (3.6), см. §3.2. Таким образом, синтез системы управления, обеспечивающей стабилизацию параметров управляемого ТО, приводит к созданию многомерного регулятора, обеспечивающего необходимое качество поддержания заданных значений регулируемых параметров. Он нужен как для статического, так и астатического регулирования с учетом существующих линейных и нелинейных технологических связей.

4.3. Алгоритмы автоматической оптимизации

4.3.1. Статическая и динамическая оптимизация

Оптимизация технологических процессов является основным назначением АСУТП. Уже на стадии проектирования ставится задача

обеспечения возможностей оптимизации управляемого техпроцесса в различных направлениях: достижение максимума производительности, минимума себестоимости, требуемого качества продукции, экономии энергии и сырьевых ресурсов и пр. Зависимость между критерием эффективности и параметрами техпроцесса описывается *целевой функцией* (см. §1.2). Управление техпроцессом должно быть организовано таким образом, чтобы соотношение его параметров соответствовало заданному экстремуму целевой функции или допустимому граничному режиму, близкому к экстремальному. Если оптимальный режим может быть рассчитан заранее и поддерживаться в соответствии с рассчитанными значениями параметров, то такую оптимизацию называют *статической*, и ее задачи решают методами программного управления. Если же условия функционирования ТО непрерывно меняются и эти изменения трудно оценить заранее, то АСУТП должна вести автоматический поиск оптимального режима. Организацию такого поиска называют динамической оптимизацией.

Реальные техпроцессы зачастую подвергаются возмущениям, имеющим случайный характер и существенно изменяющим характеристики техпроцессов. В результате воздействия различных возмущений режим функционирования технологического оборудования может отклоняться от оптимального несмотря на поддержание управляющих параметров на заданном первоначально уровне. В таком случае возникает задача определения и задания новых значений управляющих параметров, обеспечивающих оптимальный режим в изменившихся условиях. Если эти значения не могут быть по тем или иным причинам определены заранее и заданы программно, то они определяются по ходу техпроцесса по мере отклонения последнего от оптимального режима. В описанных условиях такие отклонения возникают постоянно, так что необходимо вести поиск оптимального режима в течение всего времени техпроцесса. Иными словами, необходимо непрерывно совершать, одну за другой, попытки достижения оптимального режима. Такие попытки носят название *шагов оптимизации*. Алгоритмы управления, обеспечивающие непрерывный пошаговый поиск оптимального режима, называются *алгоритмами автоматической оптимизации*.

Методы оптимизации зависят от характера решаемых задач. Если целевая функция линейна и линейны уравнения ограничений, действующих в управляемом техпроцессе, то прибегают к методам *линейного программирования*. Если целевая функция нелинейна, то применяют преимущественно *градиентные методы*. Линейное программирование обычно представлено различными модификациями *симплексного метода*. Чаще всего оно применяется при решении задач экономического характера, являющихся для АСУТП типичными задачами статической оптимизации. Однако возможно применение

линейного программирования и для автоматической оптимизации в процессе функционирования ТО, если целевая функция и ограничения техпроцесса являются линейными или могут быть линеаризованы.

4.3.2. Симплексный метод линейного программирования

При поиске оптимального режима симплексным методом целевая функция должна быть представлена в виде

$$F_{\text{u}} = c_1 x_1 + c_2 x_2 + \dots + c_n x_n, \quad (4.6)$$

а уравнения ограничений – в виде

[illegible]

$$x_j \geq 0 \quad (j=1,2,\dots,n). \quad (4.8)$$

Принято отображать сомножители произведений, входящих в состав F_{II} , в виде векторов-строк

$$\mathbf{C}=[c_1, c_{21}, \dots, c_n] \text{ и } \mathbf{X}=[x_1, x_2, \dots, x_n],$$

а коэффициенты и свободные члены в системе уравнений (4.7) – в виде векторов – столбцов

$$\mathbf{a}_j = \begin{bmatrix} a_{1j} \\ a_{2j} \\ \vdots \\ a_{mj} \end{bmatrix}, j = 1, 2, \dots, n \quad \text{и} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

Векторы $\mathbf{a}_j$ называют векторами условий задачи, а вектор $\mathbf{b}$ называют вектором ограничений задачи. Компоненты векторов $\mathbf{a}_j$ в своей совокупности образуют матрицу.

$$A=[\mathbf{a}_1, \mathbf{a}_2, \dots \mathbf{a}_n],$$

которую называют матрицей условий задачи.

В векторной форме выражение (4.6) записывается в виде скалярного произведения векторов $\mathbf{c}$ и $\mathbf{x}$, а левая часть системы уравнений (4.7) записывается в виде произведения матрицы $\mathbf{A}$ на вектор $\mathbf{x}$. Теперь в очень краткой форме можно записать *основную задачу линейного программирования*:

найти максимум функции $F_{Ц} = \mathbf{c} \mathbf{x}$

$$\text{при условии } A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq 0; \quad (4.9)$$

ИЛИ

найти минимум функции $F_{II} = -cx$

при условии $-Ax \geq -b, \quad x \geq 0.$

(4.9')

Чтобы перейти к стандартной процедуре решения основной задачи линейного программирования симплексным методом, необходимо, чтобы среди векторов $\mathbf{a}_i$ было m единичных. Так, в системе уравнений

(4.7) векторы $\mathbf{a}_1 - \mathbf{a}_m$ – единичные, т.е. у каждого из этих векторов имеется только одна компонента a_{ij} ($i=1,2,\dots,m$), не равная нулю, и она равна единице. Если количество единичных векторов $\mathbf{a}_j$ меньше m , то в соответствующие уравнения ограничений необходимо ввести дополнительные переменные x_j , с помощью которых уравнения ограничений (4.9) будут записаны в канонической форме (4.7). В общем виде это производится методом искусственного базиса [9]. Однако в большинстве случаев недостающие переменные x_j можно просто прибавить в соответствующие уравнения ограничений, где их до того не хватало (см. пример 4.1.). Это возможно, если ограничения в условиях задачи представлены в виде неравенств.

Процедура симплексного метода основана на следующих положениях.

1. Совокупность точек $\mathbf{x}$ в n -мерном евклидовом пространстве E_n , ограниченная условиями (4.9), образует область допустимых решений рассматриваемой задачи. Каждую точку $\mathbf{x}=(x_1, x_2, \dots, x_n)$ из указанной области в экономических расчетах принято именовать *планом* (допустимым). Точка, в которой достигается либо максимум, либо минимум заданной целевой функции, называется *оптимальной точкой*, или *оптимальным планом*.
2. В пространстве E_n область допустимых решений образуется в результате пересечения полупространств $\mathbf{x} \geq 0$ (при $n=2$ – полуплоскостей) и $A\mathbf{x} \leq \mathbf{b}$ или $-A\mathbf{x} \geq -\mathbf{b}$, заданных условиями (4.9), и существует в виде *многогранника решений*. Оптимальная точка (оптимальный план) является одной из вершин многогранника решений.
3. Любые m линейно независимых векторов $\mathbf{a}_j$ ($j=1,2,\dots,n$, $m \leq n$), удовлетворяющих системе уравнений (4.7), порождают точку $\mathbf{x}=(b_1, b_2, \dots, b_m, 0, \dots, 0)$, являющуюся вершиной многогранника решений. Такую точку принято называть *угловой точкой* или *опорным планом*. Оптимальная точка является одной из угловых точек. В этом нетрудно убедиться на примере нахождения максимального значения функции двух переменных ($n=2$)

$$F_{\text{ц}} = c_1 x_1 + c_2 x_2 \quad (4.10)$$

при условиях

$$\begin{aligned} a_{i1}x_1 + a_{i2}x_2 &\leq b_i \quad (i=1,2), \\ x_j &\geq 0 \quad (j=1,2). \end{aligned} \quad (4.11)$$

Каждое из неравенств (4.11) геометрически определяет полуплоскость в соответствии с граничными прямыми

$$a_{i1}x_1 + a_{i2}x_2 = b_i \quad (i=1,2), \quad x_1 = 0 \text{ и } x_2 = 0.$$

Если система неравенств (4.11) совместна, то получим многоугольник (вырожденный многогранник, так как $n=2$) решений вроде приведенного на рис.4.2.

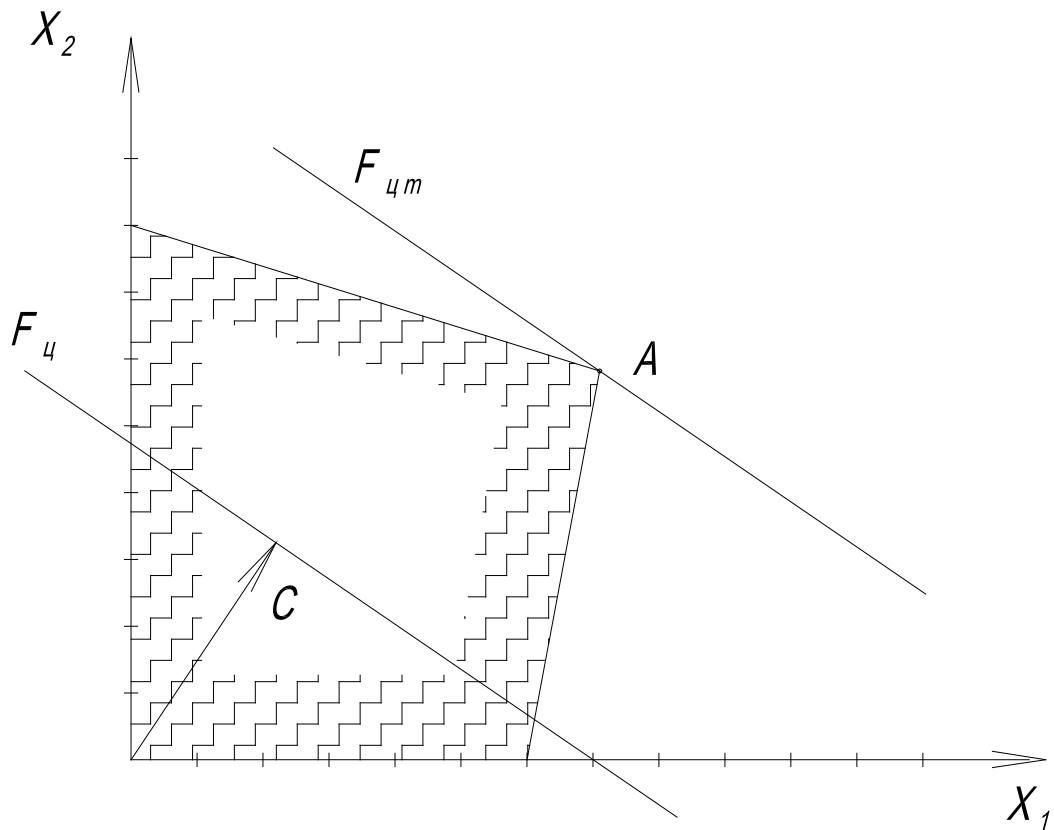


Рис. 4.2. Многоугольник решений

Область допустимых решений на рис.4.2 отмечена штриховкой ее границ. На графике рис.4.2 изображены также линии $F_{ц} = \text{const}$, соответствующие уравнению (4.10) при $c_1 > 0$ и $c_2 > 0$. Если перемещать линию $F_{ц}$ в направлении вектора $\mathbf{c} = (c_1, c_2)$, то значение $F_{ц}$ будет возрастать и достигнет максимального значения $F_{цm}$ в угловой точке A . Следовательно, точка A является оптимальной, а ее координаты определяют оптимальный план решения данной задачи.

Процедура симплексного метода заключается в том, что переходят от одной угловой точки (опорного плана) к другой угловой точке (к другому опорному плану) таким образом, что значение целевой функции обязательно увеличивается (при поиске максимума целевой функции) или обязательно уменьшается (при поиске минимума). При этом указываются признаки достижения максимума или, соответственно, минимума целевой функции.

Первые m единичных векторов $\mathbf{a}_j$ системы уравнений (4.7) составляют базис поиска оптимального плана. Поскольку оптимальная точка является одной из угловых точек, то искать ее следует путем замены одного из векторов базиса на один из векторов $\mathbf{a}_j$ ($j \geq m+1$), до того не входивших в состав базиса. После ввода в базис вектора $\mathbf{a}_j$ значение целевой функции можно рассчитать по формуле

$$F_{цj} = F_{ц0} - \lambda \Delta_j, \quad (4.12)$$

где $F_{ц0}$ – предыдущее значение целевой функции;

λ – коэффициент пропорциональности, значение которого определяется в зависимости от того, какой вектор выводится из базиса.

Параметр Δ_j определяет величину и знак приращения, которое получает целевая функция после ввода в базис нового вектора $\mathbf{a}_j$. Если $\Delta_j > 0$, то целевая функция убывает ($F_{цj} < F_{ц0}$), а при $\Delta_j < 0$ целевая функция увеличивается. Знак Δ_j является критерием полезности ввода нового вектора в базис. При поиске максимума имеет смысл вводить в базис только такой вектор, которому соответствует $\Delta_j < 0$, а при поиске минимума поиск оптимального режима (оптимального плана) продолжают лишь тогда, когда находится вектор, ввод которого в базис приводит к $\Delta_j > 0$ и соответственно – к уменьшению $F_{ц}$. Значение Δ_j рассчитывается по формуле

$$\Delta_j = \sum_{i=1}^m c_i a_{ij} - c_j \quad (j = 1, 2, \dots, n), \quad (4.13)$$

где c_i – первые m коэффициентов целевой функции (4.6);

a_{ij} – коэффициенты при x_j ($j = 1, 2, \dots, n$) системы уравнений (4.7).

При $j \leq m$, т.е. для m единичных векторов, входивших перед этим расчетом в состав базиса, согласно (4.13) получим $\Delta_j = 0$.

Рассмотрим более подробно поиск оптимального режима, когда последний соответствует максимуму целевой функции. Сначала перечислим признаки оптимальности режима (опорного плана).

1) Опорный план $\mathbf{x}^* = (x_1^*, x_2^*, \dots, x_m^*, 0, 0, \dots, 0)$ решения задачи (4.6)–(4.8) является оптимальным, если $\Delta_j \geq 0$ для любого j ($j = 1, 2, \dots, n$).

2) Если $\Delta_k < 0$ для некоторого $j = k$ и среди чисел a_{ik} нет положительных, то целевая функция (4.6) не ограничена.

3) Если $\Delta_k < 0$, но среди чисел a_{ik} есть положительные (не все $a_{ik} \leq 0$), то существует опорный план $\mathbf{x}'$ такой, что $F_{ц}(\mathbf{x}') > F_{ц}(\mathbf{x})$.

Далее составляем таблицу 4.1а. В табл. 4.1а первые m строк определяются исходными данными задачи, а показатели $(m+1)$ -й строки вычисляют.

Таблица 4.1а

i	Базис	c_0	$\mathbf{b}$	c_1	c_2	...	c_r	...	c_m	c_{m+1}	...	c_k	...	c_n
				$\mathbf{a}_1$	$\mathbf{a}_2$	...	$\mathbf{a}_r$	...	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	...	$\mathbf{a}_k$	...	$\mathbf{a}_n$

1	$\mathbf{a}_1$	c_1	b_1	1	0	...	0	...	0	a_{1m+1}	...	a_{1k}	...	a_{1n}
2	$\mathbf{a}_2$	c_2	b_2	0	1	...	0	...	0	a_{2m+1}	...	a_{2k}	...	a_{2n}
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
r	$\mathbf{a}_r$	c_r	b_r	0	0	...	1	...	0	a_{rm+1}	...	a_{rk}	...	a_{rn}
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
m	$\mathbf{a}_m$	c_m	b_m	0	0	...	0	...	1	a_{mm+1}	...	a_{mk}	...	a_{mn}

$m+1$ F_{y0} 0 0 ... 0 ... 0 Δ_{m+1} ... Δ_k ...
 Δ_n Исходные данные таблицы 4.1а соответствуют опорному плану $\mathbf{x}=(b_1, b_2, \dots, b_m, 0, \dots, 0)$.

Таблица 4.1б

i	Базис	$\mathbf{c}_b$	$\mathbf{b}$	c_1	c_2	...	c_r	...	c_m	c_{m+1}	...	c_k	...	c_n
				$\mathbf{a}_1$	$\mathbf{a}_2$	...	$\mathbf{a}_r$	...	$\mathbf{a}_m$	$\mathbf{a}_{m+1}$	...	$\mathbf{a}_k$	...	$\mathbf{a}_n$
1	$\mathbf{a}_1$	c_1	b'_1	1	0	...	a'_{1r}	...	0	a'_{1m+1}	...	0	...	a'_{1n}
2	$\mathbf{a}_2$	c_2	b'_2	0	1	...	a'_{2r}	...	0	a'_{2m+1}	...	0	...	a'_{2n}
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
r	$\mathbf{a}_k$	c_k	b'_r	0	0	...	a'_{rr}	...	0	a'_{rm+1}	...	1	...	a'_{rn}
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
m	$\mathbf{a}_m$	c_m	b'_m	0	0	...	a'_{mr}	...	1	a'_{mm+1}	...	0	...	a'_{mn}

$m+1$ F'_{y0} 0 0 ... Δ'_r ... 0 Δ'_{m+1} ... 0 ...
 Δ'_n

В $(m+1)$ -й строке таблицы 4.1а в столбце $\mathbf{b}$ записывают значение F_{y0} , рассчитанное по формуле (4.6), а в каждом столбце $\mathbf{a}_j$ ($j=1, 2, \dots, n$) записывают значение Δ_j , рассчитанное по формуле (4.13). Если $\Delta_j < 0$ для некоторых индексов j и для каждого такого j по крайней мере одно из чисел a_{ij} положительно, то можно составить новый опорный план, при котором значение целевой функции увеличится. Чтобы перейти к улучшенному опорному плану, нужно ввести в базис вектор $\mathbf{a}_k$, у которого $\Delta_k < 0$, и это Δ_k является наибольшим (по абсолютной величине) отрицательным числом среди всех Δ_j . Взамен из базиса исключается вектор $\mathbf{a}_r$. Номер r соответствует элементу a_{rk} вектора $\mathbf{a}_k$, а элемент a_{rk} находят по минимуму выражения b_i/a_{ik} для всех $a_{ik} > 0$. Если же все $a_{ik} \leq 0$ ($i=1, 2, \dots, m$), то целевая функция не ограничена сверху. Число a_{rk} называется разрешающим элементом. Столбец $\mathbf{a}_k$ и строку r , на пересечении которых находится разрешающий элемент, называют направляющими (или разрешающими). Улучшенный опорный план

находят методом Жордана-Гаусса. При этом положительные компоненты нового опорного плана находят по формулам:

$$b'_i = \begin{cases} b_i (b_{rk} \square / a_{rk}) \cdot a_{ik} & \text{при } i \neq r, \\ b_{rk} \square / a_{rk} & \text{при } i = r, \end{cases} \quad (4.14)$$

а коэффициенты разложения векторов $\mathbf{a}_j$ через векторы нового базиса, соответствующего новому опорному плану, - по формулам:

$$a'_{ij} = \begin{cases} a_{ij} (a_{rk} \square / a_{rk}) \cdot a_{ik} & \text{при } i \neq r, \\ a_{rk} \square / a_{rk} & \text{при } i = r. \end{cases} \quad (4.15)$$

После этого вместо таблицы 4.1а получится таблица 4.1б. Элементы (m+1)-й строки таблицы 4.1б находят по формулам:

$$F'_{yo} = F_{yo} - (b_r \square / a_{rk}) \Delta_k, \quad (4.16)$$

$$\Delta'_j = \Delta_j - (a_{rj} / a_{rk}) \Delta_k, \quad (4.17)$$

либо на основании их определения, см. формулы (4.6) и (4.13). Затем просматривают элементы (m+1)-й строки. Если все $\Delta'_j \geq 0$, то новый опорный план оптимален. Если же имеются $\Delta'_j < 0$, то поиск оптимального плана продолжается. Заметим, что формулы (4.14)–(4.17) можно интерпретировать для визуального поиска элементов расчёта *методом треугольника*, когда все расчёты (кроме случая $i=r$) ведутся по формуле

$$b = a_1 - a_{2/1} a_3,$$

где a_1 – это либо b_i , либо a_{ij} ;

$a_{2/1}$ – либо $b_{rk} \square / a_{rk}$, либо a_{rj} / a_{rk} ;

a_3 – либо a_{ik} , либо Δ_k .

Пример 4.1. Оптимизация прокатки путем линейного программирования симплексным методом.

Определить месячную программу прокатки стали двух марок на сортовом стане, обеспечивающую максимальную прибыль.

Дано:

затраты времени по каждому сортаменту:

$$t_1 = 0,04 \text{ ч/т}; \quad t_2 = 0,08 \text{ ч/т};$$

расход энергии при прокатке:

$$w_1 = 8 \text{ кВт} \cdot \text{ч/т}; \quad w_2 = 3 \text{ кВт} \cdot \text{ч/т};$$

месячный фонд технологического времени $t_m \leq 600 \text{ ч}$;

месячный лимит электроэнергии на текущий месяц $W_m \leq 100000 \text{ кВт} \cdot \text{ч}$.

Найти: такой вариант месячной программы прокатки, который обеспечит в текущем месяце, когда имеются ограничения по электроэнергии, максимум прибыли при условии, что прибыльность второго сортамента в 1,2 раза выше прибыльности первого сортамента стали:

$$F_y = x_1 + 1,2x_2.$$

В выражении целевой функции $F_{ц}$ прибыльность выражена в относительных единицах (о.е.). За единицу прибыли принята прибыль от 1 т проката первого сортамента.

Выпуск проката ограничивается неравенствами:

$$0,04x_1 + 0,08x_2 \leq 600$$

$$8x_1 + 3x_2 \leq 100000.$$

Вводим: x_3 – неиспользованное технологическое время;

x_4 – неиспользованная в пределах лимита электроэнергия;

Получим систему уравнений ограничений задачи, выраженную в канонической форме:

$$0,04x_1 + 0,08x_2 + x_3 = 600;$$

$$8x_1 + 3x_2 + x_4 = 100000.$$

Расчёты производим в соответствии с алгоритмом, представленным в таблицах 4.1а и 4.1б. Результаты расчётов сводим в таблицу 4.2.

Таблица 4.2

i	Базис	c_b	b	1	1,2	0	0
				a_1	a_2	a_3	a_4
1	a_3	0	600	0,04	0,08	1	0
2	a_4	0	10^5	8	3	0	1
Прибыль			0	-1	-1,2	0	0

1	a_2	1,2	7500	0,5	1	12,5	0
2	a_4	0	77500	6,5	0	-37,5	1
Прибыль			9000	-0,4	0	15	0

1	a_2	1,2	~1500	0	1	15,5	-0,08
2	a_1	1	~12000	1	0	-6	0,16
Прибыль			13800	0	0	12,6	0,052

Возможный максимум прибыли составляет 13800о.е. при выпуске проката марок: $x_1=12000$ т; $x_2=1500$ т.

4.4. Градиентные методы автоматической оптимизации

4.4.1. Поиск экстремума целевой функции

Если целевая функция нелинейна, то поиск оптимального режима работы ТО производится, чаще всего, *градиентными методами*, применение которых позволяет решить *практически любую задачу*

нелинейной оптимизации. Сущность градиентных методов заключается в том, что на каждом шаге оптимизации при поиске максимума или минимума целевой функции *приращения управляющих параметров выбираются пропорциональными частным производным целевой функции по этим параметрам.*

Это правило можно отобразить с помощью следующего рекуррентного соотношения:

$$x_{ij+1} = x_{ij} \mp k_j \cdot \left. \frac{dF_u}{dx_i} \right|_j, \quad i = 1, 2, \dots, n, \quad (4.18)$$

где x_{ij+1} – значение i -го параметра на $(j+1)$ -м шаге оптимизации;

x_{ij} – значение того же параметра на j -том шаге оптимизации;

k_j – коэффициент, определяющий величину шага оптимизационной процедуры, называемый также *шагом*;

$\frac{\partial F_u}{\partial x_i}$ – частная производная целевой функции F_u по i -му параметру

x_i , которая вычисляется на каждом шаге оптимизации;

n – количество управляющих параметров оптимизируемого процесса.

Частные производные $\frac{\partial F_u}{\partial x_i}$ являются проекциями градиента функции F_u в

$(n+1)$ -мерном пространстве, которое составляют параметры x_i и функция F_u . Примем во внимание, что градиент – это вектор, указывающий направление на максимум функции. По этой причине знак плюс в формуле (4.18) ставится, если оптимальный режим соответствует максимуму целевой функции, а знак минус, если оптимальный режим соответствует минимуму целевой функции.

В пределах шага оптимизационной процедуры, когда оптимизируемые параметры получают приращения

$$\Delta x_{ij} = \mp k_j \cdot \left. \frac{\partial F_u}{\partial x_i} \right|_j, \quad (4.18')$$

расчётные значения частных производных не изменяются. После выполнения шага оптимизации значение x_{ij} , согласно соотношению (4.18), увеличивается или уменьшается до значения x_{ij+1} , а это означает, что при изменении параметров техпроцесса в пределах $x_{ij} \div x_{ij+1}$, $i = 1, 2, \dots, n$, значения частных производных функции F_u по этим параметрам принимались неизменными. Следовательно, на каждом шаге оптимизации, в пределах шага, *целевая функция принимается линейной*, хотя в действительности она нелинейна. Если шаг чрезмерно велик (значение k_j чрезмерно велико), то в пределах шага *фактические* значения частных производных

функции $F_{ц}$ могут уменьшиться до нуля и даже изменить свой знак, а последнее будет означать, что система управления «проскочила» точку оптимального режима, точку, в которой все частные производные целевой функции равны нулю, т.е.

$$\frac{\partial F_{ц}}{\partial x_i} = 0, \quad i = 1, 2, \dots, n. \quad (4.19)$$

На следующем шаге при чрезмерно большом k_j система управления может опять пройти точку оптимального режима, но уже в обратном направлении. Начнется так называемое рысканье, из-за которого оптимальный режим не будет достигнут. Если значение k_j достаточно мало, то рысканья не будет, но при чрезмерно малом k_j продвижение к оптимальному режиму будет замедленным, из-за чего оптимальный режим также не будет достигнут: во-первых, поиск оптимального режима может затянуться на всё время реализации техпроцесса (даже при неизменности его параметров); во-вторых, за время поиска параметры техпроцесса могут изменяться так быстро, что система управления не будет успевать найти управляющие параметры, соответствующие оптимальному режиму. Дело в том, что при динамической оптимизации, производимой в течение работы ТО, значения частных производных целевой функции должны быть определены экспериментально, методом конечных разностей. Если контролируются n параметров, то на каждом шаге оптимизации, одновременно с выполнением технологической задачи, необходимо произвести *поочередно* малое приращение каждого из параметров x_i ($i = 1, 2, \dots, n$). Эти приращения должны быть достаточно малы, с тем чтобы не нарушать достигнутый уровень оптимизации режима работы, а с другой стороны – достаточно велики, чтобы их можно было измерить с заданной точностью с помощью существующей приборной базы. Кроме того, данные приращения должны оставаться неизменными в течение серии шагов оптимизации, чтобы уменьшить объём производимых при поиске оптимального режима вычислений.

Допустим, что установлены оптимизированные значения управляющих параметров x_{ij} ($i = 1, 2, \dots, n$) на j -том шаге оптимизации. Им соответствует исходное на текущем шаге оптимизации значение целевой функции $F_{ц0}$. Далее следует последовательно, одно за другим, выставить значения управляющих параметров $x_{ij} + \Delta x_i$, где Δx_i – запланированное пробное приращение i -того параметра, и вычислить n новых значений целевой функции $F_{цi}$ ($i = 1, 2, \dots, n$), соответствующих текущему шагу оптимизации. После этого можно будет вычислить все необходимые для реализации очередного шага оптимизации значения n частных производных целевой функции:

$$\frac{\partial F_{ц}}{\partial x_i} = \frac{F_{цi} - F_{ц0}}{\Delta x_i}, \quad i = 1, 2, \dots, n. \quad (4.20)$$

Новые оптимизированные значения x_{ij+1} на $(j+1)$ -м шаге оптимизации рассчитываются по формуле (4.18), после чего рассчитывается новое значение $F_{ц0}$.

Рассмотренная процедура измерений и расчетов может оказаться достаточно длительной при управлении реальным быстротекущим технологическим процессом, и тогда она, как показано выше, не гарантирует оптимальности режима работы из-за возможного рысканья или недостаточности быстрогодействия. Если же система управления достаточно быстродействующая, то итерационный процесс поиска оптимального режима должен продолжаться до тех пор, пока не получится

$$|F_{цi} - F_{ц0}| < \varepsilon, \quad i=1, 2, \dots, n, \quad (4.21)$$

где ε — малое положительное число, выбранное так, чтобы неравенство (4.21) обеспечивало достаточную близость к оптимальному режиму, т.е. к выполнению условий (4.19).

При поиске оптимального режима в точке экстремума целевой функции необходимо следить за тем, чтобы этот экстремум был *глобальным*, а не локальным, т.е. чтобы он соответствовал точке наибольшего значения целевой функции из всех значений, которых она достигает в точках локальных экстремумов. Эта проблема решается на этапе выбора начальной точки задания рабочего режима, которая должна находиться в такой окрестности глобального оптимума, что дальнейший поиск оптимального режима методом градиента автоматически приводил бы именно к глобальному оптимуму. Выбор такой начальной точки, т.е. совокупности управляющих параметров, задающих исходный рабочий режим, предшествующий автоматическому поиску оптимального режима, зависит в значительной степени от опыта и интуиции проектировщиков и наладчиков системы оптимизации.

Далее необходимо обеспечить максимальное быстродействие системы оптимизации без чрезмерного рысканья в окрестности точки оптимального режима. При этом следует учитывать, что наибольшее время в течение шага оптимизации затрачивается на сбор экспериментальных данных для расчета проекций градиента целевой функции в соответствии с соотношением (4.20). Чтобы наиболее эффективно использовать полученный экспериментальный материал, следует правильно выбрать значение шагового коэффициента k_j . Начальное его значение выбирается исходя из опыта проектирования и наладки аналогичных систем автоматической оптимизации. Наиболее простым способом уточнения значений k_j является *метод удвоения длины шага*. Он заключается в том, что если после выполнения шага оптимизации с начальным значением k_j окажется $F_{ц} < F_{ц0}$ (при минимизации $F_{ц}$) или $F_{ц} > F_{ц0}$ (при максимизации $F_{ц}$), то производится еще один шаг *без сбора экспериментальных данных*, с прежними

значениями $\frac{dF_u}{dx_i}$ и k_j , что эквивалентно удвоению k_j . Если после этого значение F_u снова уменьшится (при минимизации) или увеличится (при максимизации), то приступают к обычному шагу оптимизации, т. е. начинают сбор экспериментальных данных но с новым, удвоенным значением k_j . В противном случае попытка удвоения k_j считается неудачной и очередной шаг оптимизации производится с прежним значением k_j . Если же на очередном шаге оптимизации сразу окажется $F_u > F_{uo}$ (при минимизации) или $F_u < F_{uo}$ при максимизации, то следующий шаг производится обычным порядком, но с уменьшенным вдвое значением k_j . Тем самым предотвращается рысканье.

Значения k_j могут быть уточнены и расчётным путём, например – методом одномерной минимизации (или максимизации). Тогда они на каждом шаге оптимизации определяются исходя из условия минимальности или, соответственно – максимальной функции

$$\varphi_u(k_j) = F_u(x_{ij} \pm k_j \cdot \frac{\partial F_u}{\partial x_i}). \quad (4.22)$$

Здесь функция φ_u трактуется как функция одного лишь параметра k_j . Так получается, когда в формулу для F_u подставляют, вместо каждого x_i , его конкретное значение, полученное на j -том шаге оптимизации, в виде

$$x_{ij} + k_j \frac{\partial F_u}{\partial x_{ij}},$$

причем значение k_j считается неизвестным. Далее, исходя из уравнения

$$d\varphi_u/dk_j = 0,$$

определяется значение k_j , соответствующее минимуму (при минимизации) или максимуму (при максимизации) функции φ_u в заданной точке x_{ij} . Если оптимизация проводится в соответствии с соотношением (4.18), а значение k_j при этом определяется исходя из минимума φ_u , то такой метод оптимизации называют *методом наискорейшего спуска*. Имеется в виду скорейший спуск к точке минимума целевой функции, но этот термин применяется и при поиске оптимального режима в точке максимума целевой функции.

Уточнение значений k_j может производиться либо экспериментально, скажем – методом удвоения длины шага, либо аналитически, с использованием математической модели техпроцесса. В первом случае процедура удвоения длины шага требует дополнительного экспериментального определения значения F_u после удвоения шага, что ведет к увеличению времени уточнения k_j , но дополнительные расчёты, которые также требуют времени, при этом не нужны. Использование модели техпроцесса, необходимое для уточнения k_j аналитическим путём, скажем, методом одномерной оптимизации, позволяет избежать

дополнительного сбора экспериментальных данных, но объём расчётов при этом многократно возрастает.

4.4.2. Автоматическая оптимизация электрохимической обработки

В качестве примера применения метода градиента рассмотрим оптимизацию электрохимической обработки (ЭХО).

При электрохимической обработке металла происходит анодное растворение обрабатываемого участка в среде электролита. В процессе анодного растворения электролит загрязняется, и скорость внедрения обрабатывающего электрода в изделие постепенно уменьшается. Чтобы ее повысить, обрабатывающий электрод отодвигают от изделия и в это время производят промывку межэлектродного промежутка. Во время промывки анодное растворение не производится. После промывки производится контроль (измерение глубины внедрения обрабатывающего электрода в изделие), а затем, если заданная глубина ещё не достигнута, вновь приступают к анодному растворению металла. Таким образом, цикл ЭХО состоит из трёх стадий: времени анодного растворения t_a , времени промывки $t_{\text{п}}$ и времени контроля t_k , причём рабочим операциям всегда предшествует контроль.

Если за время t_a обрабатывающий электрод внедряется в изделие на глубину h , то средняя скорость внедрения электрода в изделие за цикл обработки составит

$$V_{cp} = \frac{h}{t_a + t_{\text{п}} + t_k} . \quad (4.23)$$

Величину t_k делают минимально возможной, а величины t_a и $t_{\text{п}}$ варьируют так, чтобы скорость V_{cp} внедрения обрабатывающего электрода в изделие стала как можно больше. Чем больше V_{cp} , тем выше производительность ЭХО. Поэтому, если настраивать ЭХО на максимум производительности, то в качестве целевой функции системы автоматической оптимизации (САО) целесообразно принять выражение (4.23), причем САО должна вести поиск максимума V_{cp} . Этот поиск заключается в том, что САО периодически так изменяет параметры t_a и $t_{\text{п}}$, что значение V_{cp} увеличивается. В идеале такая САО должна поддерживать значение V_{cp} на максимально возможном уровне при всех возможных возмущениях. Величина и знак изменений управляющих параметров t_a и $t_{\text{п}}$ будем определять методом градиента, описанным выше. Каждый цикл расчёта приращений управляющих параметров и задания системе их новых значений будем называть *шагом оптимизации*. Поскольку в процессе реализации ЭХО качество электролита непрерывно ухудшается и при этом, как мы увидим ниже, смещается точка оптимального режима, то оптимизация ЭХО должна проводиться непрерывно в течение всего времени ЭХО.

Опишем методику автоматической оптимизации ЭХО, выразив параметры ЭХО в относительных единицах:

$$v_{cp} = \frac{V_{cp}}{V_{cpm}},$$

где V_{cpm} – максимально возможное значение скорости внедрения электрода при самом благоприятном сочетании параметров;

$\tau_a = t_a/T_a$ – относительное время анодного растворения;

$\tau_{\pi} = t_{\pi}/T_a$ – относительное время промывки;

$\tau_k = t_k/T_a$ – относительное время контроля;

T_a – постоянная времени загрязнения электролита на стадии анодного растворения.

При оптимизации по методу градиента согласно соотношению (4.18) управляющие параметры τ_a и τ_{π} должны изменяться на каждом шаге оптимизации по следующим правилам:

$$\tau_{ai+1} = \tau_{ai} + k_i \left(\frac{\partial v_{cp}}{\partial \tau_a} \right)_i; \quad (4.24)$$

$$\tau_{\pi i+1} = \tau_{\pi i} + k_i \left(\frac{\partial v_{cp}}{\partial \tau_{\pi}} \right)_i. \quad (4.24')$$

Здесь i – это индекс, которым помечены значения параметров и их производных на i -ом шаге оптимизации,

k_i – коэффициент, определяющий величину шага.

Очевидно, что чем ближе мы к максимуму v_{cp} , тем меньше должны быть изменения значений τ_a и τ_{π} , чтобы избежать рысканья САО в окрестности оптимума. Поэтому значения k_i выбирают обратно пропорционально величине v_{cp} . А величины производных от v_{cp} определяют при помощи факторного анализа (см. § 3.4.2). С этой целью на каждом шаге оптимизации САО производит полнофакторный эксперимент, при котором она задаёт управляющим параметрам τ_a и τ_{π} малые приращения $\Delta\tau_a$ и $\Delta\tau_{\pi}$, которые ввиду своей незначительности существенно не влияют на ход техпроцесса. Величина этих пробных приращений всегда одна и та же. С каждым приращением производится один цикл ЭХО. Затем по формуле (4.23) САО производит расчет значений v_{cp} , используя данные о ходе ЭХО, полученные по каналам текущей информации. Результаты расчётов на каждом шаге оптимизации, состоящем из четырёх циклов ЭХО, сводятся в таблицу 4.3.

Таблица 4.3.

Номер цикла ЭХО	Время активной стадии	Время промывки	Средняя скорость внедрения в изделие
1	τ_{ai}	$\tau_{\pi i}$	V_{cpi1}

2	τ_{ai}	$\tau_{\pi i} + \Delta\tau_{\pi}$	v_{cpi2}
3	$\tau_{ai} + \Delta\tau_a$	$\tau_{\pi i} + \Delta\tau_{\pi}$	v_{cpi3}
4	$\tau_{ai} + \Delta\tau_a$	$\tau_{\pi i}$	v_{cpi4}

Как видно из табл. 4.3., по результатам эксперимента можно вычислить по два приближенных значения интересующих нас производных целевой функции v_{cp} . За истинные значения производных САО принимает средние значения, вычисленные по результатам полнофакторного эксперимента:

$$\left(\frac{\partial v_{cp}}{\partial \tau_a}\right)_i = \frac{1}{2} \left(\frac{v_{cpi3} - v_{cpi2}}{\Delta\tau_a} + \frac{v_{cpi4} - v_{cpi1}}{\Delta\tau_a} \right);$$

$$\left(\frac{\partial v_{cp}}{\partial \tau_{\pi}}\right)_i = \frac{1}{2} \left(\frac{v_{cpi2} - v_{cpi1}}{\Delta\tau_{\pi}} + \frac{v_{cpi3} - v_{cpi4}}{\Delta\tau_{\pi}} \right).$$

Если подставить полученные значения производных в выражения (4.24), то после соответствующих преобразований получим следующий алгоритм управления, по которому должна действовать САО ЭХО на каждом шаге оптимизации:

$$\tau_{a(i+1)} = \tau_{ai} + k_1 \frac{v_{cpi3} + v_{cpi4} - v_{cpi1} - v_{cpi2}}{v_{cpi}}; \quad (4.25)$$

$$\tau_{\pi(i+1)} = \tau_{\pi i} + k_1 \frac{v_{cpi3} + v_{cpi2} - v_{cpi1} - v_{cpi4}}{v_{cpi}} \cdot \frac{\Delta\tau_a}{\Delta\tau_{\pi}}, \quad (4.26)$$

где k_1 – постоянный коэффициент, учитывающий значение $\Delta\tau_a$ и другие неизменные параметры САО;

v_{cpi} – среднее значение величины v_{cp} в пределах шага оптимизации:

$$v_{cpi} = \frac{v_{cpi1} + v_{cpi2} + v_{cpi3} + v_{cpi4}}{4}. \quad (4.27)$$

При работе по алгоритму, описанному в таблице 4.3, и выражениями (4.23)-(4.27), САО ЭХО будет периодически вносить поправки в соотношение τ_a и τ_{π} , удерживая техпроцесс на максимуме производительности. Полнофакторные эксперименты, необходимые для удержания техпроцесса на максимуме производительности, производятся непрерывно, один за другим, а истинное значение производительности определяется формулой (4.27).

Дальнейшее изучение возможностей оптимизации ЭХО будем вести с помощью математической модели ЭХО, решённой относительно скорости v_{cp} , являющейся целевой функцией. Исходя из представления об изменении концентрации продуктов электрохимических реакций на каждой стадии ЭХО по экспоненциальному закону, придем к следующему выражению для средней скорости внедрения электролита в изделие:

$$v_{cp} = \frac{(1-e^{-\tau_a}) \cdot (1-e^{-\tau_{\pi}/k_T})}{(\tau_a + \tau_{\pi} + \tau_{\kappa}) \cdot (1-e^{-\tau_{\pi}/k_T} \cdot e^{-\tau_a})}, \quad (4.28)$$

где $k_T = T_{\pi}/T_a$ характеризует степень загрязнения электролита: чем больше значение k_T , тем больше загрязнение;

T_{π} – постоянная времени промывки электролита.

При расчёте методом градиента параметров оптимального режима значения целевой функции v_{cp} будем вычислять по выражению математической модели (4.28) в соответствии с программой полнофакторного эксперимента, приведённой в таблице 4.3. Затем по формулам (4.25) и (4.26) будем определять значения оптимизированных параметров τ_a и τ_{π} на каждом шаге оптимизации, а по формуле (4.27) – оценивать достигаемое увеличение производительности. Поиск оптимального решения будем вести до тех пор, пока достигаемое на текущем шаге оптимизации увеличение значения v_{cp} не станет меньше заданного ε в соответствии с неравенством

$$\frac{v_{cpi+1} - v_{cpi}}{v_{cpi}} < \varepsilon. \quad (4.29)$$

С помощью увеличения значения k_T можно моделировать ухудшение качества электролита в процессе реализации ЭХО и оценить влияние этого ухудшения на производительность и параметры оптимального режима ЭХО.

Пример 4.2. Определить параметры оптимального режима ЭХО методом градиента и оценить влияние ухудшения качества электролита.

Исходные данные: $\tau_{a0}=0,18$; $\tau_{\pi0}=0,88$; $\Delta\tau_a=0,036$; $\Delta\tau_{\pi}=0,072$; $\tau_{\kappa}=0,3$; $k_1=1$; $\varepsilon_{don}=0,005$.

Определить параметры оптимального режима при $k_{T1}=1,8$ и $k_{T2}=5,8$.

Для проведения расчётов на компьютере рекомендуется составить расчётную программу. Величину ε необходимо определять исходя из неравенства (4.29) и сравнивать с допустимым значением ε_{don} .

Результаты расчётов приведены в таблицах 4.4. и 4.5.

$k_{T1}=1,8$

Таблица 4.4

ε	—	0,38	0,030	0,015	0,0091	0,0060	0,0042
τ_a	0,18	0,388	0,437	0,472	0,499	0,521	0,539
τ_{π}	0,88	0,851	0,841	0,835	0,832	0,830	0,830
v_{cp}	0,0974	0,1374	0,1415	0,1436	0,1450	0,1458	0,1464

$k_{T2}=5,8$

Таблица 4.5

ε	-0,53	0,011	0,0087	0,0069	0,0055	0,0045
τ_a	0,554	0,538	0,524	0,512	0,502	0,493
τ_{π}	0,831	0,855	0,877	0,896	0,914	0,930

v_{cp}	0,0676	0,0684	0,0690	0,0694	0,0698	0,0702
----------	--------	--------	--------	--------	--------	--------

Параметры оптимального режима:

1). $k_T=1,8$; $\tau_a=0,539$; $\tau_n=0,830$; $v_{cp}=0,1464$;

2). $k_T=5,8$; $\tau_a=0,493$; $\tau_n=0,930$; $v_{cp}=0,0702$.

Прочерк в таблице 4.4 означает, что начальное значение v_{cp} (до начала процесса оптимизации) не с чем сравнить. Переход к $k_{T2}=5,8$, т.е. ухудшение качества электролита, ведёт к резкому уменьшению v_{cp} . В действительности же таких резких скачков не случается, так как качество электролита ухудшается постепенно.

4.4.3. Поиск предельно допустимого оптимального режима

Зачастую нужный экстремум целевой функции не может быть достигнут из-за имеющихся технологических ограничений. Тогда оптимальный режим реализуется в одной из граничных точек области допустимых значений параметров техпроцесса. Если ограничения будут заданы в виде линейных неравенств, то напрашивается применение приёма *линеаризации целевой функции*, присущей методу градиента, в окрестности оптимального режима для сведения решения исходной задачи к линейному программированию.

Целевая функция $F_{ц}$ после линеаризации принимает вид:

$$F_{цл} = \frac{\partial F_{ц}}{\partial x_1} x_1 + \frac{\partial F_{ц}}{\partial x_2} x_2 + \dots + \frac{\partial F_{ц}}{\partial x_n} x_n. \quad (4.30)$$

Исходя из выражения (4.30) и уравнений линейных ограничений, находят точку оптимального режима $x_{отпл}$ для линеаризованных условий задачи путём линейного программирования, например, симплексным методом, описанным в § 4.3. Полученная точка $x_{отпл}$ не будет, скорее всего, точкой искомого оптимального режима, так как исходная целевая функция $F_{ц}$ является в действительности нелинейной. Истинная оптимальная точка будет находиться у вогнутых функций ближе к исходной рабочей точке $x_o=(x_{1o}, x_{2o}, \dots, x_{no})$, а у выпуклых – дальше от неё.

Поэтому новую рабочую точку находят исходя из соотношения

$$x = x_o + \lambda(x_{отпл} - x_o), \quad (4.31)$$

где λ – коэффициент пропорциональности, определяющий величину шага вычислений.

Величину λ (обычно $0 \leq \lambda \leq 1$) можно определить с помощью процедуры удвоения шага, но лучше начинать с половинного шага, т. е. сначала положить $\lambda=1/2$, а далее ополовинивать или удваивать шаг до тех пор, пока значение $F_{ц}$ будет изменяться в нужную сторону. Можно также пользоваться приёмами одномерной минимизации (максимизации). После определения значения λ вычисляют координаты точки x и соответствующее им значение $F_{ц}$, которое на первом шаге оптимизации

будет, скорее всего, недостаточно близким к оптимальному значению. Если это так, то придётся опять определить проекции градиента от $F_{ц}$, составить, согласно (4.30), новое значение $F_{цл}$ и приступить к новому шагу оптимизации, начиная с линейного программирования.

Более универсален *метод штрафных функций*, при применении которого не требуется соблюдение линейности уравнений ограничений. Применение этого метода рассмотрим на примере определения максимального значения вогнутой целевой функции $F_{ц}(\mathbf{x})$, $\mathbf{x}=(x_1, x_2, \dots, x_n)$ при условиях

$$g_i(\mathbf{x}) \leq b_i \quad (i=1, 2, \dots, m), \quad x_j \geq 0 \quad (j=1, 2, \dots, n),$$

где $g_i(\mathbf{x})$ – линейные или нелинейные выпуклые функции.

Для решения данной задачи формируют *штрафную функцию*, которая чаще всего имеет вид

$$H(x) = \sum_{i=1}^m a_i(x) g_i(x), \quad (4.32)$$

$$a_i(x) = \begin{cases} 0, & \text{если } b_i - g_i(x) \geq 0, \\ a_i & \text{если } b_i - g_i(x) < 0 \end{cases}$$

а $a_i > 0$ – весовые коэффициенты.

Далее, вместо непосредственного решения заданной задачи, находят максимальное значение функции

$$F_{цл} = F_{ц}(x) - H(x)$$

с помощью рекуррентного соотношения

$$x_{j(k+1)} = x_{jk} + \left(\frac{\partial F_{ц}}{\partial x_j} - \sum_{i=1}^m a_i \frac{\partial g_i(x)}{\partial x_j} \right)_{k_k}, \quad (4.33)$$

причём фигурирующие в соотношении (4.33) частные производные относятся к k -тому шагу оптимизации.

Из (4.33) следует, что если значение x_{jk} находится в пределах допустимых решений исходной задачи, то второе слагаемое в скобках равно нулю ($a_i=0$) и процесс решения идёт обычным методом градиента, ср. с соотношением (4.18). Если же данная точка не принадлежит области допустимых решений, то за счёт второго слагаемого в скобках на последующих итерациях достигается возвращение в область допустимых решений.

Заметим, что динамическая оптимизация в процессе функционирования ТО проводится в условиях задания необходимых ограничений режима работы ТО независимо от ведущегося поиска оптимального режима и с приоритетом над ним. В результате, система управления ТО не выполняет указаний по оптимизации режима работы, если эти указания выходят за рамки запрограммированных ограничений. В качестве примера можно привести токоограничение и ограничение скорости вращения двигателя в электроприводах, запрограммированные независимо от поиска оптимального режима. В этих условиях нет

необходимости усложнять алгоритм поиска режима введением ограничивающих условий. Сложные алгоритмы автоматической оптимизации типа метода штрафных функций используются лишь тогда, когда более простые способы введения ограничений режима работы ТО неэффективны. С другой стороны, динамическая оптимизация быстротекущих процессов, в отличие от статической, не прекращается даже в случае достижения точки оптимального режима ввиду постоянных, непрерывно возникающих отклонений от него. Оптимизация техпроцесса в течение его реализации ведется поэтому непрерывно, с высокой периодичностью.

4.5. Применение методов нечеткой логики в АСУТП

4.5.1. Понятия и операции нечеткой логики

Наиболее сложным и ответственным этапом в процессе управления является принятие решений. Моделирование процессов принятия решений требует использования аппарата понятий с нечеткими границами и высказываний с многозначной шкалой истинности. Построение моделей принятия решений для задач, имеющих нечеткое словесное описание, базируется на понятиях *нечеткого множества* и *лингвистической переменной*. Эти понятия употребляются в случаях, когда описываемый объект может вполне определенно принадлежать к интересующему нас классу, может и не принадлежать к нему, возможны и промежуточные градации принадлежности, которые можно обозначить числами, принадлежащими промежутку от 1 до 0. После этого появляется возможность перейти от словесных описаний элементов задачи принятия решения к числовым.

Нечетким множеством A (Alternative) на универсальном множестве U (Universum), состоящим из элементов u , называется совокупность пар

$$A = \{ \mu_A(u), u \}, \quad (4.34)$$

где $\mu_A(u)$ - отображение множества U в единичный отрезок $[0,1]$, называемое *функцией принадлежности* нечеткого множества A . Иными словами, нечеткое множество A состоит из совокупности пар чисел, когда каждому числу из множества U ставится в соответствие число μ_A , причем имеет место

$$0 < \mu_A \leq 1. \quad (4.35)$$

Значение функции принадлежности $\mu_A(u)$ для любого элемента u будем называть *степенью принадлежности*. Нечеткое множество является количественной характеристикой нечеткого понятия, а степень принадлежности $\mu_A(u)$ есть субъективная оценка лицом, принимающим решение, того, насколько элемент u соответствует понятию, формализуемому нечетким множеством A . Можно также полагать, что степень принадлежности в какой-то мере соответствует вероятности

того, что элемент u будет отнесен к множеству A . Множество значений u , для которых имеет место $\mu_A > 0$ согласно соотношению (4.35), называется *носителем* S_A (*Set A*) нечеткого множества A . При графическом отображении нечеткого множества значение носителя нечеткого множества откладывают по оси абсцисс, а степень принадлежности – по оси ординат, формируя тем самым кривую функции принадлежности. Пример отображения посредством нечеткого множества нечеткого понятия «скорость движения V около 130 км/ч» приведен на рис. 4.3. Зависимость $\mu(V)$, приведенная на рис. 4.3, свидетельствует о том, что эксперт (лицо, принимающее решение) полагает возможным считать в заданных конкретных условиях значения $V = 100-160$ км/ч, соответствующими понятию «скорость движения V около 130 км/ч», но значения, более близкие к $V = 130$ км/ч, он считает более соответствующими данному понятию. Это мнение отображено на графике $\mu(V)$ значениями μ , более близкими к единице. Вне указанных пределов лежат значения V , не охваченные данным понятием, что формально отображается равенством $\mu(V) = 0$. Рассматриваемая оценка скорости V может не иметь непосредственного отношения к фактическому распределению скоростей интересующего нас объекта.

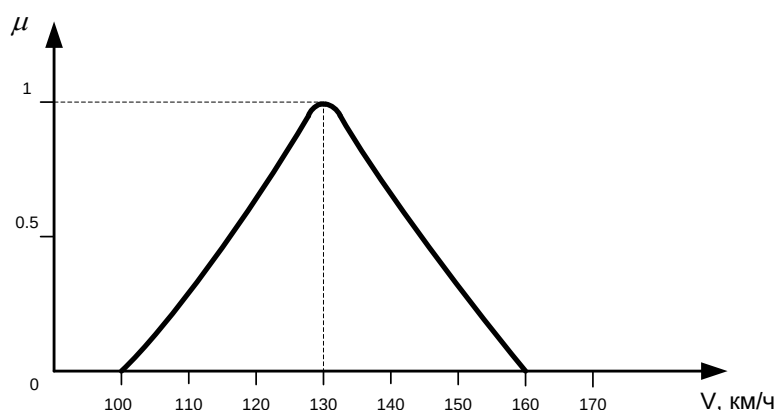


Рис. 4.3. Функция принадлежности нечеткого понятия «скорость движения V около 130 км/час»

Если возможны различные варианты нечетких множеств в пределах отображаемого нечеткого понятия, то такие варианты образуют *нечеткую переменную*. Так, нечеткая переменная может быть сформирована из различных оценок скорости движения, аналогичных представленной на рис. 4.3.

Лингвистическая переменная является числовой характеристикой сложного понятия, определенного на множестве U и состоящего из ряда более простых понятий. Отображения последних называются *термами* и представляют собой нечеткие переменные, областью определения каждой из которых также является множество U . Так, если мы хотим оценить величину скорости движения с помощью понятий «малая»,

средняя», «большая», то можем сделать это с помощью лингвистической переменной СКОРОСТЬ, содержащей три терма: МАЛАЯ, СРЕДНЯЯ и БОЛЬШАЯ. Эти термы должны перекрывать всю область определения скорости V , которую мы определим в процентах (0:-100)% от максимального значения скорости в интересующем нас диапазоне скоростей, хотя действительное значение V может быть и больше. Будем считать, что «малая» скорость не может превышать 50% от максимальной, а «большая» скорость не может быть менее тех же 50%. Наиболее подходящим значением для «средней» скорости будем считать те же 50% от максимальной. В таком случае лингвистическая переменная СКОРОСТЬ может быть отображена, например, графиком, представленным на рис.4.4.

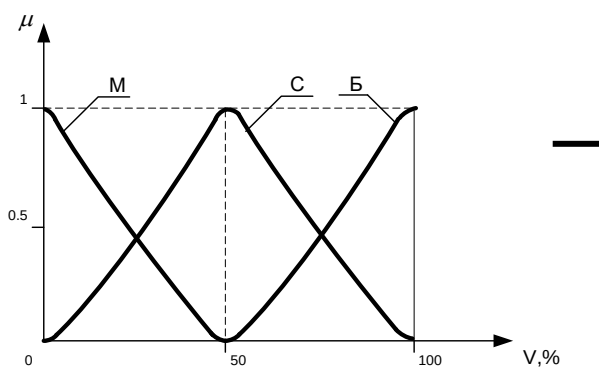


Рис. 4.4. Функции принадлежности лингвистической переменной СКОРОСТЬ: М – МАЛАЯ; С – СРЕДНЯЯ; Б – БОЛЬШАЯ

Здесь имеются в виду абсолютные значения скорости, о которых можно говорить, когда направление движения не имеет значения. Примером такого движения может быть полет самолета при неизменной высоте полета, неизменном атмосферном давлении и отсутствии ветра. В рассматриваемом случае значения $V < 0$ невозможны, но значения $V > 100\%$ вполне возможны, поскольку соответствуют понятию БОЛЬШАЯ, что отображается значением $\mu = 1$ при $V > 100\%$.

Нечеткие множества, соответствующие числовым значениям лингвистической переменной, называют *нечеткими числами*. Операции с нечеткими числами требуют введения понятия *нечеткого отношения*. Если имеются универсальные множества U и V , состоящие из элементов u и v , то нечетким отношением R (*Relation*) на множестве $U \times V$ называется совокупность пар

$$R = \{ \mu_R(u, v) / (u, v) \}, \quad (4.36)$$

Где $\mu_R(u, v)$ – функция принадлежности нечеткого отношения, отображающая множество $U \times V$ в единичный отрезок $[0, 1]$. Нечеткое отношение – это нечеткое множество с векторной базовой переменной. Нечеткие отношения дают численную характеристику таким высказываниям, как « X много больше Y » или « A значительно хуже B » и

т.п. Истинность высказываний такого рода определяется не значениями u и v в отдельности, а их соотношением. Соответственно и функция принадлежности $\mu_R(u, v)$ определяется совокупностью значений пары u, v .

Арифметические операции над нечеткими числами производят в соответствии с *принципом обобщения*, который состоит в следующем. Пусть имеются два нечетких числа, A и B , определенных на некотором интервале действительной оси. Пусть этим числам соответствуют носители: S_A , заданный на интервале (a_1, a_2) , и S_B , заданный на интервале (b_1, b_2) .

Определим нечеткое число

$$D = A \bullet B, \quad (4.37)$$

где $\bullet$ знак какой-либо из четырех арифметических операций: $+$, $-$, $\times$ или $/$, через функцию принадлежности

$$\mu_D(x) = \sup_{a \bullet b = x} \min\{\mu_A(a), \mu_B(b)\}, \quad (4.38)$$

где a и b – числа, принадлежащие соответственно носителям S_A и S_B . Значение x определяется путем совершения заданной арифметической операции над числами a и b , а $\mu_A(a)$ и $\mu_B(b)$ – это степени принадлежности чисел a и b в соответствии с функциями принадлежности нечетных множеств A и B .

Очевидно, что путем подбора значений a и b можно получить одно и то же значение x для многих пар (a, b) , которым будут соответствовать различные пары значений $\mu_A(a)$ и $\mu_B(b)$. Согласно соотношению (4.38) в каждой такой паре нужно выбрать меньшее ($\min$) значение, а затем из всей совокупности полученных значений μ отобрать наибольшее ($\sup$) значение, которое и будет искомым $\mu_D(x)$. Поскольку при определении степени принадлежности $\mu_D(x)$ результата арифметической операции мы выбираем в каждой паре исходных значений μ меньшее значение, а в качестве окончательного значения μ_D признаем наибольшее из выбранных меньших значений, то данный принцип обобщения называется *максиминным*, т. е. ориентированным на максимальный минимум значения μ .

В качестве обоснования принципа обобщения обратим внимание на то обстоятельство, что основой арифметических операций являются логические операции И (совпадения), см., например, формулы (5.14-5.16). Следовательно, если трактовать степень принадлежности μ_A как вероятность отнесения числа a к множеству A , а μ_B – как вероятность отнесения числа b к множеству B , то вероятность отнесения к множеству D результата произведенной над числами a и b арифметической операции будет близкой к произведению $\mu_A \mu_B$ при условии, что процессы отнесения a к A и b к B независимы. Поскольку обычно неизвестны ни распределения вероятностей на S_A и S_B , ни вид зависимости процессов отнесения аргументов a и b к нечетким числам A

и B , то приходится ограничиться максиминным соотношением (4.38). Его можно рассматривать как верхнюю оценку для $\mu_D(x)$, которая тем точнее, чем больше различие значений $\mu_A(a)$ и $\mu_B(b)$ и чем полнее перебор пар a и b , таких что $x = a \bullet b$.

Пример 4.3. Выполнить сложение нечетких чисел A и B :

$$A = (0,1/5; 0,8/6; 0,4/7);$$

$$B = (0,2/4; 0,9/5; 0,3/6).$$

В знаменателе каждой реализации нечеткого числа стоит значение его носителя, а в числителе – степень принадлежности. Заданная арифметическая операция выполняется относительно всех возможных пар носителей. Минимальное значение суммы $A + B$ равно 9, из исходных степеней принадлежности 0,1 и 0,2 выбираем меньшее. Следующее значение суммы равно 10. Оно может быть реализовано двумя способами: сложением 0,1/5 и 0,9/5 с минимальным значением $\mu_A = 0,1$ и сложением 0,8/6 с 0,2/4 с минимальным значением $\mu_B = 0,2$. В качестве μ_{A+B} выбираем большее из меньших: $\mu_{A+B} = 0,2$. Аналогичными рассуждениями приходим к окончательному результату:

$$A+B = (0,1/9; 0,2/10; 0,8/11; 0,4/12; 0,3/13).$$

Пример 4.4. Выполнить умножение тех же нечетких чисел A и B , что и в примере 4.3.

Умножение, как и сложение, выполняется относительно всех возможных пар носителей, а результату умножения присваивается меньшая из двух степеней принадлежности. В результате получим:

$$AB = (0,1/20; 0,2/24; 0,1/25; 0,2/28; 0,8/30; 0,4/35; 0,3/36; 0,3/42).$$

При выполнении операции умножения были получены два результата с носителем, равным 30: 0,1/30 и 0,8/30. В соответствии с максиминным принципом в окончательный результат вошло произведение 0,8/30.

4.5.2. Синтез нечеткого регулятора положения

Типичным режимом работы следящих электроприводов СПУ является позиционирование с обеспечением точного останова в заданной точке. Подход к точке останова может осуществляться при различных нагрузках, что приводит к потере точности при статическом регуляторе положения или к затягиванию процесса останова при наличии астатического регулятора. Противодействовать данному явлению можно с помощью нечеткого регулятора положения (НРП), предназначенного для оптимизации режима позиционирования следящего электропривода, структурная схема которого приведена на рис. 4.5.

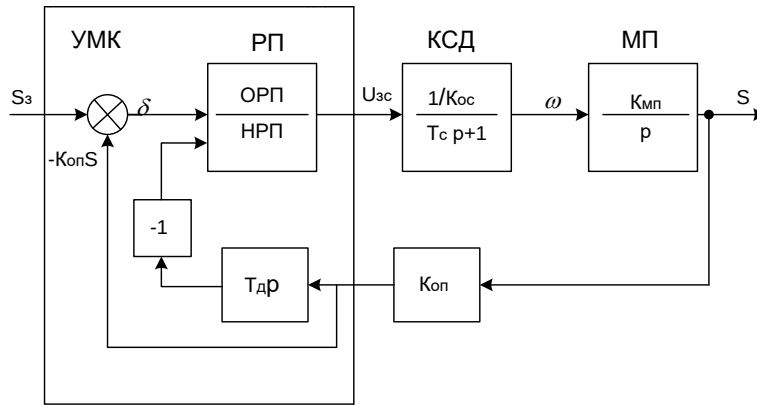


Рис. 4.5. Структурная схема следящего электропривода с нечетким регулятором положения:

УМК- управляющий микроконтроллер;

РП- регулятор положения в составе основного регулятора (ОРП) и нечеткого регулятора (НРП) ;

КСД- контур скорости электродвигателя;

МП- механическая передача.

Здесь микроконтроллер УМК управляет контуром КСД скорости ω двигателя посредством управляющего напряжения $U_{зс}$. Контур скорости может быть реализован комплектным электроприводом постоянного или переменного тока. Настройка КСД на модульный оптимум позволяет отобразить его инерционным звеном с постоянной времени T_c , а K_{oc} –это коэффициент обратной связи по скорости двигателя в контуре скорости. Двигатель приводит в движение рабочий орган через механическую передачу МП с коэффициентом передачи $K_{мп}$. Сигнал обратной связи по положению S рабочего органа подается на вход следящей системы через датчик обратной связи по положению с коэффициентом передачи $K_{он}$. Данный сигнал используется также для формирования обратной связи по скорости перемещения рабочего органа с помощью дифференцирующего звена с постоянной времени T_d .

Значение рассогласования

$$\delta = S_z - K_{он}S \quad (4.39)$$

между заданным положением (уставкой положения) S_z и сигналом обратной связи по положению $K_{он}S$ подается как на основной регулятор положения ОРП, так и на нечеткий регулятор положения НРП, а на НРП кроме того подается сигнал обратной связи по скорости рабочего органа, взятый с обратным знаком. Учитывая, что производная δ по времени равна (при $S_z = const$)

$$\frac{d\delta}{dt} = -K_{он} \frac{ds}{dt},$$

приходим к выводу, что на вход НРП подается сигнал ε , равный

$$\varepsilon = \delta + T_d \frac{d\delta}{dt}. \quad (4.40)$$

Будем далее именовать величину δ ошибкой по положению, а величину $d\delta/dt$ – производной ошибки по положению

Из структурной схемы рис. 4.5 получаем следующее дифференциальное уравнение непрерывной части данного следящего привода:

$$-U_{zc}(\varepsilon) = T_c T_\varepsilon \frac{d^2 \delta}{dt^2} + T_\varepsilon \frac{d\delta}{dt}, \quad (4.41)$$

где

$$T_\varepsilon = \frac{K_{oc}}{K_{on} K_{mn}} - \quad (4.42)$$

эквивалентный коэффициент вязкого трения, численно равный обратной величине коэффициента усиления разомкнутого контура непрерывной части привода;

$U_{zc}(\varepsilon)$ - управляющее напряжение, формируемое на выходе РП.

Значение U_{zc} при управлении от НРП изменяется дискретно. Полагаем, что при $\varepsilon > 0$ имеет место $U_{zc} = U_{з макс}$, при $\varepsilon < 0$ имеем $U_{zc} = -U_{з макс}$, а при малых значениях $|\varepsilon| < \varepsilon_1$, соответствующих зоне нечувствительности РП, должно быть $U_{zc} = 0$.

Полагаем $y = \frac{d\delta}{dt}$ и, с учетом того, что

$$\frac{d^2 \delta}{dt^2} = \frac{dy}{dt} = \frac{dy}{d\delta} \cdot \frac{d\delta}{dt} = \frac{dy}{d\delta} y,$$

записываем дифференциальное уравнение фазовых траекторий рассматриваемого привода в виде

$$\frac{T_c T_\varepsilon y dy}{U_{zc} + T_\varepsilon y} = -d\delta.$$

Интегрируя полученное уравнение, приходим к следующему уравнению фазовых траекторий:

$$T_c(y - y_0) + U_{zc} \frac{T_c}{T_\varepsilon} \ln \frac{U_{zc} + T_\varepsilon y_0}{U_{zc} + T_\varepsilon y} = -(\delta - \delta_0), \quad (4.43)$$

где y_0 , δ_0 - начальные значения фазовых переменных при интегрировании.

При $U_{zc} = 0$ получим

$$T_c(y - y_0) = -(\delta - \delta_0), \quad (4.43')$$

т.е. фазовые характеристики являются при $U_{zc} = 0$ прямыми с отрицательным наклоном. Это означает, что при $U_{zc} = 0$ происходит торможение привода. Если при $y=0$ окажется что $U_{zc} = 0$, то наступит состояние останова ($\delta = \text{const}$), так как отсутствует побудительная причина дальнейшего изменения скорости ($U_{zc} = 0$), а последняя стала равной нулю ($y = \frac{d\delta}{dt} = 0$). С другой стороны, перерегулирование по скорости невозможно, так как контур скорости КСД (см.рис.4.5) эквивалентен апериодическому звену.

Далее будем полагать, что зона нечувствительности релейного НРП определяется заданной точностью позиционирования и составляет $\delta_{zp} = \pm \varepsilon_1$.

В таком случае условиями переключения из режимов $U_{zc} \neq 0$ в режим $U_{zc} = 0$ и обратно будут, согласно (4.40) уравнения

$$\begin{aligned}\varepsilon_1 &= \delta + T_d y; \\ -\varepsilon_1 &= \delta + T_d y.\end{aligned}\quad (4.44)$$

Определим максимальное (граничное) значение y_{zp} , которое при переходе к режиму $U_{zc} = 0$ обеспечивает уменьшение скорости до значения $y = 0$ при $|\delta| \leq \varepsilon_1$. Поскольку уравнения (4.44) симметричны, достаточно рассмотреть случай, когда имеем конечное значение $\delta \leq \varepsilon_1$ при конечном значении $y = 0$. Соотношение между начальными значениями $y_0 = y_{zp}$ и δ_0 в этом случае будет определяться вторым уравнением (4.44), откуда найдем

$$\delta_0 = -(\varepsilon_1 + T_d y_{zp}).$$

Подставив последнее соотношение в (4.43') при $y = 0$, $y_0 = y_{zp}$ и $\delta = \varepsilon_1$, получим (с учетом симметрии параметров)

$$y_{zp} = \pm \frac{2\varepsilon_1}{T_c - T_d}. \quad (4.45)$$

При $T_d \rightarrow T_c$ имеем $y_{zp} \rightarrow \pm \infty$, так что при $T_d \geq T_c$ всегда обеспечивается устойчивый останов привода в пределах заданной зоны $\delta = \pm \varepsilon_1$. Однако оптимальный по быстродействию (скользящий) режим работы следящего электропривода с релейным управлением реализуется лишь при $T_d < T_c$. При работе в скользящем режиме фазовая траектория привода проходит вдоль линий переключения (4.44), отклоняясь от них на величину гистерезиса релейной характеристики РП, который незначителен. Поэтому найдем зависимость от параметров привода максимально допустимого значения U_{zc} , обеспечивающего выход на переключение от $|U_{zc}| > 0$ к $U_{zc} = 0$ при условии, что

$$-\frac{2\varepsilon_1}{T_c - T_d} < y < \frac{2\varepsilon_1}{T_c - T_d}, \quad T_d < T_c.$$

Отсюда найдем условия, при которых обеспечивается устойчивый останов в заданной точке позиционирования, причем внутри зоны нечувствительности.

Полагаем, $U_{zc} > 0$, откуда следует, что движение в сторону уменьшения δ начнется при $\delta_0 > 0$ и $y_0 = 0$.

При $U_{zc} < 0$ начальное значение δ_0 будет противоположным по знаку, а в остальном расчетные соотношения будут такими же. Движение при $U_{zc} > 0$, будет продолжаться до границы, определяемой первым уравнением (4.44), откуда получим условие переключения на работу с $U_{zc} = 0$ в интересующем нас граничном режиме:

$$\delta = \varepsilon_1 - T_d y_{zp}.$$

Подставив полученное значение δ при $y_o=0$ и $y=y_{zp}$ в уравнение (4.43), получим

$$T_c y_{zp} - U_{zc} \frac{T_c}{T_\delta} \ln\left(1 + \frac{T_\delta}{U_{zc}} y_{zp}\right) = T_d y_{zp} - \varepsilon_1 + \delta_0.$$

Теперь необходимо подставить в полученное уравнение значение y_{zp} из соотношения (4.45) в варианте со знаком минус; поскольку имеем $T_d < T_c$, а при $U_{zc} > 0$ и $\delta > 0$ при работе в скользящем режиме имеет место $y < 0$. После подстановки получим:

$$\delta_0 + \varepsilon_1 = -U_{zc} \frac{T_c}{T_\delta} \ln\left[1 - \frac{2\varepsilon_1 T_\delta}{U_{zc}(T_c - T_d)}\right]. \quad (4.46)$$

Из полученного уравнения (4.46) следует, что при

$$|U_{zc}| < \frac{2\varepsilon_1 T_\delta}{T_c - T_d}, \quad T_d < T_c, \quad (4.47)$$

обеспечивается устойчивое завершение позиционирования в пределах $\pm \varepsilon_1$ независимо от величины начального рассогласования δ_0 . Последнее же перед остановом не может превышать значение ε_1 более чем на ширину зоны гистерезиса, т.е. перед остановом имеет место $\delta_0 \cong \varepsilon_1$.

Полученные соотношения используем в алгоритме построения нечеткого релейного регулятора положения. Так, величину ε будем рассчитывать по формуле (4.40). Если $\varepsilon > \varepsilon_1$ и достигнута граница области функционирования нечеткого регулятора ($\delta < \delta_{zp}$), то устанавливаем $U_{zc} = U_{z\text{макс}}$, а если $\varepsilon < -\varepsilon_1$, то устанавливаем $U_{zc} = -U_{z\text{макс}}$. При достижении значения ε пределов $|\varepsilon| \leq \varepsilon_1$, устанавливаем $U_{zc} = 0$. Одновременно задаем пониженное значение U_{zc} , определенное с помощью формулы (4.47) и уточненное при настройке регулятора. Таким образом, подход к заданному положению будет осуществлен с максимальной скоростью, а предотвращение колебаний у заданного положения будет осуществлено после достижения заданной зоны останова (зоны нечувствительности). На этом функции НРП должны заканчиваться. При поступлении большого возмущения, когда $|\delta| > \delta_{zp}$, где δ_{zp} – граничное значение ошибки, определяющее пределы функционирования НРП, нечеткий регулятор должен быть автоматически отключен, а вместо него должен быть включен ОРП.

Необходимость в нечетком регуляторе возникла потому, что при подходе к заданному положению величина δ измеряется малым количеством дискрет (отсчетов) датчика положения, а каждая дискрета занимает относительно соседней дискреты не вполне определенное положение из-за погрешности шкалы датчика. Деления шкалы устанавливаются с некоторой погрешностью, которая при переходе от дискреты к дискрете изменяется, так как имеет случайный характер. При большом количестве отсчетов погрешности отдельных дискрет взаимно компенсируются, но при подходе к заданному положению, когда речь

идет о значениях δ , равных 1-2 дискретам, погрешность может оказаться значительной, близкой к величине одной дискреты. Распределение погрешностей отсчета положения рабочего органа можно отобразить в виде совокупности нечетких множеств

$$A_\delta = \{\mu_\delta(\delta), \delta\},$$

функции принадлежности которых приведены на рис.4.6. Величина δ здесь отложена по оси абсцисс в дискретах, приведенных ко входу РП в соответствии с соотношением (4.39). Распределение функции принадлежности μ_δ на каждой дискрете соответствует равномерному убыванию возможности предполагаемого расположения дискреты в обе стороны от заданной позиции δ_i , которой соответствует $\mu_\delta=1$, до крайних позиций $\delta_i \pm 1$. Предполагается, что максимальная погрешность по положению не превышает одной дискреты. На рис. 4.6.заштрихована область функции принадлежности дискреты $\delta_2 = 2$.

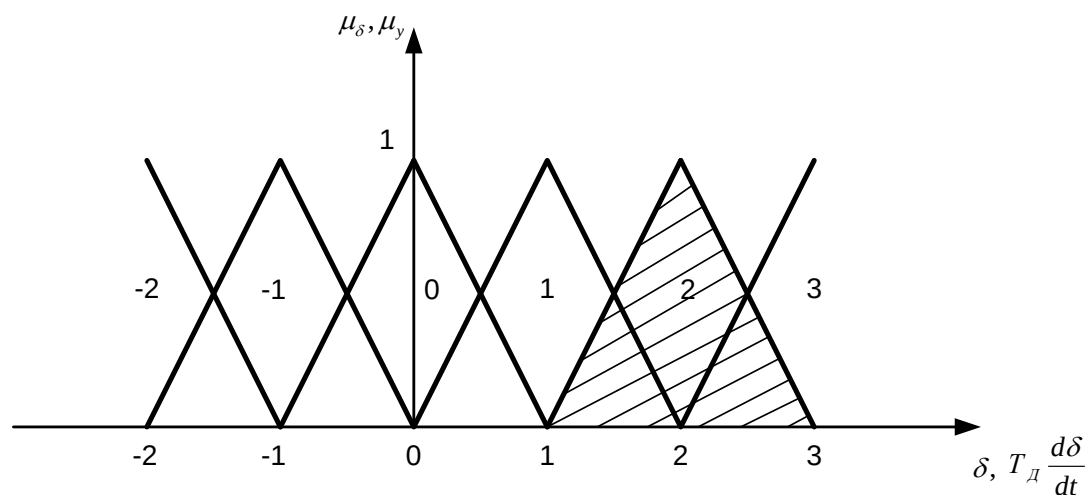


Рис.4.6. Функции принадлежности дискрет ошибки по положению рабочего органа

Видно, что помимо заданной позиции $\delta_2 = 2$ линия дискреты может быть случайным образом расположена с убывающей вероятностью от позиции $\delta = 1$ до позиции $\delta = 3$. Функции μ_γ принадлежности величины $T_d \frac{d\delta}{dt}$, второй составляющей сигнала ε , подаваемого на вход НРП, задаются по тем же правилам, что и μ_δ . Они отображают множества

$$A_\gamma = \{\mu_\gamma(T_d \frac{d\delta}{dt}), T_d \frac{d\delta}{dt}\}.$$

Величина сигнала ε , являющаяся, согласно соотношению(4.40), суммой двух нечетких чисел, также является нечетким числом. Нас будут интересовать реализации ε , соответствующие реализациям δ и $T_d \frac{d\delta}{dt}$, поступающим на вход НРП. Их можно получить по правилам сложения нечетких чисел, рассмотренным выше (см.§ 4.5.1.).

При сложении нечетких чисел значения треугольной функции принадлежности удобно рассчитывать по формуле

$$\mu_i = 1 - \left| \frac{u_i}{a} - i \right|, \quad (4.48)$$

где u_i – значение носителя i - того нечеткого множества;

a – величина дискреты носителя нечеткого множества.

В рассматриваемом случае имеем $a=1$. При $i=2$ заштрихованное нечеткое множество на рис.4.6 имеет $\mu_i=1$ при $u_i=2$, $\mu_i=0$ при $u_i=1$ и при $u_i=3$. В каждом нечетком множестве имеем

$$i-1 \leq u_i \leq i+1.$$

Допустим, в качестве примера, что $\delta=1,7$ и $T_d \frac{d\delta}{dt} = -0,9$ (при работе

в скользящем режиме значения δ и $\frac{d\delta}{dt}$ имеют противоположные знаки).

Носитель $\delta=1,7$ принадлежит одновременно множествам 1 и 2, а носитель $T_d \frac{d\delta}{dt} = -0,9$ – множествам -1 и 0. Отсюда, с помощью формулы (4.48), полагая $a=1$ и $u_i=1,7$ или $u_i=-0,9$, получаем следующие реализации нечетких чисел A_δ и A_y :

$$A_{\delta 1} = 0,3 / 1,7, A_{\delta 2} = 0,7 / 1,7; A_{y(-1)} = 0,9 / -0,9, A_{y0} = 0,1 / -0,9.$$

При вычислении значения ε в соответствии с формулой (4.40) носители A_δ поочередно складываются с носителями A_y и каждой сумме присваивается минимальное значение функции принадлежности слагаемых, см. (4.38). Получим следующий ряд:

$$A_\delta + A_y = \{ 0,3 / 0,8; 0,1 / 0,8; 0,7 / 0,8 \},$$

причем повторяющееся значение $0,1 / 0,8$ записываем только один раз. За истинное значение суммы принимаем, согласно максиминному принципу обобщения,

$$A_\delta + A_y = 0,7 / 0,8,$$

что соответствует значению сигнала управления

$$\varepsilon = 0,7 \cdot 0,8 = 0,56.$$

Если величина ε больше единицы, то значение ε следует вычислять по формуле

$$\varepsilon = a(i) + \Delta a \cdot \mu_a,$$

где $a(i)$ – целая часть значения ε ;

Δa – дробная часть;

μ_a – функция принадлежности ε , вычисленная по вышеизложенной методике.

Для вычисления $T_d \frac{d\delta}{dt}$ рекомендуем использовать разностную формулу

$$T_d \frac{d\delta}{dt} = \frac{T_d}{T} (\delta_2 - \delta_1), \quad (4.49)$$

где T – время цикла управления;

δ_1 и δ_2 – значения δ в начале и в конце цикла.

Значения δ вычисляются по формуле (4.39) с учетом нечеткости S .

Контрольные вопросы к главе 4

1. Перечислите основные направления решения задачи оптимизации функционирования ТО.
2. Каково назначение алгоритмов управления и их классификация?
3. Каковы разновидности и порядок функционирования алгоритмов стабилизации?
4. Как действуют внутренние и внешние возмущения в многосвязных (многомерных) системах управления?
5. Что такое инвариантная (к какому-либо возмущению) система управления?
6. Сформулируйте условия инвариантности управляемых параметров ТО, заданных системой линейных уравнений.
7. Как компенсируются возмущения в инвариантных многосвязных системах?
8. Как и с какой целью вводятся отрицательные обратные связи в отдельных каналах управления?
9. Приведите уравнение i - того канала управления с учетом действия отрицательной обратной связи и объясните его смысл.
10. Сформулируйте понятие статической и динамической оптимизации.
11. Что такое шаг оптимизации?
12. В каких случаях применяется линейное программирование?
13. Сформулируйте основную задачу линейного программирования
14. В каком виде должны быть представлены условия задачи оптимизации для решения ее симплексным методом?
15. Приведите основные положения симплексного метода оптимизации.
16. Опишите методику нахождения точки оптимального режима путем построения многоугольника решений, если целевая функция является линейной функцией двух переменных.
17. Перечислите признаки оптимальности опорного плана при оптимизации симплексным методом.
18. Объясните порядок составления исходной расчетной таблицы при оптимизации симплексным методом.

19. Объясните порядок улучшения опорного плана путем замены одного из векторов базиса на один из векторов $\mathbf{a}_j$, не входящих в состав базиса.
20. Приведите расчетные формулы для нахождения улучшенного опорного плана при оптимизации симплексным методом.
21. В чем заключается сущность градиентных методов?
22. Какие проблемы решаются при выборе величины шага при оптимизации методом градиента?
23. Каков признак достижения оптимального режима при оптимизации методом градиента?
24. Какова методика определения значений частных производных $\frac{\partial F_{\text{ц}}}{\partial x_i}$ при оптимизации методом градиента?
25. Как подбирается значение шагового коэффициента k_j методом удвоения длины шага?
26. В чем заключается метод одномерной минимизации (максимизации), как он связан с методом наискорейшего спуска?
27. Как используется метод градиента при оптимизации электрохимической обработки (ЭХО)?
28. Как используется полнофакторный эксперимент на каждом шаге оптимизации ЭХО?
29. Как ведется поиск допустимого оптимального режима при линейности ограничений параметров техпроцесса?
30. Как ведется поиск допустимого оптимального режима методом штрафных функций?

Глава 5. Алгоритмы управления технологическим циклом

5.1. Задачи управления технологическим циклом

Технологический цикл представляет собой последовательность сменяющих друг друга технологических операций (см. §3.3). После завершения последней из операций, входящих в состав цикла, управляющее устройство переводит технологический объект в режим выполнения первой по счету операции и технологический цикл повторяется.

Системы управления технологическим циклом, называемые обычно системами программного управления (см.гл.7) обеспечивают как последовательную смену технологических операций по мере их выполнения, так и управление каждой из операций, входящих в состав технологического цикла. Таким образом, в порядке управления технологическим циклом, во время выполнения отдельных операций производится необходимая стабилизация параметров техпроцесса на заданном уровне, их программно-следящее изменение, а также решаются задачи автоматической оптимизации техпроцесса в соответствии с его целевой функцией. Вопросы синтеза алгоритмов автоматической стабилизации параметров и оптимизации техпроцессов были разобраны выше. Здесь же мы разбираем вопросы управления последовательной сменой операций в течение технологического цикла. Сложность организации автоматического управления полным технологическим циклом определяется количеством и сложностью выполняемых в составе цикла операций, а также возможностями оперативного выбора различных вариантов продолжения цикла или повторения его частей. Необходимость такого выбора определяется как разнообразием решаемых технологических задач, так и требованиями эффективного управления в аварийных ситуациях. Так, при обработке резанием необходимо обеспечить вывод рабочего органа (резца, сверла и пр.) в рабочую зону, перемещение рабочего органа по заданной траектории в процессе обработки, периодическую смену инструмента, технологических скоростей, режимов работы вспомогательных механизмов в зависимости от производственной ситуации.

АСУТП должна производить логический анализ производственной ситуации и необходимые расчеты, на основании которых она вырабатывает управляющие сигналы по выполнению технологического цикла. Такой анализ аналогичен логическому анализу и расчетам, которые производит человек-оператор перед принятием решения о воздействии на те или иные командные органы при ручном управлении технологическим циклом, таким как, например, вышеупомянутый цикл механической обработки.

В §3.3 мы уже выяснили, что технологические циклы целесообразно представлять в виде последовательности тактов. Содержание операций, производимых системой управления в пределах каждого такта, определяется как комбинацией входных сигналов, поступающих в управляющее устройство в данном такте, так и информацией о состоянии техпроцесса и системы управления в предыдущих тактах. В результате управляющие устройства, обеспечивающие управление технологическим циклом, оказываются по преимуществу *последовательностными автоматами*. Такие автоматы обеспечивают управление *последовательностью тактов* работы технологического объекта, причем выполнение большинства тактов

связано с использованием информации, полученной по каналам обратной связи при выполнении предыдущих тактов.

Устройство управления технологическим циклом делится на узлы и элементы, находящиеся в иерархической соподчиненности по отношению друг к другу. В его составе всегда имеются отдельные узлы, выходные сигналы которых зависят только от *комбинации* входных сигналов, поступивших в текущем такте управления. Такие узлы именуются *комбинационными автоматами*.

Логический анализ производственной ситуации и технологические расчеты, которые его сопровождают, связаны с обработкой информации в дискретном виде (см. §§2.3-2.4). В основе всех УВМ лежит обработка информации в двоичных кодах. Причиной этого является двоичный характер формальных логических рассуждений и экономичность двоичного кодирования (см. §2.4.1). По указанным причинам в основе теории анализа и синтеза систем управления технологическим циклом лежит логический анализ комбинационных и последовательностных автоматов методами булевой алгебры (см. приложение 3).

5.2. Синтез алгоритмов комбинационных схем управления

Комбинационными схемами управления мы будем далее называть комбинационные автоматы, т.е. управляющие устройства, выходные сигналы которых полностью определяются текущей комбинацией входных сигналов. Такие устройства не обладают свойством памяти, неотъемлемым свойством любых реальных управляющих устройств, реализующих полный цикл автоматического управления. Однако комбинационные схемы непременно входят в состав реальных управляющих устройств в качестве их узлов или элементов.

Входными и выходными сигналами комбинационных схем являются дискретные сигналы двух уровней, высокого и низкого. Сигнал высокого уровня при синтезе схем управления обозначают в виде 1 (единица), а сигнал низкого уровня в виде 0 (нуль). Типичными примерами технической реализации единичного сигнала являются:

- замыкание контакта, в результате которого на вход или выход релейной схемы управления поступает напряжение питания данной схемы;
- подача высокого потенциала на вход или выход электронной схемы управления, близкого по величине к напряжению питания схемы (например, +3В при напряжении питания +5В);
- прохождение намагниченного участка магнитного диска (дискеты) в зоне считывания информации.

Аналогичными примерами нулевого сигнала являются:

- размыкание контакта, в результате которого вход или выход схемы отключается от источника питания;

- подача низкого потенциала, близкого к нулю, на вход или выход электронной схемы (например, +0,3В при напряжении питания +5В);
- прохождение ненамагниченного участка магнитного диска (дискеты) в зоне считывания информации.

Перечислим основные этапы составления комбинационных схем управления:

- 1) Определяются все возможные комбинации выходных сигналов (выходных сигналов может быть несколько), соответствующие всем возможным комбинациям входных сигналов, определяющим функционирование управляемого объекта. На основании найденных соответствий входных и выходных сигналов составляется таблица истинности (таблица задания) проектируемого устройства (см. §3.3, табл.2.1).
- 2) По полученной таблице истинности, с помощью аппарата булевой алгебры, составляются выражения логических (булевых) функций, реализуемых проектируемым устройством (см. §ПЗ.4 приложения 3), которые являются его алгоритмом управления. Полученные выражения максимально упрощаются посредством применения операций склеивания и поглощения (см. §§ПЗ.2-ПЗ.3).
- 3) Составляется принципиальная электросхема устройства по формулам полученных логических функций с учетом электротехнических свойств и логических возможностей примененной элементной базы. При составлении принципиальной схемы учитывается нагрузочная способность источников входных сигналов и требования по обеспечению нагрузочной способности выходных цепей.
- 4) Производится конструктивная разработка встройки проектируемой схемы управления в общую систему управления технологическим циклом.

Методику составления комбинационной схемы рассмотрим на примере синтеза преобразователя кода Грея, заданного таблицей 2.1, в арифметический двоичный код. Такой преобразователь может понадобиться для согласования сигналов кодового датчика положения с микропроцессорной системой управления, ведущей обработку числовой информации в двоичном арифметическом коде.

В той же таблице 2.1 приведены кодовые комбинации арифметического двоичного кода, отображающие те же позиции датчика положения, что и заданный код Грея. Эти комбинации соответствуют сигналам, которые должны формироваться на выходе преобразователя кода при подаче на его вход сигналов в коде Грея, формируемых датчиком положения. Следовательно, таблица 2.1 является *таблицей*

истинности синтезируемого преобразователя кода и первичной формой его алгоритма.

При логическом синтезе булевых функций данного преобразователя будем считать, что каждый выходной сигнал является особой функцией четырех входных сигналов, а в дальнейшем учтем интересные для нас связи между данными функциями. Входные сигналы обозначим через X_1, X_2, X_3 и X_4 , а выходные сигналы – через Y_1, Y_2, Y_3 и Y_4 . После этого таблица истинности проектируемого преобразователя кода примет вид таблицы задания (см.таблицу 5.1).

Таблица 5.1

Таблица истинности преобразователя кода

Пози- ции	Входные сигналы				Выходные сигналы			
	X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4
0	0	0	0	1	0	0	0	0
1	0	1	0	1	0	0	0	1
2	0	1	0	0	0	0	1	0
3	1	1	0	0	0	0	1	1
4	1	1	0	1	0	1	0	0
5	1	0	0	1	0	1	0	1
6	1	0	0	0	0	1	1	0
7	1	0	1	0	0	1	1	1
8	1	0	1	1	1	0	0	0
9	1	1	1	1	1	0	0	1
10	1	1	1	0	1	0	1	0
11	0	1	1	0	1	0	1	1
12	0	1	1	1	1	1	0	0
13	0	0	1	1	1	1	0	1
14	0	0	1	0	1	1	1	0
15	0	0	0	0	1	1	1	1

Составление исходного логического выражения по таблице задания производится либо в дизъюнктивной, либо в конъюнктивной форме. При синтезе булевой функции

$$Y_i = f(X_1, X_2, \dots, X_n), \quad i=1, 2, \dots, m,$$

в *дизъюнктивной форме*, т.е. в виде суммы (дизъюнкции) логических произведений, выписываются все наборы значений аргументов $X_1, X_2, \dots, X_n$, которым соответствует значение $Y_i=1$. Затем каждый набор представляют в виде логического произведения (конъюнкции) аргументов $X_1, X_2, \dots, X_n$ или их инверсий, причем в составе конъюнкции оставляют сам аргумент, если в рассматриваемом наборе его значение равно единице, и берут его инверсию, если его значение в рассматриваемой комбинации $X_1, X_2, \dots, X_n$ равно нулю. Формирование логической (булевой) функции Y_i завершается путем суммирования

(взятия дизъюнкции) всех полученных указанным способом произведений, которые принято называть минитермами.

В качестве примера получим в дизъюнктивной форме выражение для Y_4 . Функция Y_4 равна единице во всех нечетных позициях датчика положения (см.таблицу 5.1). В позиции 1 имеем $X_1=0, X_2=1, X_3=0, X_4=1$, что соответствует конъюнкции $\bar{X}_1X_2\bar{X}_3X_4$. Сформировав таким способом конъюнкции входных сигналов для всех случаев, когда $Y_4=1$, и соединив полученные конъюнкции знаками сложения, получим:

$$Y_4 = \bar{X}_1X_2\bar{X}_3X_4 + X_1X_2\bar{X}_3\bar{X}_4 + X_1\bar{X}_2\bar{X}_3X_4 + X_1\bar{X}_2X_3\bar{X}_4 + X_1X_2X_3X_4 + \bar{X}_1X_2X_3\bar{X}_4 + \bar{X}_1\bar{X}_2X_3X_4 + \bar{X}_1\bar{X}_2\bar{X}_3\bar{X}_4. \quad (5.1)$$

Обратим внимание на то, что каждая конъюнкция, входящая в состав выражения (5.1), равна единице только тогда, когда ее аргументы принимают значения, на основании которых данная конъюнкция была сформирована. Так, конъюнкция, $\bar{X}_1X_2\bar{X}_3X_4=1$ если $X_1=0, X_2=1, X_3=0, X_4=1$. Следовательно, вычисление значения Y_4 по формуле (5.1) даст $Y_4=1$ только в тех случаях, когда $Y_4=1$ в таблице 5.1, что и требуется.

Для составления логического (булевого) выражения функции в конъюнктивной форме, т.е. в виде произведения (конъюнкции) сумм аргументов или их инверсий, необходимо выписать из таблицы истинности (таблицы задания) все наборы значений аргументов $X_1, X_2, \dots, X_n$, которым соответствует значение $Y_i=0$. Затем в каждом наборе следует заменить значение $X_i=1$ инверсией $\bar{X}_i$, а вместо значения $X_i=0$ поставить сам X_i и сформировать из этих наборов суммы аргументов $X_1, X_2, \dots, X_n$ или их инверсий. Выражение для Y_i получится после перемножения (взятия конъюнкции) полученных сумм. Далее мы будем работать с логическими выражениями, записанными в дизъюнктивной форме, более удобной для синтеза схем на той элементной базе, какой мы будем пользоваться в разбираемых примерах.

После составления исходного логического выражения булевой функции его следует максимально упростить методом склеивания (см. §ПЗ.3). Поскольку склеивать можно лишь минитермы, которые различаются значением только одного аргумента, то функция Y_4 упрощению данным методом не подлежит. Упрощение логических выражений методом склеивания будет произведено в полной мере лишь тогда, когда будут произведены все возможные для заданной функции склеивания. Рассмотрим регулярную процедуру склеивания, пригодную для машинной обработки, известную под названием метода Квайна-Мак-Класки. Предварительно заметим, что нет необходимости обязательно подставлять значения X_i и их инверсий при выписывании наборов значений аргументов, соответствующих $Y=1$, из таблицы истинности. Ведь принадлежность того или иного значения тому или иному аргументу нетрудно определить по месту расположения единицы или нуля в выписанном наборе. Соответственно, наборы аргументов,

подлежащие склеиванию, будут различаться значениями только одного аргумента.

Теперь переходим к изложению методики Квайна-Мак-Класки.

- 1) Выписывают в столбик наборы аргументов $X_1, X_2, \dots, X_n$ (минитермы) на которых заданная функция равна единице. Минитермы разбивают на группы так, чтобы внутри группы были лишь минитермы с одинаковым числом единиц, а количество единиц в соседних группах было минимально различным. Результаты разбиения на группы минитермов функций, заданных в таблице 5.1, приведены в таблице 5.2. Там же указан ранг минитермов, равный числу входящих в них аргументов.
- 2) Производят все возможные склеивания минитермов соседних групп, отличающихся значением только одного аргумента. При склеивании на месте аргументов, значения которых в склеиваемых минитермах было различным, ставят прочерк. Вместо двух склеенных минитермов появляется один новый, ранг которого понижен на единицу. Склеиваемые минитермы подчеркивают (см.таблицу 5.2).
- 3) Полученные в результате склеивания минитермы сниженного на единицу ранга помещают в новую таблицу. Так, результаты склеивания минитермов 4-го ранга, приведенных в таблице 5.2, помещены в таблицу 5.3. Вновь полученные минитермы разбивают на группы и по возможности склеивают по прежнему правилу, но только тогда, когда прочерки у них находятся в одних и тех же местах. Результаты склеивания минитермов 3-го ранга разбираемого примера приведены в таблице 5.4. А из минитермов 2-го ранга (см.табл.5.4.) склеиваются только те, которые входят в состав функции Y_3 . В результате этого склеивания у функции Y_3 появляется один минитерм 1-го ранга ---0.

Таблица 5.2

Минитермы 4-го ранга	Y_1	Y_2	Y_3	Y_4
----------------------	-------	-------	-------	-------

Нулевая группа	<u>0010</u>	<u>0000</u>	<u>0000</u>	0000
Первая группа	<u>0010</u>	<u>0010</u> <u>1000</u>	<u>0010</u> <u>1000</u> <u>0100</u>	
Вторая группа	<u>0110</u> <u>0011</u>	<u>0011</u> <u>1001</u> <u>1010</u>	<u>0110</u> <u>1010</u> <u>1100</u>	0101 1100 1001 1010 0110 0011
Третья группа	<u>1011</u> <u>1110</u> <u>0111</u>	<u>1101</u> <u>0111</u>	<u>1110</u>	
Четвертая группа	<u>1111</u>			1111

Таблица 5.3

Минитермы 3-го ранга	Y_1	Y_2	Y_3
Нулевая группа	00-0	<u>00-0</u>	<u>00-0</u>
		- <u>000</u>	- <u>000</u>
Первая группа	<u>0-10</u>	<u>001-</u>	<u>0-10</u>
	<u>001-</u>	- <u>010</u>	- <u>010</u>
		100-	<u>10-0</u>
		10-0	<u>1-00</u>
			<u>01-0</u>
			- <u>100</u>
Вторая группа	- <u>110</u>	0-11	- <u>110</u>
	<u>011</u>	1-01	<u>1-10</u>
	- <u>011</u>		<u>11-0</u>
	<u>0-11</u>		
Третья группа	<u>1-11-</u>		
	- <u>011</u>		
	<u>0-11</u>		

Таблица 5.4

Минитермы 2-го порядка	Y_1	Y_2	Y_3
Нулевая группа		-0-0	- <u>0-0</u>
			<u>0--0</u>
			-- <u>00</u>
Первая группа			-- <u>10</u>
	0-1-		- <u>1-0</u>
			<u>1--1</u>
Вторая группа	-11-		
	--11		

- 4) Если в соседних группах нет минитермов, различающихся значением только одного аргумента, то склеивание невозможно. Анализируя в этом смысле минитермы функции Y_4 , приведенные в таблице 5.2, мы видим, что они не склеиваются.
- 5) Минитермы, оставшиеся неподчеркнутыми, т.е. минитермы не подлежащие дальнейшему склеиванию, называются простыми импликантами. Они составляют в своей совокупности минимизированное выражение каждой из логической функций алгоритма синтезируемой комбинационной схемы. Дальнейшая минимизация производится *методом поглощения* (см. §П3.2). При минимизации этим методом часть простых импликантов может быть исключена, если их логические функции выполняются в совокупности другими простыми импликантами. Если логические функции какого-либо простого импликанта полностью перекрываются одним импликантом, то последний хорошо заметен. Если же эти функции перекрываются совокупностью нескольких импликантов, то возможности поглощения не очевидны. Чтобы не упустить имеющиеся возможности упрощения полученного алгоритма, прибегают к составлению *таблицы меток*.

В строках такой таблицы помещают все простые импликанты, а в столбцах – все минитермы высшего ранга минимизируемой функции (в нашем примере – это минитермы 4-го ранга). Далее ставят метки на пересечении каждой строки и столбца, соответствующего минитерму, имеющему в своем составе те же значения аргументов, что и у простого импликанта, представленного в строке (см.табл.5.5 и 5.6). Если какой-либо минитерм покрывается только одним простым импликантом, то последний отмечают звездочкой (см.табл.5.5 и 5.6) и далее именуют существенным импликантом. Последний обязательно включают в минимизированное выражение функции. В разбираемом примере существенными импликантами покрываются все минитермы, так что их достаточно для отображения заданных функций. Если же в таблице меток остаются непокрытые существенными импликантами минитермы, то для них выбирают минимальное покрытие из оставшихся простых импликантов. В заключение заметим, что для функций Y_3 и Y_4 строить таблицы меток нет необходимости, так как их выражения проверяются непосредственно по таблице задания (табл.5.1).

Таблица 5.5

Таблица меток для функции Y_1

Y_1	0000	0010	0110	0011	1011	1110	0111	1111
*00-0	V	V						
0-1-		V	V	V			V	
*-11-			V			V	V	V
*- -11				V	V		V	V

Таблица 5.6

Таблица меток для функции Y_2

Y_2	0000	0010	1000	0011	1001	1010	1101	0111
*-0-0	V	V	V			V		
001-		V		V				
100-			V		V			
*0-11				V				V
*1-01					V		V	

- 6) Минимизированные выражения логических функций, входящих в состав алгоритма комбинационной схемы, переводят в буквенную дизъюнктивную форму по тем же правилам, по каким была получена формула (5.1) для функции Y_4 , но с учетом того, что прочерк в таблице меток означает отсутствие соответствующего аргумента логической формуле. В нашем примере, проставляя в существенных импликантах, приведенных в табл. 5.5 и 5.6, вместо единицы соответствующий аргумент, а вместо нуля инверсию аргумента и суммируя полученные путем такой подстановки конъюнкции, получим следующие выражения для функций Y_1 и Y_2 :

$$Y_1 = \bar{X}_1 \bar{X}_2 \bar{X}_4 + X_2 X_3 + X_3 X_4; \quad (5.2)$$

$$Y_2 = \bar{X}_2 \bar{X}_4 + \bar{X}_1 X_3 X_4 + X_1 \bar{X}_3 X_4. \quad (5.3)$$

Так как у функции Y_3 имеется единственный простой импликант: ---0, то ее логическая формула до предела проста:

$$Y_3 = \bar{X}_4. \quad (5.4)$$

Выражение для Y_4 определяется формулой (5.1), поскольку не подлежит, как показано выше, упрощению методом Квайна – Мак-Класки.

5.3.Схемная реализация релейно-контактных комбинационных схем

Порядок составления комбинационных схем управления определяется используемой элементной базой. Последняя состоит из двух резко отличных друг от друга классов: релейно-контактной аппаратуры и электронной аппаратуры. Сложные логические функции управления реализуются преимущественно с помощью электронной аппаратуры, но силовые исполнительные устройства нередко комплектуются контакторами, электромагнитами и другой

электромеханической аппаратурой, которая выполняет частично и логические операции. Рассмотрим схемную реализацию комбинационных алгоритмов, полученных в виде набора логических (булевых) функций, с помощью, с одной стороны, релейно-контактной аппаратуры, а с другой стороны – с помощью типовых логических элементов, являющихся электронными устройствами.

При рассмотрении релейно-контактных логических устройств будем полагать, что, если цепь питания катушки реле разомкнута, то на входную цепь реле (цепь ее катушки) подан нулевой сигнал, а если на катушку реле подано питающее напряжение, то это означает подачу единичного входного сигнала на вход данного реле. Выходными сигналами реле, рассматриваемого как управляющее устройство, являются сигналы, формируемые его контактами, переключаемыми при притяжении якоря реле к сердечнику и при отпуске якоря. Контакты реле рисуют разомкнутыми, если они разомкнуты при отключенном питании катушки данного реле (нормально разомкнутые, или замыкающие контакты). Если же контакты реле замкнуты при отключенном питании его катушки, то их рисуют замкнутыми и именуют нормально замкнутыми или размыкающими.

При подаче единичного сигнала (напряжения питания) на вход реле (на его катушку) замыкающие контакты данного реле замыкаются и подключают к источнику питания нагрузку этого реле путем подачи на нагрузку (например, на катушку другого реле) единичного сигнала. В этом случае реле реализует функцию логического повторения:

$$Y = X,$$

где X – входной сигнал реле;

Y – выходной сигнал того же реле.

Здесь учитывается также, что при подаче на вход реле нулевого сигнала (снятия питания катушки) сигнал на выходе также станет равным нулю (нагрузка будет отключена). Если на вход реле поданы параллельно несколько различных сигналов (см. рис. 5.1), то при равенстве любого из них порознь или вместе единице единичное напряжение, т.е. напряжение питания, будет подано на обмотку выходного реле Y , что приведет к подаче напряжения питания на нагрузку его замыкающим контактом Y .

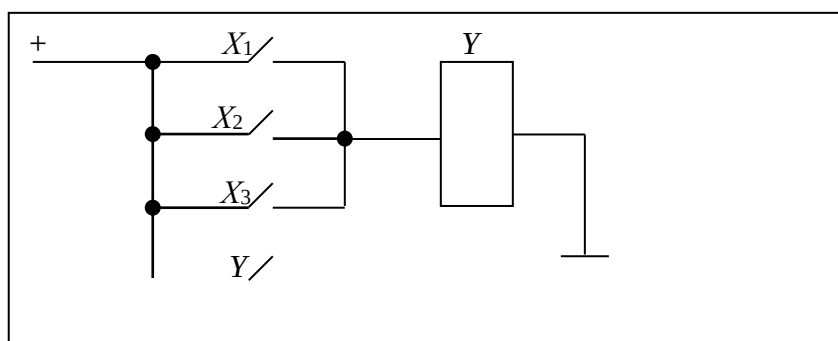


Рис. 5.1. Реализация функции ИЛИ на реле.

На рис. 5.1 подача единичного сигнала на вход реле Y обеспечивается подачей такого же единичного сигнала на вход, т.е. на обмотку любого из входных реле X_1 , X_2 или X_3 порознь или вместе. Обмотка реле Y потеряет питание, т.е. на вход реле Y будет подан нулевой сигнал, только тогда, когда потеряют питание все реле: X_1 , X_2 и X_3 , что означает подачу на их входы (на обмотки) нулевых сигналов. Итак, достаточно появиться хотя бы одному единичному сигналу на входе реле Y , как на его выходе также появится единичный сигнал. Таким образом, работа реле X_1 , X_2 и X_3 на рис.5.1 соответствует закону логического сложения (см. §ПЗ.3):

$$Y = X_1 + X_2 + X_3,$$

согласно которому имеем $Y=1$ при любом единичном значении X_1 , X_2 или X_3 порознь или вместе. Тем самым доказано, что *логическое сложение (дизъюнкция, функция ИЛИ) схемно реализуется параллельным соединением контактов реле.*

Если в цепь нагрузки реле подключены его размыкающие контакты, то при наличии нулевого сигнала на его входе (обмотка отключена от источника питания) на нагрузку реле через размыкающий контакт будет подано напряжение питания, т.е. единичный сигнал. При подаче единичного сигнала на вход этого реле, его размыкающие контакты на выходе разомкнутся и отключат нагрузку от источника питания, т.е. подадут на нагрузку нулевой сигнал. Таким образом, благодаря использованию размыкающих контактов подача на вход X единичного сигнала приводит к появлению на выходе Y нулевого сигнала и наоборот: подача на вход реле нулевого сигнала приводит к формированию единичного сигнала на его выходе. Это соответствует функции логической инверсии (см. §ПЗ.3)

$$Y = \bar{X}.$$

Отсюда правило: *логическая инверсия (логическое отрицание, функция НЕ) реализуется размыкающим контактом реле.*

Если контакты нескольких реле соединены последовательно и через них подается питающее напряжение на нагрузку, то питающее напряжение будет подано, если все контакты окажутся замкнутыми. Это случится, если на входы реле, которым принадлежат замыкающие контакты, будут поданы единичные сигналы, а на входы реле, которым принадлежат размыкающие контакты, включенные в ту же последовательную цепочку, будут поданы нулевые сигналы. Так, все контакты последовательного соединения, приведенного на рис.5.2., будут замкнуты, если будет реализовано следующее сочетание входных сигналов: $X_1=1$, $X_2=0$, $X_3=1$.

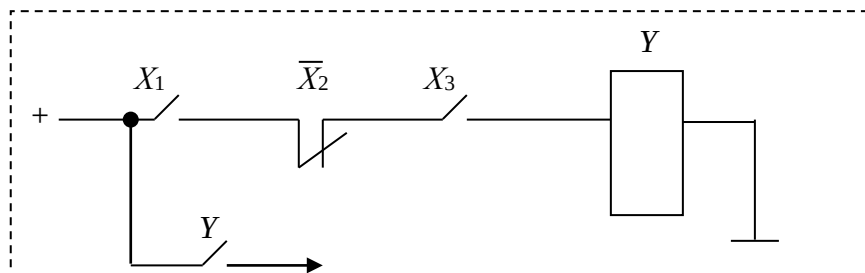


Рис. 5.2. Реализация функции И на реле

Это соответствует реализации функции логического умножения в форме (см. §П3.2)

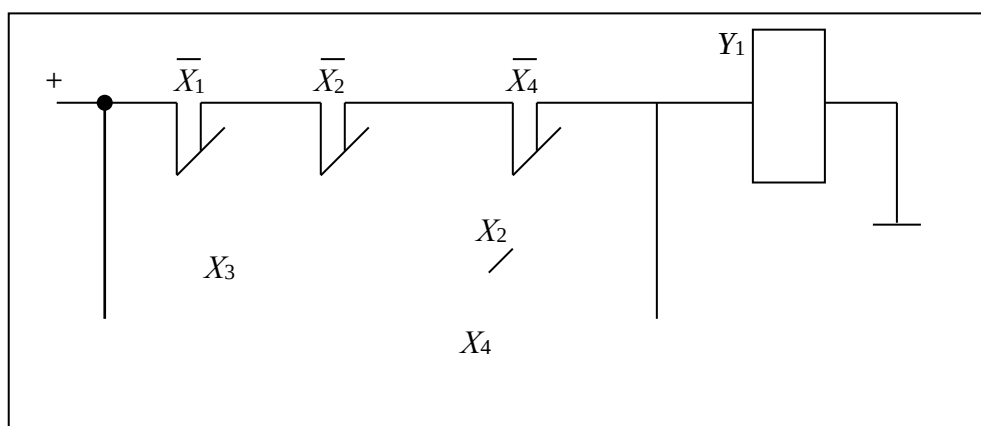
$$Y = X_1 \bar{X}_2 X_3,$$

поскольку функция Y равна единице только в случае равенства единице всех сомножителей заданного произведения. Отсюда получаем правило: *логическое произведение (конъюнкция, функция совпадения, функция И) схемно реализуется последовательным соединением контактов реле.*

Сформулированные правила схемной реализации логических алгоритмов носят название метода последовательно-параллельных соединений. Они естественным образом распространяются на более сложные сочетания контактов. Так, при схемной реализации функции Y_1 , заданной формулой (5.2), мы сначала вынесем за скобки X_3 :

$$Y_1 = \bar{X}_1 \bar{X}_2 \bar{X}_4 + X_3(X_2 + X_4),$$

а затем составим релейную схему, реализующую функцию Y_1 , по сформулированным выше правилам. Составленная схема показана на рис.5.3. Видно, что применение скобочной формы позволило уменьшить количество контактов, необходимых для схемной реализации функции Y_1 .



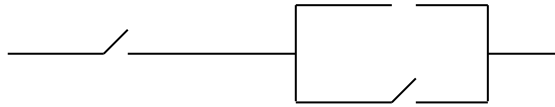
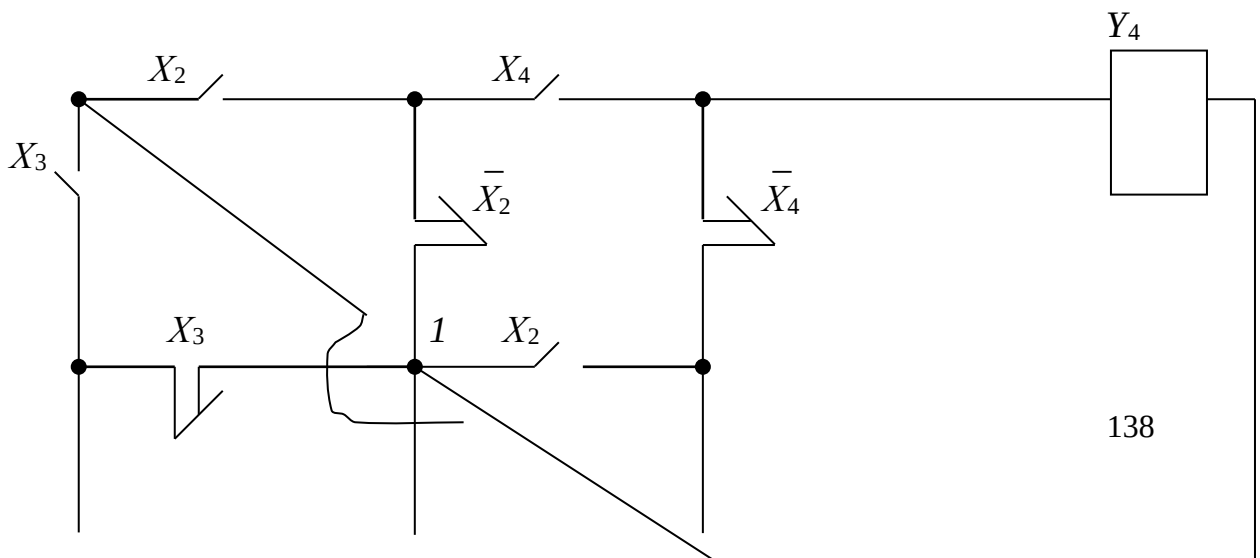


Рис. 5.3. Реализация Y_1 на реле

Особенно существенного упрощения схемы удастся добиться при сочетании скобочных форм с мостиковыми структурами для реализации логических формул типа формулы (5.1.), отображающей функцию Y_4 . Непосредственное схемное воплощение формулы (5.1) методом последовательно-параллельных соединений приводит к релейной структуре из 32 пар контактов. Чтобы реализовать Y_4 более экономно, предварительно представим формулу (5.1) в такой скобочной форме:

$$Y_4 = (X_1\bar{X}_3 + \bar{X}_1X_3)(X_2\bar{X}_4 + \bar{X}_2X_4) + (X_1X_3 + \bar{X}_1\bar{X}_3)(X_2X_4 + \bar{X}_2\bar{X}_4) . \quad (5.5)$$

Непосредственная релейно-контактная реализация Y_4 , исходя из формулы (5.5), приводит, как нетрудно подсчитать, к схеме, построенной на 16 парах контактов. Эту схему можно дополнительно упростить, если сначала реализовать первую половину формулы (5.5), как показано на рис.5.4, в качестве сердцевины, к которой пристроить мостики из 4 контактов: $X_3, X_2, \bar{X}_3 \text{ и } \bar{X}_2$ (см.рис.5.4). Теперь формула (5.5) реализована только на 12 контактах.



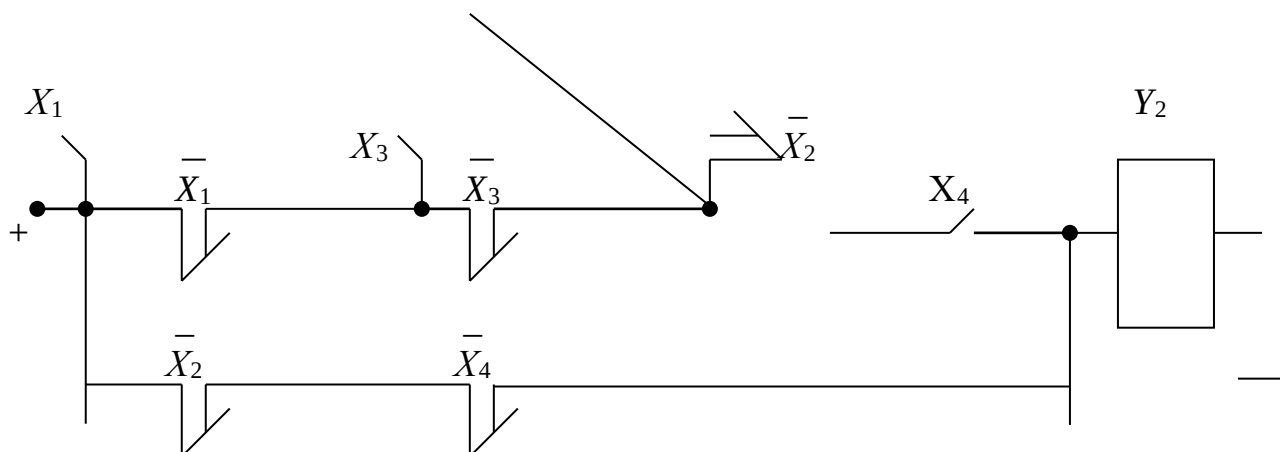


Рис.5.4. Мостиковая реализация функций Y_4 и Y_2 .

Полученная мостиковая структура (рис.5.4) удобна еще и тем, что ее отдельные участки могут быть использованы для реализации других заданных функций. Так, если в формуле (5.3) вынести за скобки X_4 , то получим

$$Y_2 = \bar{X}_2 \bar{X}_4 + X_4 (X_1 \bar{X}_3 + \bar{X}_1 X_3). \quad (5.6)$$

Формула (5.6) содержит составляющую $(X_1 \bar{X}_3 + \bar{X}_1 X_3)$, которая уже имеется в составе формулы (5.5), отображающей функцию Y_4 . Это дает возможность использовать часть релейной структуры Y_4 , реализующей $(X_1 \bar{X}_3 + \bar{X}_1 X_3)$ для реализации функции Y_2 (см.рис.5.4). Остальная часть функции Y_2 реализуется обычным способом.

В полученной совместной релейной структуре функций Y_2 и Y_4 дополнительные контакты $\bar{X}_2, \bar{X}_4$ и X_4 , добавленные ради реализации Y_2 , не нужны для реализации Y_4 . Чтобы функция не была искажена дополнительными релейными цепочками, подобными цепочке $\bar{X}_2 - \bar{X}_4 - X_4$ необходимо, чтобы в каждой подсоединенной цепочке были взаимоисключающие контакты или группы контактов. Это правило является общим законом объединения релейно-контактных структур. В разбираемом примере взаимоисключающими контактами в составе релейной структуры Y_2 являются контакты X_4 и $\bar{X}_4$ благодаря которым в точку 1 соединения релейных структур Y_2 и Y_4 никогда не поступит единичный сигнал от клеммы «+» источника питания по цепочке $\bar{X}_2 - \bar{X}_4 - X_4$. Действительно, если $X_4=1$, то контакт X_4 замкнут, но соединенный с ним последовательно контакт $\bar{X}_4$ обязательно разомкнут и препятствует поступлению потенциала + в точку 1 через контакт $\bar{X}_2$. Если же $X_4=0$, то обязательно разомкнут контакт X_4 с теми же последствиями. Единичный сигнал может поступать от клеммы + в точку 1 только по последовательным цепочкам $X_1 - \bar{X}_3$ и $\bar{X}_1 - X_3$, так как во всех остальных цепочках контактов, подведенных к точке 1 со стороны релейной структуры Y_4 , имеются взаимоисключающие контакты.

Добавив к релейной схеме рис.5.4 релейную схему рис.5.3 и сформировав с помощью размыкающего контакта сигнал $\bar{X}_4$ необходимый для реализации функции Y_3 , мы завершим синтез задуманного нами еще в §5.2 преобразователя кода. Однако схема такого преобразователя на электромагнитных реле не годится для практического применения ввиду ограниченного быстродействия и недостаточной надежности. Ниже будет рассмотрена схема данного преобразователя кода на электронных логических элементах.

5.4. Схемная реализация комбинационных схем на логических элементах

Логическими элементами называют микросхемы малой степени интеграции, реализующие простейшие логические функции 2-4 аргументов. Наиболее распространены логические элементы, реализующие логические функции И (рис. 5.5а), ИЛИ (рис. 5.5б), И-НЕ (рис. 5.5в) и ИЛИ-НЕ (рис. 5.5г). К логическим элементам относят также микросхемы, реализующие простейшие последовательностные алгоритмы (например, триггеры), но они будут рассмотрены ниже.

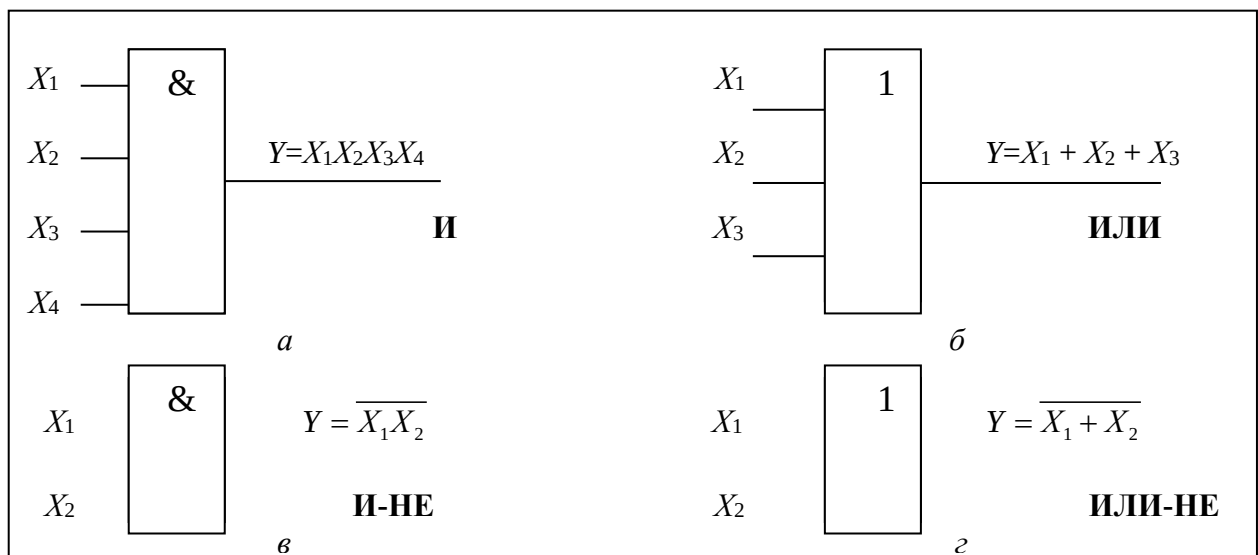




Рис.5.5. Схемные обозначения логических элементов

Логические элементы И реализуют функцию логического умножения (конъюнкцию). Это означает, что выходной сигнал схемы И равен единице только в том случае, когда все ее входные сигналы равны единице. Логический элемент И называют также схемой совпадения.

Логические элементы ИЛИ реализуют функцию логического сложения (дизъюнкцию), т.е. сигнал на выходе схемы ИЛИ равен нулю только тогда, когда все входные сигналы равны нулю.

Логические элементы И-НЕ реализуют функцию инверсии логического произведения (функцию Шеффера), а элементы ИЛИ-НЕ – функцию инверсии логической суммы (функцию Пирса). Таким образом, если логическое произведение равно единице, то элемент И-НЕ выдает нулевой сигнал на своем выходе, а если логическая сумма равна единице, то элемент ИЛИ-НЕ также выдает нулевой сигнал. В противном случае на выходах элементов данного типа формируется единичный сигнал (см. табл. ПЗ.1).

В одном корпусе микросхемы обычно имеется четыре логических схемы на два входа каждая, либо три схемы на три входа каждая, либо две схемы на четыре входа независимо от вида элементарных логических функций, которые данные микросхемы реализуют. Если не все входы логической схемы используются в проектируемом устройстве, то неиспользуемые входы следует объединять с используемыми. Так, для реализации функции инверсии необходимо объединить все входы схемы И-НЕ (см.рис.5.6а). Тогда получим

$$Y = \overline{XX} = \overline{X}.$$

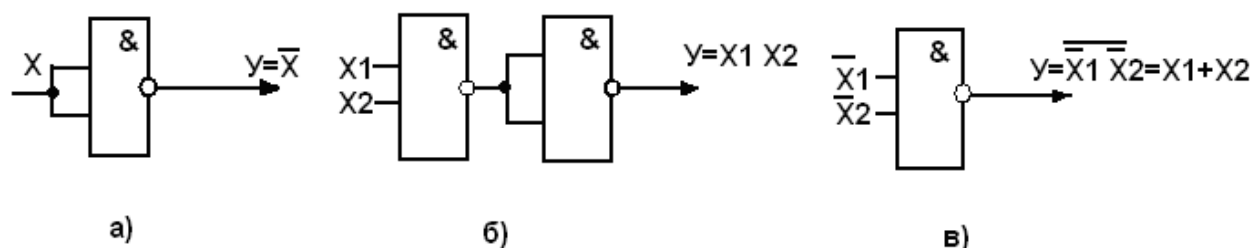


Рис.5.6. Реализация базовых логических функций на элементах И-НЕ

Если две схемы И-НЕ соединить последовательно, как показано на рис.5.6б, то вторая схема инвертирует инверсию логического произведения, полученного на первой схеме, так что на выходе второй схемы получим само логическое произведение. Если же на вход схемы И-НЕ подать инверсии интересующих нас сигналов (см. рис. 5.6в), полученных предварительно с помощью опять же схемы И-НЕ, то на

выходе получим логическую сумму исходных сигналов согласно закону Де Моргана (см. §ПЗ.3). Таким образом, с помощью элементов И-НЕ можно реализовать все базовые функции булевой алгебры, а следовательно – любые логические функции. Так же универсальны и элементы ИЛИ-НЕ. Элементы других типов, которые при наличии элементов И-НЕ или ИЛИ-НЕ не являются обязательными для реализации алгоритмов управления, имеют, как правило, специальное назначение. Так, элементы И (см. рис. 5.5а) обычно являются усилительными элементами. Их допустимый выходной ток достигает 100мА, в то время как обычные логические элементы имеют допустимый выходной ток до 5 мА.

Пример 5.1. Реализация на логических элементах преобразователя кода, логические функции которого отображаются формулами (5.2) – (5.5). Один из возможных вариантов схемы приведен на рис 5.7.

Выходные сигналы датчика положения должны быть, прежде всего, инвертированы, чтобы их можно было подать во входные цепи преобразователя как в прямом, так и в инверсном виде. Соответственно, в левой части схемы (рис.5.7а) показано, как входные сигналы X_1, X_2, X_3 и X_4 инвертируются на логических элементах D1/1, D1/2, D1/2 и D1/4, реализующих функцию И-НЕ в составе микросхемы D1. Схема, производящая непосредственное преобразование кода Грея, в котором работает датчик положения, в прямой арифметической двоичный код, удобный для использования в УВМ, приведена на рис.5.7б. При построении схемы были учтены возможности использования отдельных ее цепей, реализующих функцию Y_4 , для реализации остальных заданных функций. Такие возможности видны при сравнении логических формул, реализующих заданные функции. Так, при сравнении выражений (5.5) и (5.6), видно, что у функций Y_4 и Y_2 имеются общие члены

$$X_1\bar{X}_3 + \bar{X}_1X_3 \text{ и } \bar{X}_2\bar{X}_4 ,$$

а у функции Y_1 , согласно выражению (5.2), можно выделить кроме $\bar{X}_2\bar{X}_4$ еще и инверсное ему выражение

$$X_2 + X_4 = \overline{\bar{X}_2\bar{X}_4}.$$

Реализация самой функции Y_4 значительно упростилась, поскольку было принято во внимание, что если обозначить

$$\begin{aligned} a &= X_1\bar{X}_3 + \bar{X}_1X_3 \\ b &= \bar{X}_2\bar{X}_4 + X_2X_4 \end{aligned} \quad (5.7), (5.8)$$

то окажется, что

$$\begin{aligned} \bar{a} &= (\bar{X}_1\bar{X}_3 + X_1X_3) \\ \bar{b} &= (X_2\bar{X}_4 + \bar{X}_2X_4) \end{aligned}$$

Поэтому достаточно реализовать функции a и b по формулам (5.7), а выражения $\bar{a}$ и $\bar{b}$, необходимые, как видно из формулы (5.5), для реализации функции Y_4 , получить путем простой инверсии.

Функция a реализована на элементах И-НЕ D2/1, D2/2 и D3/1. На выходах D2/1 и D2/2 получены соответственно инверсии $\overline{X_1 \bar{X}_3}$ и $\overline{\bar{X}_1 X_3}$ произведений входных сигналов X_1 и $\bar{X}_3$, а также $\bar{X}_1$ и X_3 . На выходе D3/1 получена, в свою очередь, инверсия произведений входных сигналов схемы D3/1, т.е. получено:

$$a = \overline{X_1 \bar{X}_3 \cdot \bar{X}_1 X_3} = X_1 \bar{X}_3 + \bar{X}_1 X_3 .$$

Аналогично на выходе D3/4 получено:

$$b = \overline{\bar{X}_2 \bar{X}_4 \cdot X_2 X_4} = \bar{X}_2 \bar{X}_4 + X_2 X_4 .$$

На выходах элементов D3/2 и D5/1, на входы которых поступают сигналы b и a , получены, соответственно, сигналы $\bar{b}$ и $\bar{a}$. Далее, с учетом того, что из формулы (5.5) следует

$$Y_4 = a\bar{b} + \bar{a}b ,$$

реализуем функцию Y_4 аналогично тому, как ранее реализовали функции a и b . Однако на выходе структуры, реализующей Y_4 , ставим вместо элемента И-НЕ усилительный элемент И (элемента D6). Поэтому на первом выходе D6 получаем $\bar{Y}_4$. Аналогично на втором сверху выходе D6 получаем $\bar{Y}_3$, подав на его вход сигнал X_4 , в то время как $Y_3 = \bar{X}_4$.

Инверсные сигналы на выходах нашей схемы позволяют организовать индикацию посредством сигнальных ламп. Действительно, если имеет место, например, $Y_4=1$, то $\bar{Y}_4=0$. Выходной потенциал элемента D6/1 близок к нулю и к лампочке HL4 приложено почти все напряжение питания. Следовательно, при $Y_4=1$ лампочка HL4 будет светиться. Так же будут светиться остальные сигнальные лампы. Если же необходимо получить на выходах схемы проектируемого преобразователя прямые сигналы Y_i , то полученные сигналы $\bar{Y}_i$ следует дополнительно инвертировать.

Функция Y_2 легко реализуется подобно функции a , если обратить внимание на то, что выражение (5.6) преобразуется к виду

$$Y_2 = \bar{X}_2 \bar{X}_4 + aX_4 .$$

При реализации функции Y_2 инверсию произведения $\bar{X}_2 \bar{X}_4$ берем с выхода D2/3, а инверсию произведения aX_4 формируем на выходе элемента D4/2. Далее формируем выходной сигнал Y_2 на первом выходе элемента D7 аналогично тому, как формировался выходной сигнал Y_4 на первом выходе D6.

Реализация функции Y_1 проводится аналогично, но с учетом того, что выходной сигнал элемента D2/3 – это

$$\overline{\bar{X}_2 \bar{X}_4} = X_2 + X_4 ,$$

а инвертируя его на элементе D4/1, получаем $\bar{X}_2 \bar{X}_4$.

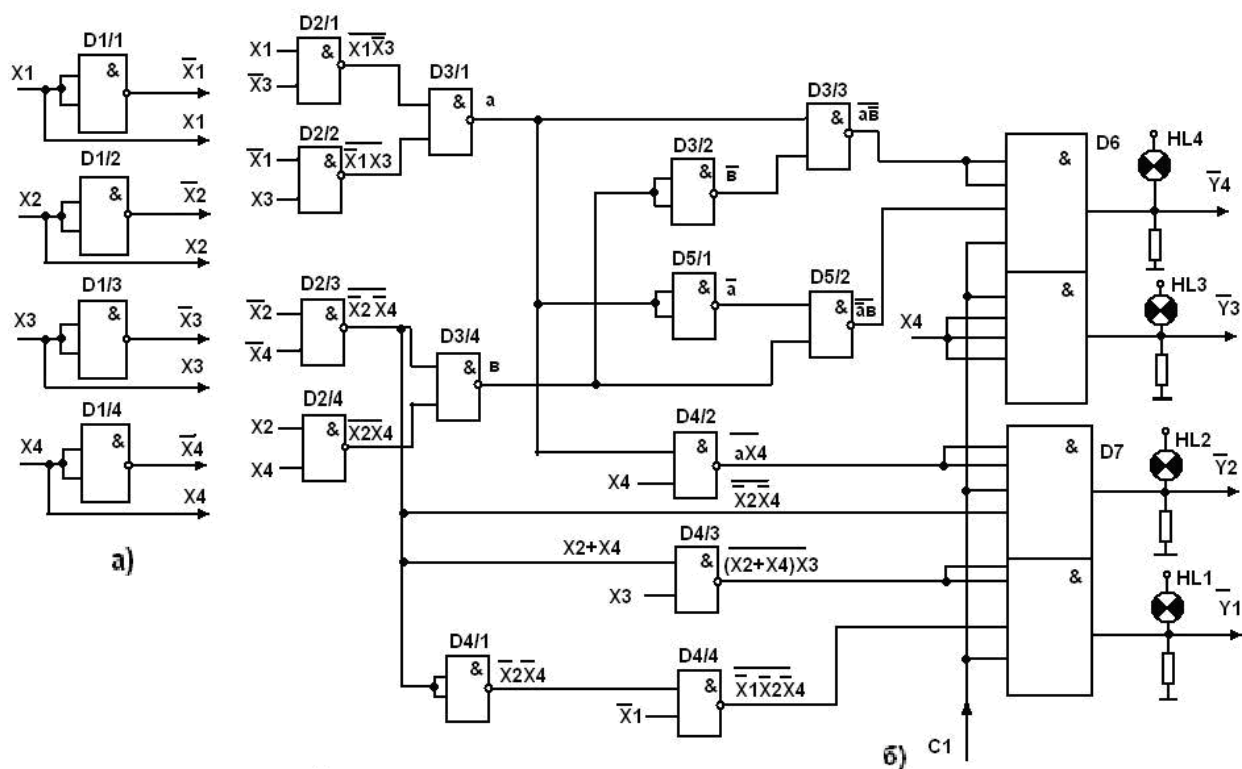


Рис.5.7. Схема преобразователя кода на логических элементах

В заключение заметим, что преобразователь кода в том виде, в каком он представлен на рис. 5.7, не обеспечивает одновременности преобразования входных сигналов X_i в выходные сигналы Y_i . Так, при изменении значения входного сигнала X_1 значение выходного сигнала Y_1 изменится скорее, чем значения Y_2 и Y_4 : сигнал со входа X_1 достигает выхода $\bar{Y}_1$ пройдя три последовательно соединенных элемента (D1/1, D4/4, D7/2), а выхода Y_2 , например, он достигает, пройдя пять элементов (D1/1, D2/2, D3/1, D4/2 и D7/1). Это явление носит название гонки импульсов и приводит, если ему не противодействовать, к появлению кратковременных ложных сигналов на выходах и к сбоям в работе системы управления. Наиболее действенное средство борьбы с гонкой импульсов – переход от асинхронного к синхронному способу управления. При *асинхронном управлении* сигналы обратной связи принимаются в произвольные моменты времени, по мере их поступления и изменения, а при *синхронном управлении* прием сигналов обратной связи производится в определенные промежутки времени, называемые временем считывания. Эти промежутки времени задаются в виде определенного количества тактов тактового генератора, определяющего временные интервалы всего технологического цикла. При синхронном управлении считывание информации с выходов преобразователя кода производилось бы только во время подачи на синхронизирующий вход C1 (см.рис.5.7б) единичных сигналов. Предполагается, что ко времени считывания переходные процессы в преобразователе кода должны быть завершены, что обеспечивает

однозначность считываемой информации. Чтобы получалось действительно так, преобразователь кода должен работать по схеме последовательностного, а не комбинационного устройства управления

5.5. Синтез алгоритмов последовательностных автоматов

5.5.1. Общая структура последовательностного автомата

Последовательностными автоматами называют управляющие устройства, выходные сигналы которых зависят не только от комбинации входных сигналов, имевших место в текущем такте технологического цикла, но и от комбинаций входных сигналов, имевших место в предыдущих тактах и повлиявших на *внутреннее состояние* автомата. Так, при перемещении рабочего органа станка по линейной траектории и достижении заданной точки рабочей зоны станка, система программного управления переходит к выполнению нового кадра программы, в котором может быть задано опять же перемещение по линейной траектории, но в другом направлении и с другой скоростью. Сигналом к переходу на отработку нового кадра программы является в этом случае совокупность сигналов датчиков положения о достижении заданной точки в пространстве.

В простейших последовательностных автоматах, которые далее будем называть просто автоматами, программа работы жестко закладывается в конструкцию автомата и определяет последовательность смены его состояний. Каждое состояние автомата характеризуется отличным от соседнего состояния способом реагирования на поступающие входные сигналы. Переход от одного состояния к другому определяется как комбинацией входных сигналов, так и конкретным состоянием, в котором находится автомат. Самым простым способом смены состояний является случай, когда все состояния пронумерованы и их смена производится в порядке возрастания (или уменьшения) номеров. Запоминание состояний автомата обычно производится с помощью двоичных элементов памяти, таких как электронный триггер (см. §5.6) или электромагнитное реле. Число состояний, которые можно запомнить с помощью совокупности из m таких элементов достигает

$$M=2^m,$$

так что количество запоминающих элементов m выбирается исходя из неравенства

$$m \geq \log_2 M, \quad (5.9)$$

где M – количество состояний, которые подлежат запоминанию.

Если обозначить через X совокупность сигналов обратной связи (входных сигналов), поступающих от ТО, через Y – совокупность сигналов управления (выходных сигналов), подаваемых на ТО, через Z – совокупность внутренних управляющих сигналов, отображающих текущее состояние автомата, то окажется, что одним и тем же значениям

X соответствуют различные значения Y , если при тех же X значения Z различны.

Общая структура последовательного автомата приведена на рис.5.8.

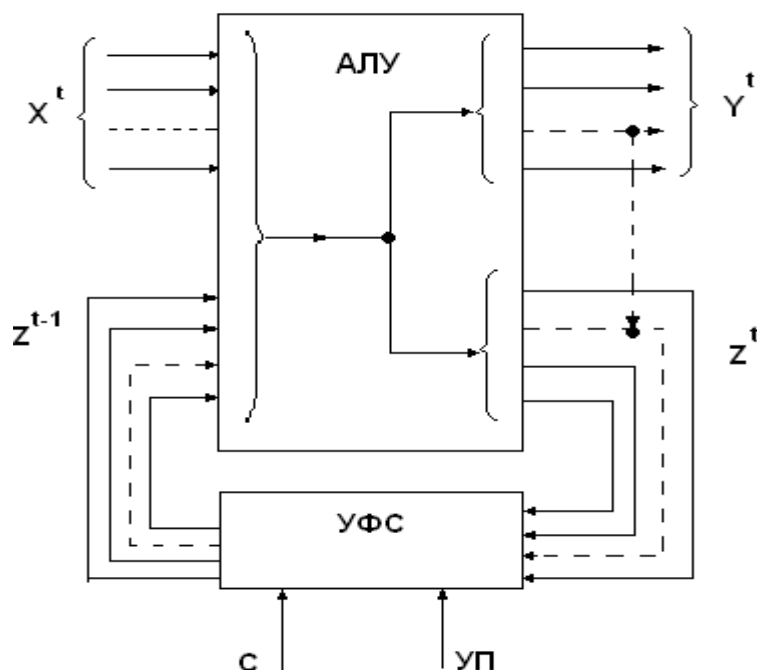


Рис.5.8. Общая структура последовательного автомата

Видно, что автомат состоит из двух основных блоков: арифметико-логического устройства (АЛУ) и устройства формирования состояний (УФС). Блок АЛУ является комбинационной частью последовательного автомата. Он производит заданные арифметические и логические операции над входными сигналами X^t и управляющими сигналами Z^{t-1} , причем у простейших автоматов производятся только логические операции. Устройство формирования состояний автомата (УФС), действуя посредством совокупности управляющих сигналов Z^{t-1} , задает операции которые должно произвести АЛУ над входными сигналами X^t в текущем рабочем такте. Выходные сигналы АЛУ подразделяются на две группы: Y^t и Z^t , причем индекс t означает момент времени или номер такта, в котором были выработаны данные сигналы. Сигналы Y^t – это управляющие сигналы, подаваемые на технологический объект, а сигналы Z^t – это внутренние управляющие сигналы, характеризующие внутреннее состояние автомата. Они подаются, совместно с некоторыми сигналами Y^t , в УФС и там запоминаются. Далее эти сигналы обрабатываются в соответствии с действиями, предусмотренными очередным кадром управляющей программы УП, и подаются на вход АЛУ после прихода очередного тактового импульса C на вход синхронизации. Синхронизация предотвращает подачу управляющих сигналов Z^{t-1} до того, как они будут полностью сформированы в УФС. После того, как они сформируются и

будут поданы в АЛУ, в УФС смогут поступить новые сигналы Z^t и Y^t , сигналы следующего такта управления. Следовательно, управляющие сигналы Z^{t-1} на входе АЛУ относятся к предыдущему такту работы АЛУ, что и отражается индексом $t-1$.

У простейших автоматов внешняя УП после задания режима работы в УФС не поступает. Их поведение в технологическом цикле целиком определяется распределением сигналов Y^t и Z^t , которые поступают в УФС и там запоминаются. После подачи тактового импульса на вход C совокупность сигналов Z (часть которых может совпадать с выходными сигналами Y) поступает на входы АЛУ, задавая логические операции, которые оно должно совершить в течение нового рабочего такта. У очень простых автоматов, работающих по принципу асинхронного управления, особый генератор тактовых импульсов отсутствует, а гонка импульсов предотвращается тем, что при любых переходах из одного состояния в другое допускается изменение значения только одного сигнала из всей совокупности внутренних управляющих сигналов Z . В этом случае изменение очередного сигнала z_i из совокупности Z является как бы синхронизирующим импульсом, задающим функционирование АЛУ в новом такте t_{i+1} работы автомата.

Алгоритм последовательностного автомата удобно составлять исходя из структурной схемы, приведенной на рис.5.8. На этой схеме АЛУ является комбинационной частью автомата, а УФС включает в себя запоминающие устройства (ЗУ), задающие посредством сигналов Z^{t-1} режим работы АЛУ в каждом рабочем такте автомата в течение технологического цикла. В простейших автоматах сигналы Z^{t-1} – это сигналы Z^t и, возможно, часть сигналов Y^t , сформированные в предыдущем такте автомата и запомненные в УФС. Следовательно, составление алгоритма простого автомата естественным образом разделяется на два этапа. На первом этапе составляется алгоритм функционирования АЛУ в виде

$$Y^t = f_1(X^t, Z^{t-1}); \quad Z^t = f_2(X^t, Z^{t-1})$$

по правилам составления комбинационных схем, а затем на втором этапе уточняется структура ЗУ в составе УФС и порядок формирования синхронизирующих импульсов. Функцию Y^t принято называть функцией выходов, а функцию Z^t – функцией переходов от состояния к состоянию автомата.

5.5.2. Составление схемы простейшего автомата

Для лучшего уяснения положений теории последовательностных автоматов разберем пример составления схемы такого автомата в релейно-контактном и электронном вариантах.

Пример 5.2. Составление схемы регулирования уровня рабочей жидкости в резервуаре в пределах от нижнего уровня, контролируемого

поплавковым датчиком d_n , до верхнего уровня, контролируемого аналогичным датчиком d_e .

Нормальным состоянием датчиков будет считаться такое, когда заданный уровень достигнут и контакты датчиков разомкнуты ($d_n=0$, $d_e=0$). Пока не достигнут верхний уровень, т.е. пока $d_e=1$, насос H , питающий резервуар, должен быть включен ($H=1$). При достижении верхнего уровня, когда $d_e=0$, насос должен отключаться и оставаться отключенным ($H=0$), пока жидкость не опустится ниже нижнего контролируемого уровня, т.е. пока не замкнутся контакты нижнего датчика ($d_n=1$).

Технологический цикл регулирования уровня жидкости может отображен циклограммой, представленной в табл.5.7.

Таблица 5.7.

Циклограмма регулирования уровня жидкости

Элементы	Такты	1	2	3	4
Датчик	d_n	1	0	0	0
Датчик	d_e	1	1	0	1
Насос	H	1	1	0	0
Состояния		1		2	

Когда оба датчика сигнализируют о том, что заданный уровень не достигнут ($d_n=1$ и $d_e=1$), насос включен ($H=1$). Когда оба датчика сигнализируют о том, что заданный уровень достигнут ($d_n=0$ и $d_e=0$), насос выключен ($H=0$). Но когда нижний датчик сигнализирует о том, что заданный уровень достигнут ($d_n=0$), а верхний – что уровень не достигнут ($d_e=1$), то в такте 2 насос должен оставаться включенным, а в такте 4 – выключиться. Для различения этих двух случаев, обозначаем один из них состоянием 1, а второй – состоянием 2. В системе управления насосом для различения состояний 1 и 2 можно было бы ввести некую функцию Z , но поскольку она совпадает с функцией H включения насоса (см.табл.5.7), полагаем $Z=H$.

Составим таблицу истинности для выхода H^t комбинационной части (АЛУ) проектируемого устройства регулирования уровня (табл.5.8). В данном случае, поскольку $Z=H$, составленная таблица выходов, отображающая функцию выхода H^t , совпадает с таблицей переходов, отображающей функцию переходов насоса из одного состояния в другое.

В табл.5.8 H^{t-1} означает значение H в предыдущем такте работы проектируемого устройства, а значение H^t – это то значение, которое должно формироваться в текущем такте. Если $H^t=H^{t-1}$, то данное состояние устойчиво, а если $H^t \neq H^{t-1}$, то данное состояние не устойчиво и перейдет в другое состояние, соответствующее тому же сочетанию входных сигналов, но другому значению H . Так, состояние, записанное во второй строке (сверху) табл.5.8, имеет $H^{t-1}=1$, а $H^t=0$. Оно переходит в состояние, записанное в первой, верхней строке, где $H^{t-1}=H^t=0$. Это

означает, что когда при работающем насосе ($H^{t-1}=1$) будет достигнут верхний уровень заполнения резервуара ($d_n=0$ и $d_v=0$), система управления должна выключить насос ($H^t=0$). Если в этой ситуации насос окажется выключенным ($H^{t-1}=0$, $d_n=0$, $d_v=0$), то он и далее должен оставаться выключенным ($H^t=0$).

Таблица 5.8

Таблица истинности для выхода H^t

Входы			Выход
d_n	d_v	H^{t-1}	H^t
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	*(0)
1	0	1	*(0)
1	1	0	1
1	1	1	1

При составлении таблицы перебирались все сочетания значений d_n , d_v и H^{t-1} от 000 до 111 в порядке счета в двоичном арифметическом коде (см. §2.3) и определялось, каково должно быть дальнейшее состояние H^t насоса при данной комбинации сигналов датчиков и текущем состоянии H^{t-1} насоса. Так, если рассмотрим предпоследнюю строчку таблицы истинности (1101), то увидим, что в этой ситуации сигналы датчиков ($d_n=1$ $d_v=1$) свидетельствуют о том, что уровень жидкости в резервуаре снизился ниже допустимого минимума, а насос отключен ($H^{t-1}=0$). Следовательно, система управления должна его включить, о чем свидетельствует последующий выходной сигнал $H^t=1$.

В пятой и шестой строках табл.5.8 на месте значений H^t проставлены звездочки. Они свидетельствуют о невозможности данной ситуации ($d_n=1$ и $d_v=0$) при нормальном функционировании системы регулирования. Ведь если действительно $d_n=1$, то уровень жидкости в резервуаре ниже допустимого минимума, так что и верхний датчик должен выдавать сигнал $d_v=1$. Если же сигнал $d_v=0$ верен, то уровень жидкости выше верхнего допустимого максимума, и нижний датчик тем более должен выдавать сигнал $d_n=0$. Следовательно, при $d_n=1$ и $d_v=0$ один из датчиков явно неисправен и система регулирования не может нормально функционировать. В этой ситуации естественным решением является блокирование автоматической системы регулирования и переход к ручному управлению наполнением резервуара. Для этого следует предусмотреть отключение насоса при появлении неправильной комбинации входных сигналов, т.е. в пятой и шестой строках табл.5.8 проставить $H^t=0$. Если неисправен датчик d_v , т.е. всегда $d_v=0$, то при появлении такой неисправности система регулирования уровня, однажды отключив насос, уже не сможет его включить. Если неисправен

датчик d_n , т.е. всегда $d_n=1$, то система регулирования сможет включать насос при $d_n=1$ и $d_6=1$, но сигнал на включение будет возникать всякий раз, когда контролируемый уровень жидкости окажется меньше максимально возможного контролируемого датчиком d_6 . При этом частота цикла включений – отключений окажется чрезмерно большой, вынуждая к переходу на ручное управление.

После окончательного оформления таблицы выходов (табл.5.8) необходимо составить логическую формулу системы регулирования в дизъюнктивной форме, упростив ее методом Квайна-Мак-Класки (см. §5.2). В результате получим

$$H^t = d_n d_6 + d_6 H^{t-1}. \quad (5.10)$$

Для реализации данной функции на электромагнитных реле выгодно применить скобочную форму

$$H^t = d_6 (d_n + H^{t-1}).$$

Ей соответствует (см. §5.3) схема, приведенная на рис.5.9а.

Чтобы реализовать выражение (5.10) на логических элементах, надо взять от него двойную инверсию, причем внутреннюю инверсию следует раскрыть по правилу Де Моргана:

$$H^t = \overline{\overline{d_n} \overline{d_6} \cdot \overline{d_6} H^{t-1}}.$$

Теперь формула для H^t выражена исключительно через И-НЕ и по ней можно (см. §5.4) непосредственно составить схему на логических элементах И-НЕ, приведенную на рис.5.9б.

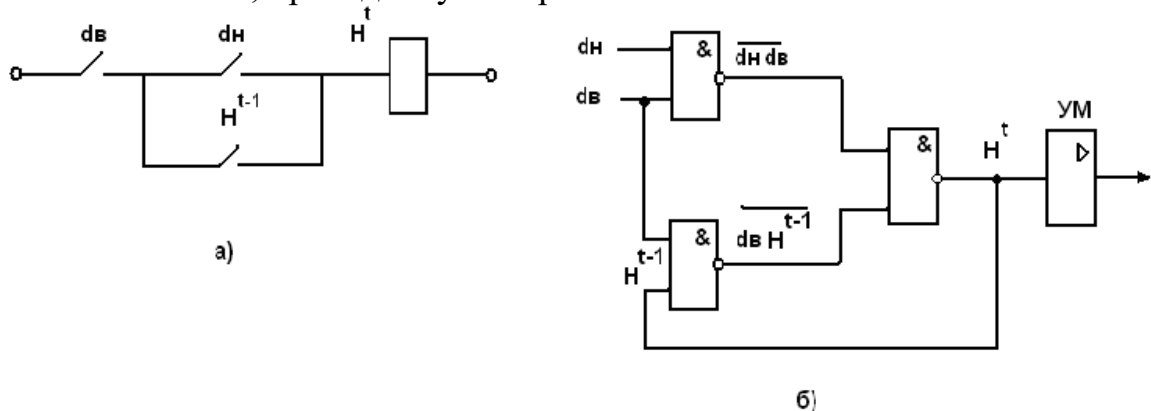


Рис.5.9. Схемы регулирования уровня жидкости в резервуаре:
а) на электромагнитных реле; б) на логических элементах И-НЕ.

На выходе релейной схемы установлено промежуточное реле H^t , а на выходе схемы на логических элементах усилитель мощности УМ. Как реле H^t , так и усилитель УМ необходимы для усиления управляющего сигнала, включающего контактор управления двигателем насоса. Но реле H^t кроме того подает сигнал обратной связи в виде своих контактов H^{t-1} , включенных в цепь управления этим реле. Срабатывание контактов реле происходит с задержкой, определяемой временем включения и отключения реле. Поэтому состояние контактов H^{t-1} отстает на один такт от состояния сигнала на обмотке реле H^t . Аналогично действует обратная связь в схеме на логических элементах. Скажем, когда

сформируются единичные входные сигналы $d_n=1$ и $d_6=1$, в этот момент времени на вход H^{t-1} еще подан сигнал $H^t=0$, а не $H^t=1$, который соответствует, согласно выражению (5.10), входным сигналам $d_n=1$ и $d_6=1$. Сигнал $H^t=1$ появится лишь после прохождения управляющего сигнала, обусловленного значениями $d_n=1$ и $d_6=1$, через два элемента И-НЕ (см.рис.5.9б). Теперь, если сравнивать схемы рис.5.9 с общей структурой последовательностного автомата, приведенной на рис.5.8, то окажется, что релейно-контактная структура рис.5.9а и вся логическая схема рис.5.9б, кроме проводника H^t-H^{t-1} входят в состав АЛУ, комбинационной части автомата. На долю УФС остаются: электромеханическая часть реле H^t , управляющая положением контактов H^{t-1} , и проводник H^t-H^{t-1} в логической схеме рис.5.9б. Однако этого достаточно, чтобы обеспечить запоминание каждого из двух состояний насоса, включенного и отключенного, и четкий переход из одного состояния в другое.

Простота АЛУ, которое реализует в данном случае только логические операции, и УФС, которое просто передает с элементарной задержкой сигнал о состоянии автомата и обеспечивает запоминание этого сигнала, не должны нас смущать. Несмотря на указанную простоту, структура рассмотренной схемы регулирования уровня жидкости в резервуаре вполне соответствует структуре последовательностного автомата, приведенной на рис.5.8.

Обратим также внимание на экономичность релейно-контактного варианта по сравнению с электронным. Ведь реле H^t может быть смонтировано на той же панели, что и силовой контактор, и питается от того же источника питания. А электронная схема требует применения особой монтажной платы и особого источника питания, да и стоимость электронных компонентов схемы может оказаться выше стоимости реле H^t . Указанные факторы определяют экономичность несложных релейно-контактных схем управления по сравнению с электронными схемами при небольшой частоте переключений.

5.6. Реализация алгоритмов управления последовательностных автоматов

5.6.1. Виды запоминающих устройств

Основной отличительной чертой последовательностных схем управления является, как мы уже убедились, наличие в их составе запоминающих устройств (ЗУ). Последовательностные автоматы называют поэтому *автоматами с памятью*. Все реальные устройства автоматического управления содержат в своем составе элементы памяти, пусть хотя бы в виде блок-контакта в схеме управления двигателем (см.рис.5.9а). Рассмотрим типовые ЗУ, применяемые в АСУТП.

По своему назначению ЗУ делятся на два класса: постоянные ЗУ (ПЗУ) и оперативные ЗУ (ОЗУ). ПЗУ предназначены для длительного хранения информации и потому их основным качеством является энергонезависимость, т.е. способность длительное время сохранять записанную информацию после отключения электропитания. По своей конструкции ПЗУ делятся на магнитомеханические и электронные. Магнитомеханические это жесткие магнитные диски и гибкие дискеты, информация записывается на их дорожках в виде последовательности намагниченных и ненамагниченных участков в двоичном коде, считая, что намагниченный участок это 1, а ненамагниченный – 0. Магнитомеханические ПЗУ энергонезависимы и надежны, допускают перезапись информации, но их быстродействие ограничено скоростью вращения дисков. Они удобны для употребления в тех случаях, когда перезапись информации и обращение за информацией к ЗУ происходит относительно редко, как это бывает, например, при записи массива УП в памяти УЧПУ. Электронные ПЗУ, в том числе перепрограммируемые ПЗУ (ППЗУ), выполняются на базе БИС. Они обладают большей скоростью обмена информацией и употребляются для формирования в УВМ системного программного обеспечения (см. §6.1), а ППЗУ – также и для записи УП. Однако электронные ПЗУ имеют относительно меньшую информационную емкость и транспортабельность.

Оперативные ЗУ предназначены для скоростной обработки текущей (оперативной) информации. Эта информация не предназначена для длительного хранения, но должна быть обработана достаточно быстро, чтобы обеспечить управление быстродействующими исполнительными механизмами, электроприводами. ОЗУ выполняются на энергозависимых БИС, но позволяют обрабатывать информацию с тактовой частотой в десятки и сотни мегагерц. Основой БИС ОЗУ являются триггерные ячейки, выполненные на электронных элементах с двумя устойчивыми состояниями, триггерах.

5.6.2. Триггеры

Простейшей триггерной ячейкой является асинхронный RS-триггер. Эта ячейка с двумя входами, R и S , и с двумя выходами: прямым выходом Q и инверсным выходом $\bar{Q}$. По сигналу $S = 1$ (*Set* – установка) RS-триггер устанавливается в единичное состояние, которому соответствует $Q = 1, \bar{Q} = 0$. По сигналу $R = 1$ (*Reset* – сброс) RS-триггер сбрасывается в нулевое состояние, которому соответствует $Q = 0, \bar{Q} = 1$. Когда на одном из входов RS-триггера имеется единичный сигнал, на другой вход должен быть подан нулевой сигнал. Одновременная подача единичных сигналов на оба входа RS-триггера не допускается. При наличии на обоих входах RS-триггера нулевых сигналов, его состояние не изменяется. Таким образом RS-триггер является последовательностной управляющей ячейкой, характеризующейся двумя

различными внутренними состояниями, каждое из которых целесообразно обозначать сигналом на прямом выходе: $Q=0$ и $Q=1$. В такую ячейку можно записать и в ней хранить информацию, не превышающую 1 бит (см. §2.2). Триггер – это стандартная ячейка ОЗУ для хранения информации в 1 бит.

Функционирование RS-триггера может быть отображено такой таблицей выходов (табл.5.9).

Таблица 5.9

Таблица выходов RS-триггера

R^t	S^t	Q^{t-1}	Q^t	$\overline{Q}^t$	Режим
0 0	0 0	0 1	0 1	1 0	Хранение информации $Q^t=Q^{t-1}$
0 0	1 1	0 1	1 1	0 0	Запись 1 $Q^t=1$
1 1	0 0	0 1	0 0	1 1	Запись 0 $Q^t=0$
1 1	1 1	0 1	*(1) *(1)	*(1) *(1)	Запретная комбинация $\overline{Q}^t=Q^t$

В табл.5.9 видно, что при $R=0$ и $S=0$ RS-триггер сохраняет свое состояние неизменным, т.е. хранит ранее записанную информацию: либо $Q=0$, либо $Q=1$. При подаче сигнала $S^t=1$ на выходе устанавливается $Q^t=1$, вместо $Q^t=0$ или сохраняется $Q=1$. Аналогично при подаче $R^t=1$ на выходе устанавливается $Q^t=0$. Запретность комбинации $R=1$ и $S=1$ (одновременно) означает, что клетки со звездочками можно заполнять произвольно, так как при нормальной эксплуатации RS-триггера одновременное поступление $R=1$ и $S=1$ исключено. При реализации RS-триггера на элементах ИЛИ-НЕ наиболее простая схема получается, если вместо звездочек проставить нули, а при реализации на элементах И-НЕ из тех же соображений лучше поставить вместо всех звездочек единицы, что и сделано. Составление по таблице 5.9 логических формул (см. §5.2) для функций Q^t и $\overline{Q}^t$ приводит к следующим выражениям:

$$Q^t = S + \overline{R}Q^{t-1} = \overline{\overline{S} \overline{\overline{R}Q^{t-1}}} \quad (5.11)$$

$$\overline{Q}^t = R + \overline{S}\overline{Q}^{t-1} = \overline{\overline{R} \overline{\overline{S}\overline{Q}^{t-1}}} \quad (5.12)$$

Вторые части формул (5.11) и (5.12) позволяют непосредственно по ним получить схему RS-триггера на элементах И-НЕ, приведенную на рис.5.10. Действительно, на выходе $\overline{Q}^t$ имеем:

$$\overline{Q}^t = \overline{\overline{R}Q^{t-1}},$$

а на выходе Q^t соответственно получим

$$Q^t = \overline{\overline{S}\overline{Q}^t} = \overline{\overline{S} \overline{\overline{R}Q^{t-1}}},$$

что соответствует выражению (5.11).

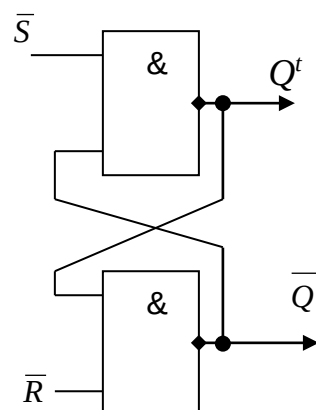


Рис.5.10. Схема RS-триггера на элементах И-НЕ

Аналогично проверяется правильность выражения (5.12) относительно выхода $\bar{Q}^t$ схемы рис.5.10.

Если в формуле (5.11) положить $Q=H$, $S=d_H d_B$, а $\bar{R} = d_B$ то она становится эквивалентной формуле (5.10). Таким образом, если на входе $\bar{S}$ триггера по схеме рис.5.12 поместить элемент И-НЕ, дающий $\bar{S} = \overline{d_H d_B}$, а на вход $\bar{R}$ подать сигнал d_B , то функции схем на рис.5.9б и 5.10 совпадут. Следовательно, схему регулирования уровня по формуле (5.10) можно строить на логическом элементе, реализующем функции RS-триггера, с добавлением элемента И-НЕ для реализации дополнительной логики по входу $\bar{S}$. Условие недопущения одновременной подачи сигналов $R=1$ и $S=1$ здесь строго соблюдается.

Чтобы получить синхронный RS-триггер, нужно на его входах предусмотреть две ячейки И-НЕ, на которые подается синхронизирующий импульс C (см.рис.5.11).

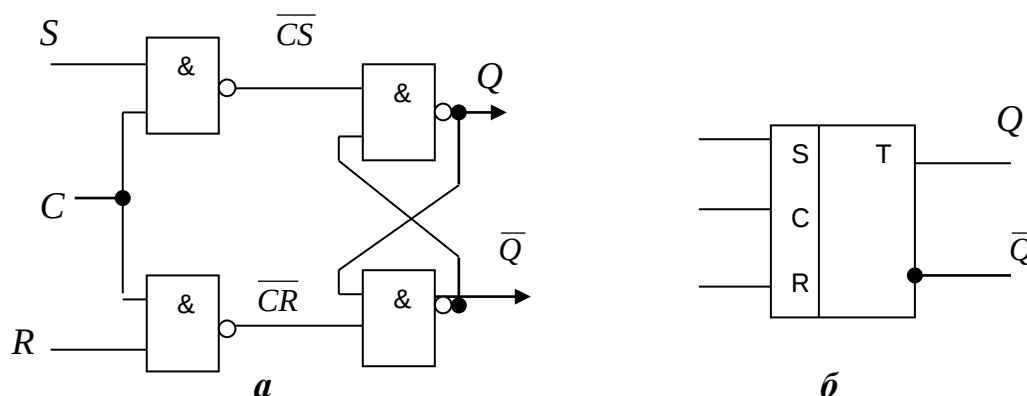


Рис.5.11. Синхронный RS-триггер:

а) схема на элементах И-НЕ; б) условное обозначение.

Синхронный триггер переключается только при одновременной подаче единичных сигналов на один из информационных входов и на вход C (*Clock*). Так, при $C=1$ и $S=1$ имеем $\overline{CS}=0$ и триггер переключается в состояние $Q=1$, $\overline{Q}=0$ или остается в этом состоянии (см.табл.5.9).

Путем подачи внутреннего сигнала $\overline{CS}$ на вход R синхронный RS-триггер превращается в триггер задержки, D -триггер (*Delay* –задержка). У D -триггера вход R является внутренним, а информационный вход S переименован в D . Состояние D -триггера принимает то значение, какое имеется на входе D : при $D=1$ получим $Q=1$, а при $D=0$ получим $Q=0$, но только при условии, что $C=1$. При $C=0$ состояние выходов D -триггера не изменяется. Это свойство D -триггера можно использовать для организации считывания информации со входов управляющего устройства. Например, можно установить D -триггеры на входе преобразователя кода (рис.5.7а) вместо элементов И-НЕ и подать входные сигналы на D -входы триггеров (см.рис.5.12). На их выходах будут получены как прямые, так и инверсные значения входных сигналов, какие они были в момент считывания информации с датчика положения, т.е. при сигнале синхронизации $C2=1$. Здесь совокупность D -триггеров играет роль регистра, в котором при $C2=0$ хранится записанная с датчиков при $C2=1$ информация.

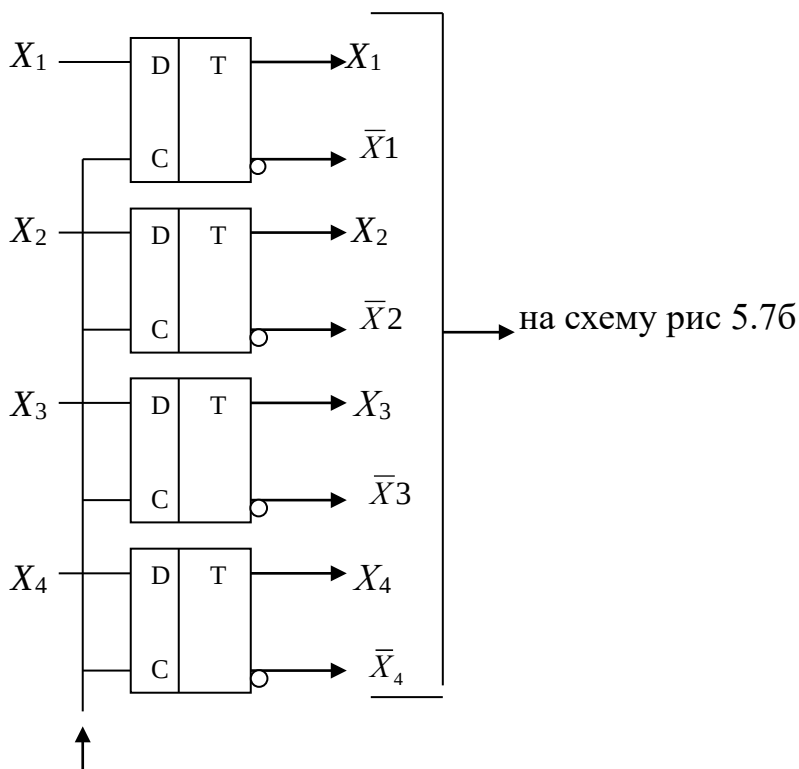


Рис.5.12. Синхронизация поступления входных сигналов $X_1 \dots X_4$

Будем полагать, что во время обработки информации управляющим устройством сигнал $C1$ (см.рис.5.7б) равен 1, а сигнал $C2$ равен нулю, что обеспечивает стабильность получаемой от D -триггеров информации. Напротив, во время считывания входной информации имеем $C1=0$, а $C2=1$. Обработка считываемой информации в это время не производится. После того, как сигнал $C2$ становится равным 0, дается небольшая выдержка времени на завершение переходных процессов в преобразователе кода, когда $C2$ также равен нулю. Благодаря этому, в период обработки считанной информации (при $C1=1$) информация на выходе преобразователя кода остается неизменной, эффект гонки импульсов полностью устраняется.

Заметим, что при синхронизации работы входных преобразователей управляющее устройство обеспечивает им работу в трех состояниях:

- 1) считывание поступившей на его входы информации ($C1=0$, $C2=1$);
- 2) преобразование поступившей информации входным преобразователем ($C1=0$, $C2=0$);
- 3) хранение поступившей информации ($C1=1$, $C2=0$).

В результате, при работе в синхронном режиме, входной преобразователь приобретает черты устройства последовательностного действия. Например, преобразователь кода по схеме рис.5.7 оказался снабженным собственным запоминающим устройством в виде комплекта D -триггеров (рис.5.12).

Из других типов триггеров для нас наиболее интересны счетный триггер (T -триггер) и JK -триггер (см.рис.5.13). T -триггеры лежат в основе схем различных счетчиков, а JK -триггеры являются универсальными в том смысле, что могут выполнять функции триггеров многих других типов. У T -триггера имеется только один информационный (счетный) вход T , который совпадает с тактовым входом C . После прохождения единичного импульса на счетном входе T состояние триггера изменяется на противоположное. Поскольку входные импульсы (такты) могут быть различной длительности, то T -триггер должен запоминать не только то состояние (либо $Q^t=1$, либо $Q^t=0$), в котором он должен оказаться по окончании текущего такта, но и предыдущее состояние (либо $Q^{t-1}=0$, либо $Q^{t-1}=1$) на все время подачи сигнала $T=1$. Иначе вновь сформировавшееся состояние будет восприниматься триггерной структурой (при условии сохранения $T=1$), как «старое», что вызовет новое, «лишнее» переключение (см.табл.5.11), за ним следующее и т.д. до тех пор, пока сохраняется $T=1$. Таким образом, необходимо запоминать «старое» значение Q на все время, пока

имеет место $T=1$, чтобы избежать автоколебаний. Для этого схема T -триггера должна содержать две ячейки памяти, а не одну, как это было у рассмотренных выше RS - и D -триггеров.

Функционирование T -триггера моделирует счет в двоичной системе исчисления. Действительно, при подаче единичного импульса на его вход ($T=1$) при условии, что значение выхода в этот момент времени равно нулю ($Q=0$), значение выхода становится равным единице ($Q=1$). Это соответствует добавлению единицы в какой-либо разряд двоичного числа, значение которого до того было равно нулю. При подаче следующего единичного импульса $T=1$ значение выхода триггера изменится от $Q=1$ к $Q=0$. То же самое происходит при прибавлении единицы в какой-либо разряд двоичного числа, если его значение до того было равно единице. Однако в двоичном числе при этом произойдет перенос единицы в следующий, старший разряд. Такой же перенос единицы произойдет и на выходе счетного триггера, если к нему подключить вход следующего счетного триггера, как это делается в счетчиках (см.рис.5.16).

Универсальность JK -триггера заключается в том, что при подаче входных сигналов $J=1$ или $K=1$ отдельно, он работает, как RS -триггер. Если же на вход K подавать инвертированный сигнал со входа J , то получится D -триггер. А при подаче $J=K=1$ одновременно, он работает, как T -триггер, т.е. изменяет свое состояние на противоположное после каждого прохождения единичного импульса на тактовом входе C . Следовательно, структура JK -триггера должна быть такой же, как и у T -триггера, но должны быть предусмотрены отдельные информационные входы J и K . Этим требованиям удовлетворяет схема, приведенная на рис. 5.13а, которой соответствует (при $J=K=1$) следующая таблица выходов (табл.5.10). Согласно табл.5.10 при $T=1$ изменяются только значения промежуточных выходов Q^t , а значения основных выходов Q^{t+1} остаются неизменными ($Q^{t+1}=Q^{t-1}$). При $T=0$ основные выходы принимают значение промежуточных: $Q^{t+1}=Q^t$, а значения Q^t при этом остаются неизменными. Функционирование основных выходов T -триггера описывается формулой

$$Q^{t+1} = Q^{t-1}T^t + Q^t\bar{T}^t \quad (5.13)$$

полученной из таблицы выходов (5.10). Функционирование промежуточных выходов T -триггера соответствует таблице переходов (5.11) и описывается формулой

$$Q^t = Q^{t-1}\bar{T} + \bar{Q}^{t-1}T \quad (5.13^1)$$

Таблица 5.11

Таблица переходов T -триггера

Таблица 5.10

Таблица выходов T -триггера

T^t	Q^{t-1}	Q^t	Q^{t+1}
-------	-----------	-------	-----------

0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

T^t	Q^{t-1}	Q^t
0	0	0
1	0	1
0	1	1
1	1	0

Автоколебаний в схеме по рис.5.13а не возникает, так как сигналы Q^{t-1} поступают не с выходов Q^t , а с выходов Q^{t+1} . При переключении триггера Q^t (при $T=1$) состояние триггера Q^{t+1} не изменяется, так как его входные цепи заперты блокировочными сигналами, поступающими либо с элемента $D1$, либо с элемента $D2$ на элементы $D5$ и $D6$.

Схема рис.5.13а, имеющая в своем составе два последовательно соединенных триггера, носит название двухступенчатого триггера. Двухступенчатость триггера обозначается на схемах знаком ТТ (см. рис.5.13.б и в).

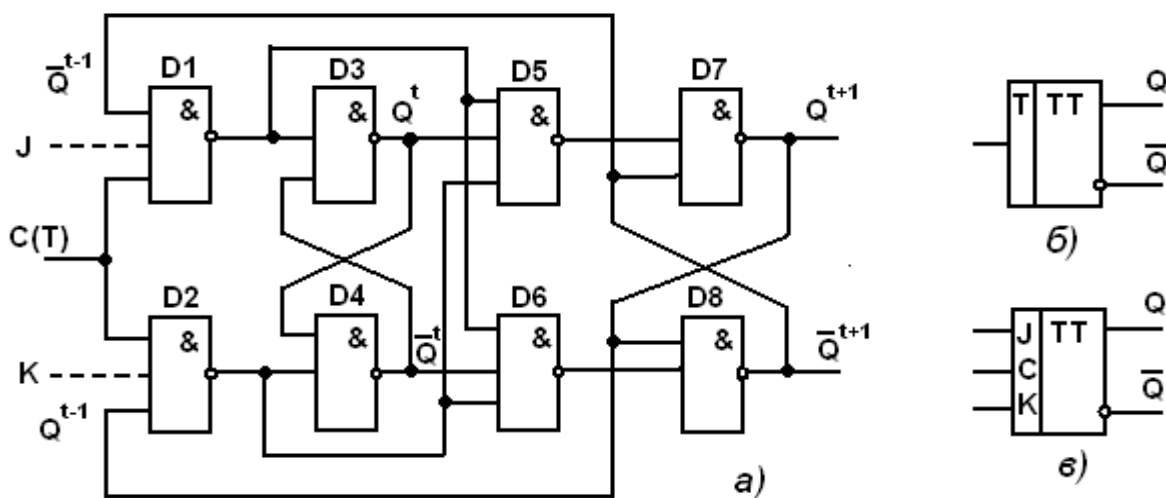


Рис.5.13. Схема(а) и условные обозначения Т-триггера(б) и ЖК-триггера(в)

5.6.3. Регистры

Микропроцессорные модули АСУТП нередко ведут обработку оперативной информации, исчисляющейся килобайтами. При этом применяются БИС электронной памяти, базирующиеся на триггерных ячейках. Емкость БИС ОЗУ может достигать нескольких Мбайт, что предполагает наличие миллионов триггеров в составе одной БИС. Триггеры используемые в ОЗУ, применяются не по одному, а группами, называемыми регистрами. Регистр – это электронное устройство, базирующееся на совокупности триггерных ячеек и предназначенное для

хранения и преобразования помещенной в него информации, записанной в двоичном коде. Регистры различных типов являются непременной принадлежностью микропроцессоров, БИС памяти и многих других БИС. Объем информации, хранимой в регистре, составляет 1-4 байта. По характеру выполняемых операций регистры принято подразделять на регистры хранения, регистры сдвига и счетные регистры (счетчики).

Регистры хранения реализуют только одну, общую для регистров всех типов функцию хранения информации в двоичном коде. Типичные регистры хранения строятся на *D*-триггерах. Примером простейшего регистра хранения является совокупность *D*-триггеров в схеме, приведенной на рис.5.12. Другим примером может служить 8-битовый (байтовый) регистр *RG*, приведенный на рис.5.14. Регистр *RG* имеет 8 *D*-входов и 8 *Q*-выходов, усиленных усилителями мощности *A* (см.рис.5.14б). Запись информации в регистр со входов *D0-D7* производится при *STB*=1 (*STB* – стробирующий сигнал). Считывание информации с выходов *Q0-Q7* возможно лишь при разрешении на считывание сигналом $\overline{OE} = 0$. Запись информации во все триггерные ячейки регистра *RG* производится, хотя одновременно, но независимо друг от друга, что является характерной особенностью регистров хранения.

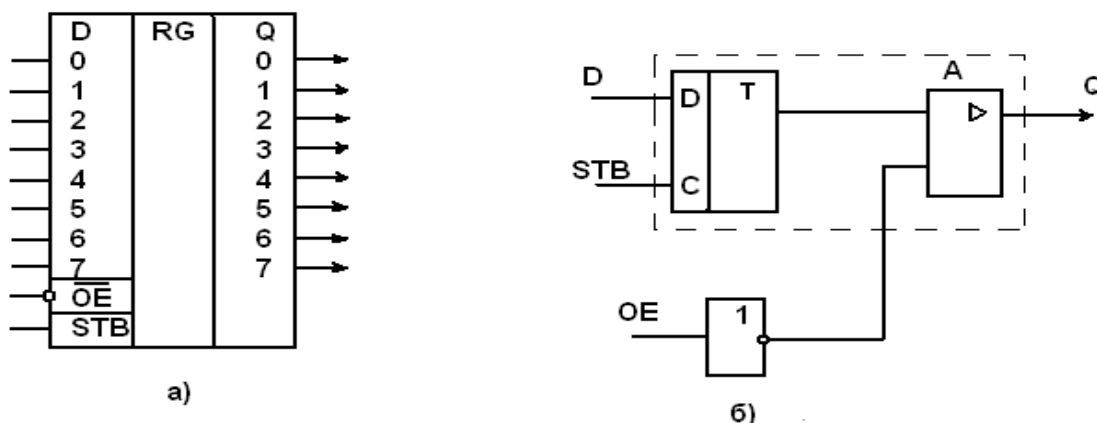


Рис.5.14. Регистр хранения, построенный на *D*-триггерах:

а) условное обозначение;

б) схема триггерной ячейки в составе регистра.

Регистры сдвига, помимо приема, хранения и выдачи информации подобно регистрам хранения, позволяют сдвигать записанную информацию, т.е. перемещать значения записанных битов информации от одной триггерной ячейки к другой, соседней, причем сдвиг информации производится одновременно во всех ячейках в одном направлении. На рис.5.15 приведена упрощенная схема регистра сдвига, по которой можно судить о том, как организуется сдвиг информации в регистрах, построенных на *D*-триггерах. Информация в такой регистр может поступать как в последовательном, так и в параллельном коде, а сдвиг информации производится подачей единичного импульса на вход

сдвига. Последний организуется путем объединения всех тактовых входов регистра. Поскольку все D -триггеры регистра соединены последовательно, а при подаче сигнала $C=1$ информация со входа D -триггера передается на его выход, то при подаче единичного импульса сдвига вся информация, записанная в триггерах регистра, сдвигается слева направо на один разряд. При этом информация, хранившаяся до подачи импульса сдвига на выходе $Q1$, будет утрачена, если ее не переписать в какое-либо другое ЗУ, включенное на выходе $Q1$. Информация, передаваемая в последовательном коде, подается в регистр сдвига через информационный вход D_4 . Предварительно регистр может быть очищен единичным импульсом, подаваемым на входы R установки нуля. При поступлении на вход D_4 информация подается поразрядно, начиная с младшего разряда. После подачи тактового импульса на вход сдвига информация переписывается из каждого разряда в соседний младший. Поразрядный сдвиг информации продолжается до тех пор, пока младший разряд слова информации не достигнет младшего разряда регистра, т.е. разряда $Q1$ (см.рис.5.15а). После этого информация может быть выдана через выходы Q в параллельном коде и в том же порядке, в каком она выдается в регистрах хранения. Таким образом, регистры сдвига преобразуют информацию из последовательного кода в параллельный, но хранят информацию они в параллельном коде. Преобразование информации из последовательного кода в параллельный необходимо при приеме информации из последовательного интерфейса, поскольку процессор, куда она далее поступает, обрабатывает информацию словами, записанными в параллельном коде.

Информация, записанная в регистре сдвига, может быть преобразована из параллельного кода в последовательный путем ее последовательного считывания с выхода $Q1$ (рис.5.15) после подачи каждого импульса сдвига. Предварительно она должна быть подана в параллельном коде на входы D в том же порядке, в каком она подается в регистры хранения (на рис.5.15 входы параллельного ввода информации не показаны). Преобразование информации из параллельного кода в последовательный необходимо при передаче ее через последовательный интерфейс. Один и тот же регистр сдвига может и принимать информацию из последовательного интерфейса, когда последний работает в режиме приема, и передавать ее через последовательный интерфейс, когда тот работает в режиме передачи.

Сдвиг информации в регистре сдвига может производиться как слева направо (см.рис.5.15а) так и справа налево. Перемещение информации в регистре на один разряд влево соответствует умножению на два, а перемещение вправо – делению на два. Поэтому регистры сдвига используются также для организации операций умножения и деления в процессорах.

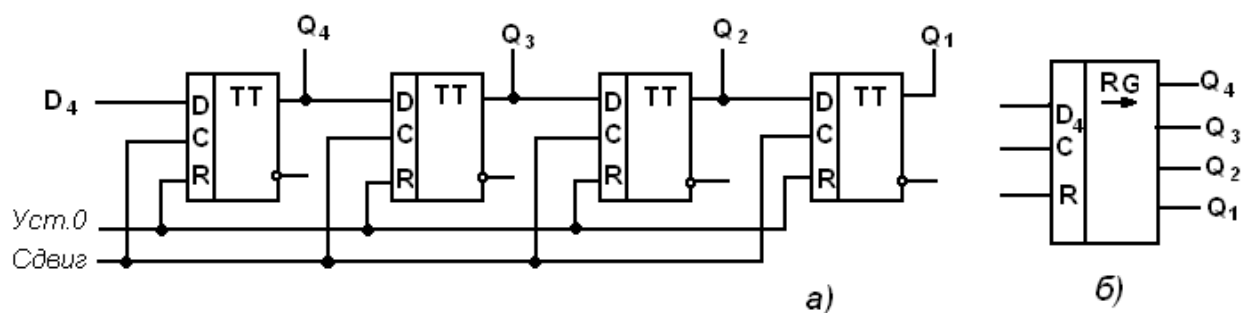


Рис.5.15. Схема регистра сдвига (а) и его условное обозначение (б)

Счетные регистры, или счетчики, отличаются тем, что помимо функций записи, хранения и выдачи информации выполняют функцию счета поступающих на них импульсов с запоминанием результатов. Счет импульсов может производиться как в привычной нам десятичной системе исчисления (декадные счетчики), так и в двоичной системе исчисления (двоичные счетчики). Последние являются основным типом счетчиков, поскольку декадные счетчики строятся на базе двоичных. Упрощенная схема двоичного счетчика приведена на рис.5.16. На схеме видно, что двоичные счетчики строятся на T -триггерах, соединенных последовательно так, что выход Q_i со входом T_{i+1} следующего триггера (схема прямого счета). Каждый триггер перебрасывается в единичное состояние по окончании (по заднему фронту) одного единичного импульса на его входе T_i и в нулевое состояние – по окончании следующего единичного входного импульса. Поэтому каждый последующий триггер перебрасывается в два раза реже предыдущего. Триггер TT_3 перебросятся в единичное состояние ($Q_3=1$) по прохождении четырех единичных импульсов на входе T_1 и вернется в нулевое состояние ($Q_3=0$) по прохождении восьми импульсов на том же входе. Тогда же вернутся в исходное нулевое состояние и остальные триггеры счетчика. Таким образом, счетчик, представленный на рис.5.16, может обеспечить счет до 8 импульсов, а состояние его триггеров будут соответствовать числам от нуля (000) до 7 (111) в двоичной системе исчисления. В целом же емкость двоичного счетчика составляет 2^n , если n – это количество счетных триггеров в его составе.

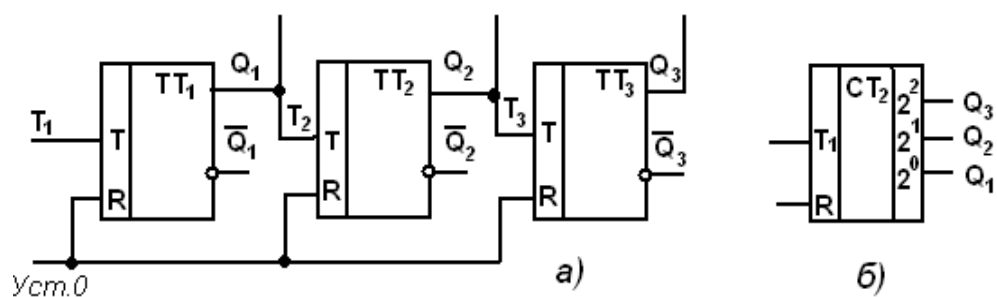


Рис.5.16. Трехразрядный двоичный счетчик: а – схема; б – условное обозначение

Счетчик, построенный по схеме рис.5.16, это счетчик прямого счета, или *суммирующий*. После прихода каждого входного импульса на вход Т1 число, записанное в таком счетчике, увеличивается на единицу. Если в счетчике соединить со входом каждого последующего триггера инверсный (а не прямой) выход предыдущего триггера, то получится счетчик обратного счета, или *вычитающий*. После прихода каждого входного импульса записанное в нем число уменьшается на единицу. С помощью логических микросхем, установленных между триггерами счетчика для изменения порядка прохождения управляющих импульсов от одного триггера к другому, можно получить *реверсивный* счетчик, который может работать как в режиме сложения, так и вычитания поступающих импульсов. Схема реверсивного счетчика приведена на рис.5.17. При поступлении счетных импульсов на вход $+1$ данного счетчика он работает как суммирующий, а при поступлении импульсов на вход -1 счетчик работает как вычитающий. На выходе счетчика схемами совпадения формируется сигнал переноса >15 единицы в младший разряд следующей микросхемы счетчика после заполнения единицами всех четырех разрядов изображенной на рис.5.17 микросхемы. Таким же образом формируется сигнал заёма <0 после обнуления счетчика при работе в режиме обратного счета.

Счетчики широко применяются для организации счета операций в рамках технологического цикла, счета циклов вычислений в порядке выполнения управляющих программ и при моделировании, для образования последовательностей адресов команд и адресов ЗУ при программировании микропроцессорных устройств. Особую группу счетчиков образуют *таймеры*, которые производят счет единиц поступающих в них сигналов времени вплоть до завершения отсчета заданного временного интервала. Таймеры являются аналогами электромагнитных реле времени. Примером таймера может служить вычитающий счетчик, в который УП засылает в параллельном коде, как во всякий регистр памяти, число, равное количеству единиц времени, соответствующему заданной выдержке времени. Затем начинается вычитание единиц времени, поступающих с течением времени на счетный вход данного счетчика. После того, как содержимое счетчика становится равным нулю, выдержка времени считается завершенной. А для организации последовательности адресов команд в регистр счетчика команд, имеющийся в составе микропроцессорного устройства, засылается адрес начальной команды. Затем, по мере выполнения очередной команды, содержимое счетчика команд увеличивается на единицу, и новую команду микропроцессор находит уже по новому адресу.

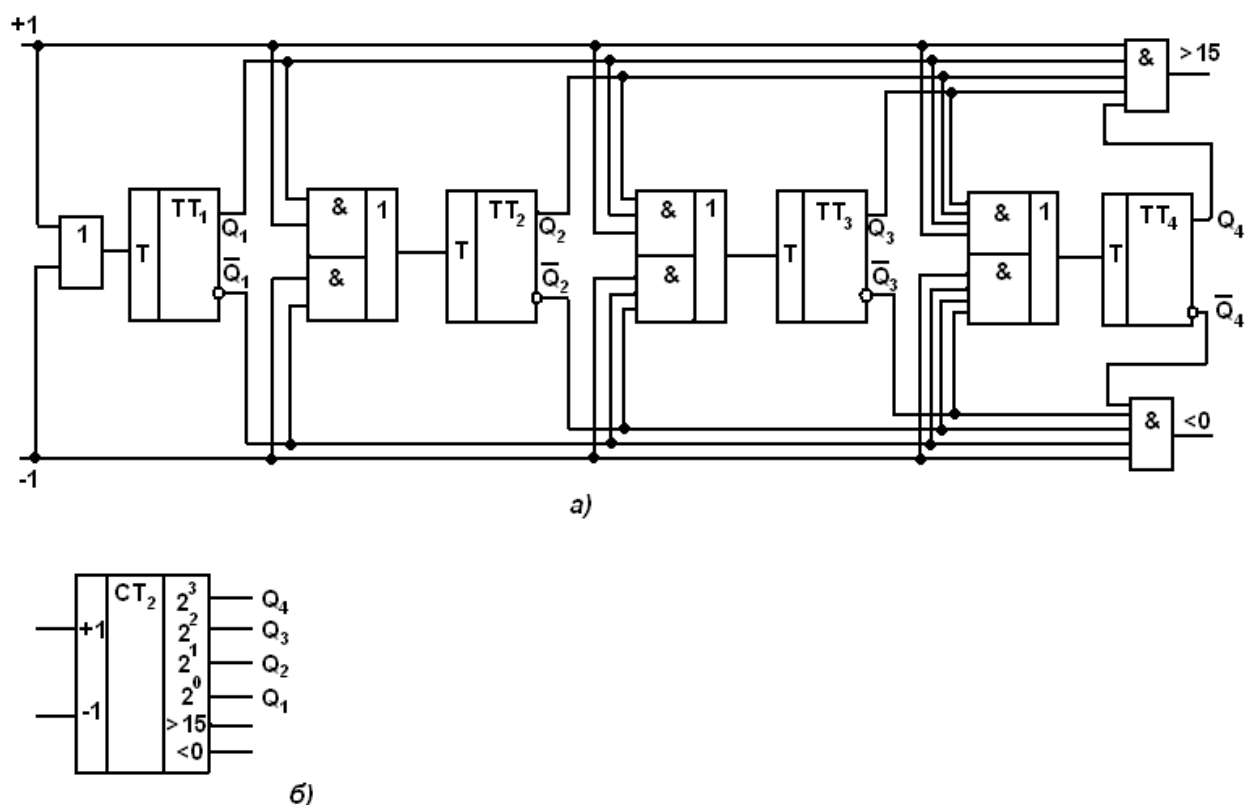


Рис.5.17. Четырехразрядный реверсивный счетчик: а – схема; б – условное обозначение

При рассмотрении различных регистровых схем (рис.5.14-5.17) мы убедились, что в их основе лежат простые цепочки триггеров, из которых, как из кирпичиков, складываются конструкции различных регистровых структур. Применяемые в регистрах типы триггеров, подробный анализ которых произведен в § 5.6.2, идеально приспособлены для применения именно в регистрах. По указанным причинам нет необходимости составлять таблицы переходов и таблицы выходов для рассмотренных регистровых схем, так как их структура непосредственно строится из триггеров и связующих простых логических цепей.

Полный логический анализ с использованием таблиц переходов и таблиц выходов следует применять лишь при необходимости уточненного анализа и синтеза новых, оригинальных конструкций управляющих устройств. При этом обычно не удастся сразу полностью предусмотреть весь необходимый набор состояний и переходов нового устройства, так что процесс проектирования идет от эвристической гипотезы к окончательному решению через ряд последовательных приближений. Нормальный процесс проектирования включает в себя широкое использование типовых конструкций логических элементов, арифметических и запоминающих устройств от простейших логических ячеек до микропроцессорных наборов.

5.6.4. Преобразователи кодов и арифметические устройства

Реальные последовательностные автоматы обязательно содержат в своем составе комбинационную часть – АЛУ (см.рис.5.8), которая производит преобразование информации, арифметическое и логическое, в соответствии с заданной программой. Проектирование АЛУ опирается на типовые решения отдельных узлов, выполняющих логические и арифметические операции. Простейшие логические ячейки, применяемые АЛУ, уже рассмотрены выше (см. §§5.3-5.4). Здесь мы сначала рассмотрим схемы узлов, через которые информация подается с входных устройств на регистры УФС для дальнейшего преобразования или посредством которых преобразуется информация, снимаемая с выходных регистров УФС для подачи на выходы автомата. Такие узлы называются *преобразователями кодов*. В §5.4 уже рассмотрена схема преобразователя кода Грея, получаемого от кодового датчика положения, в арифметический двоичный код, удобный для обработки в УВМ. Здесь мы еще рассмотрим схемы *дешифраторов*, которые преобразуют двоичный код выходных регистров УВМ в единичный сигнал на одном из выходов, номер которого задается кодом, поступающим на дешифратор. Иными словами, дешифратор определяет номер выхода на который необходимо подать единичный сигнал. Поэтому дешифратор называют преобразователем двоичного кода в единичный.

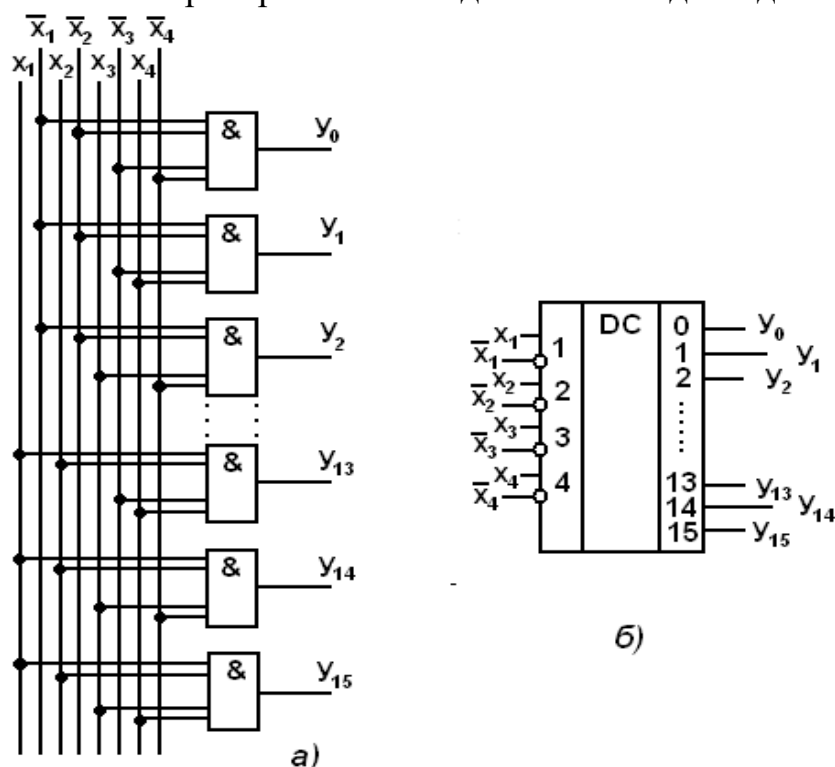


Рис.5.18. Схема (а) и условное обозначение (б) линейного дешифратора

Поскольку конъюнкция равна единице только на одном наборе логических переменных, то простейший дешифратор есть совокупность схем реализующих логическое умножение заданных кодовых сигналов. Такой дешифратор называется линейным. Схема линейного дешифратора и его условное обозначение приведены на рис.5.18. Нет никакой необходимости анализировать таблицу истинности линейного дешифратора, чтобы получить схему рис.5.18, так как его логические выражения могут быть получены непосредственно из заданных условий функционирования и не поддаются минимизации. Например, логическая функция выхода Y_2 на схеме рис.5.18 может быть непосредственно выражена в виде

$$Y_2 = \bar{X}_1 \bar{X}_2 X_3 \bar{X}_4$$

Однако схему дешифратора, приведенную на рис.5.18, можно все же несколько упростить, если разбить его входные сигналы на две группы: $X_1 - X_2$ и $X_3 - X_4$ и сначала произвести дешифрацию управляющих сигналов внутри данных групп. Логические формулы необходимых для этой цели двухвходовых дешифраторов и их реализацию на элементах И-НЕ предлагаем читателю проработать самостоятельно. Все выходные сигналы обоих двухвходовых дешифраторов необходимо попарно перемножить, как это сделано, например, в схеме, приведенной на рис.5.19. На выходах этой схемы будут получены требуемые сигналы $Y_0 - Y_{15}$.

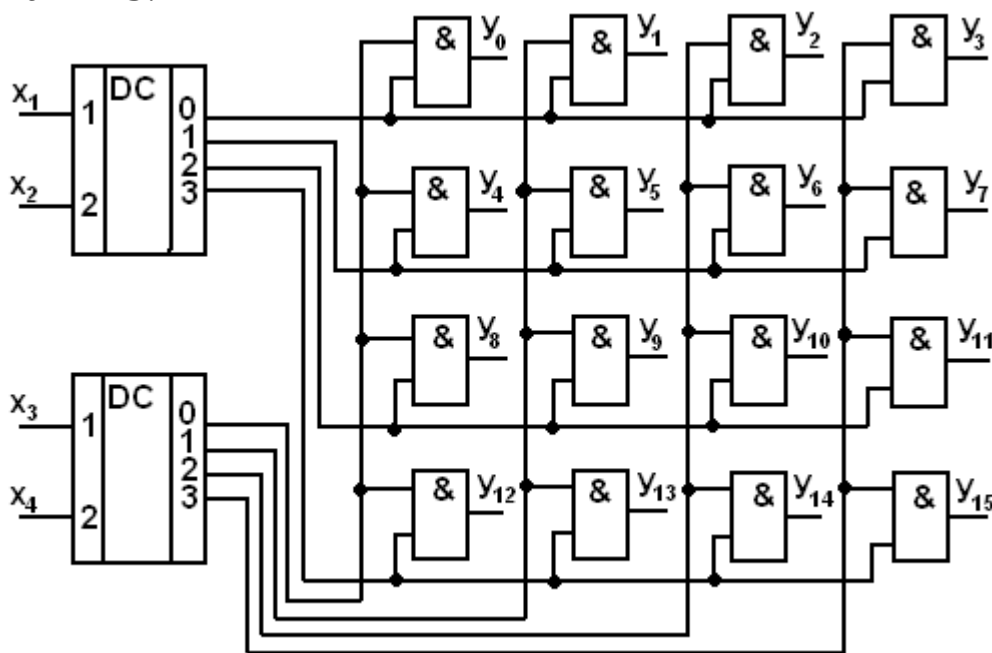


Рис. 5.19. Матричный дешифратор на 16 выходов

В качестве примера комбинационных арифметических устройств рассмотрим схему суммирования двоичных чисел. Каждый (i -тый) разряд такой схемы должен обеспечить показ единицы в трех случаях:

1) когда имеется перенос из младшего разряда, а слагаемые равны нулю

$(P_{i-1}=1)$;

2) когда одно из слагаемых равно единице ($A_i = 1$ или $B_i = 1$), а другое слагаемое равно нулю и нет переноса из младшего разряда ($P_{i-1} = 0$);

3) когда оба слагаемых равны единице и имеется перенос из младшего разряда ($A_i = 1, B_i = 1$ и $P_{i-1} = 1$).

В остальных случаях содержимое i -того разряда сумматора должно быть равно нулю. Очевидно, что этим условиям соответствует логическая формула

$$S_i = \bar{A}_i \bar{B}_i P_{i-1} + \bar{A}_i B_i \bar{P}_{i-1} + A_i \bar{B}_i \bar{P}_{i-1} + A_i B_i P_{i-1} \quad (5.14)$$

где S_i – значение i – того разряда сумматора.

Помимо формирования собственного значения каждый разряд (ячейка) сумматора должен формировать сигналы переноса в следующий, $(i+1)$ -й разряд. Условиями переноса в следующий разряд являются:

1) равенство единице обоих слагаемых одновременно ($A_i = 1$ и $B_i = 1$) независимо от значения P_{i-1} ;

2) равенство единице одного из слагаемых и сигнала переноса из младшего разряда ($A_i = 1$ или $B_i = 1$ и одновременно $P_{i-1} = 1$) независимо от значения другого слагаемого. Исходя из этих условий, получаем следующую формулу для сигнала переноса P_i :

$$P_i = A_i B_i + A_i P_{i-1} + B_i P_{i-1} \quad (5.15)$$

Схема ячейки сумматора, реализованная непосредственно по формулам (5.14) и (5.15) в базисе И-НЕ, приведена на рис.5.20. Если употребить скобочные формы формул (5.14) и (5.15):

$$a = \bar{A}_i B_i + A_i \bar{B}_i$$

$$S_i = a \bar{P}_{i-1} + \bar{a} P_{i-1}$$

$$P_i = A_i B_i + a P_{i-1}, \quad (5.16)$$

где функция a получена вынесением за скобки в формуле (5.14), то ячейку сумматора можно построить на однотипных двухвходовых логических схемах И-НЕ. Реализация ячейки сумматора будет проще, но быстродействие ее уменьшится в 2 раза, так как формирующиеся сигналы должны будут проходить через промежуточные цепи, реализующие функцию a . Заметим также, что условное обозначение, приведенное на рис.5.20б, предполагает, что инверсные значения A_i, B_i и P_{i-1} формируются внутри ячейки сумматора.

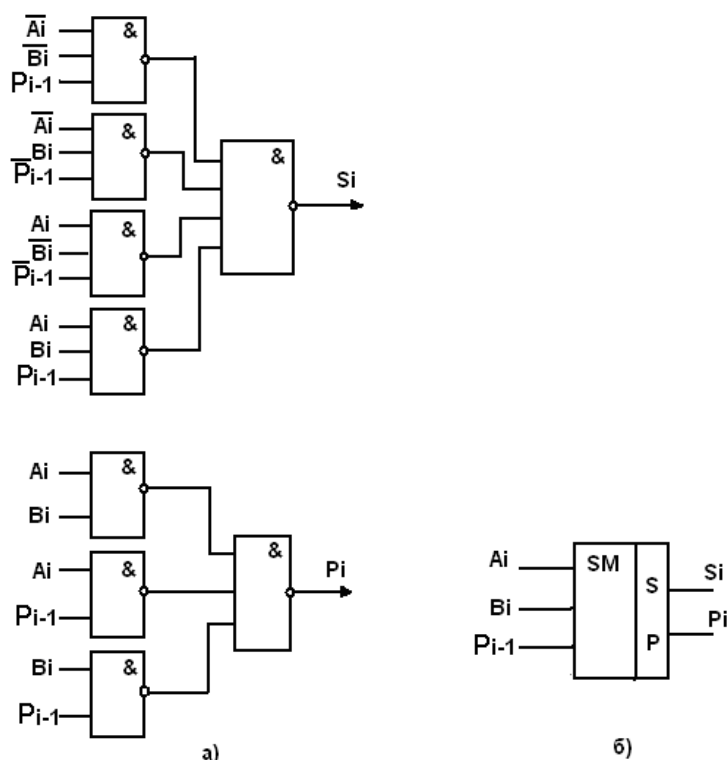


Рис. 5.20. Схема ячейки сумматора (а) и ее условное обозначение (б)

Схема сумматора, составленного из ячеек рис.5.20б, приведена на рис.5.21. Поскольку перенос в самый младший разряд сумматора не реален, то его вход переноса соединяется с общей точкой, что соответствует нулевому значению сигнала переноса. Данный сумматор является комбинационной схемой, а потому полученная сумма должна быть зафиксирована в каком-либо регистре, например – в аккумуляторе микропроцессора, управляющего вычислениями.

Сумматоры применяются и для выполнения операций вычитания (сложения отрицательных чисел), для чего последние должны быть представлены в дополнительном коде (см.§2.4.2). Что касается умножения и деления, то эти операции выполняются, как показано выше, с помощью операций сдвига, на регистрах.

Богатый выбор типовых решений как последовательностных, так и комбинационных устройств различного назначения приводит к тому, что при анализе и синтезе сложных последовательностных автоматов, таких как БИС микропроцессорных устройств управления, опираются именно на эти решения, а не на всеобъемлющий анализ состояний автомата. Кроме того сложные управляющие устройства характеризуются миллионами состояний, полностью рассмотреть которые без декомпозиции, т.е. разделения устройства на составляющие блоки, находящиеся в иерархической зависимости друг от друга, весьма затруднительно даже методами машинного анализа. При анализе управляющих устройств АСУТП мы будем, как и ранее,

последовательно производить их декомпозицию с учетом иерархической структуры АСУТП, а составлять управляющие программы будем опираясь на иерархию управленческих решений.

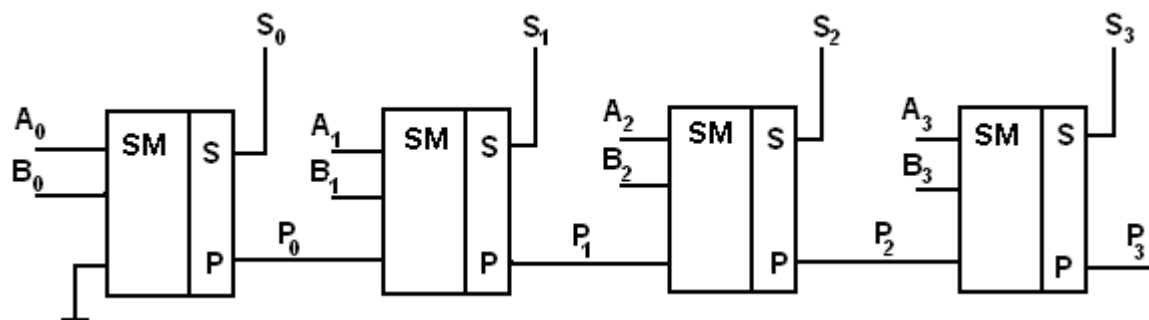


Рис.5.21. Схема четырехразрядного комбинационного сумматора

5.7. Обобщенные алгоритмы управления технологическим циклом

Сложные алгоритмы АСУТП, начиная с алгоритмов цифровых регуляторов параметров исполнительных устройств, опираются на типовые логические и арифметические процедуры, реализуемые процессорами управляющих микро-ЭВМ. Эти алгоритмы в конечном итоге воплощаются в УП, определяющих функционирование АСУТП. Порядок составления таких программ зависит от принятого языка программирования, но в любом случае в программе должны быть описаны все реализуемые ею арифметические и логические операции и даны точные указания относительно порядка их реализации, а также порядка поступления в УВМ входных и распределения выходных управляющих сигналов.

Предварительное описание действий УВМ по выполнению будущей УП может быть сделано обычным языком с применением соответствующей технической терминологии и условных обозначений. Однако подготовка программы значительно облегчается, если дополнительно составить алгоритм управления в виде особым образом подготовленной схемы.

Основными графическими элементами схемы алгоритмов являются прямоугольники и ромбы. Прямоугольники служат для отображения арифметических операций и операций с запоминающими устройствами, а ромбы служат для задания логических условий. Графические элементы алгоритма располагают по преимуществу сверху вниз в порядке выполнения действий, предусмотренных алгоритмом. Особо отмечают начало и конец алгоритма либо овалом, либо кружком. Затем графические элементы алгоритма соединяют стрелками, указывающими точную последовательность выполняемых операций и различные переходы, связанные с изменением порядка выполняемых операций.

В качестве примера рассмотрим составление алгоритма цифрового регулятора, действующего в замкнутом контуре управления

электроприводом или другим исполнительным устройством (см.рис.5.22).

Общая формула разностного уравнения для вычисления регулирующего воздействия U_p , формируемого регулятором, может быть представлена в таком виде:

$$U_p(\tau) = \sum_{i=0}^n b_{n-i} E(\tau-i) - \sum_{i=0}^{m+1} a_{m-i+1} U_p(\tau-i) ; a_{m+1}=0, \quad (5.17)$$

$$\text{где} \quad \tau = \frac{t}{T} \quad (5.18).$$

количество шагов квантования за время t работы регулятора;

T – такт, период дискретности цифровой системы управления (шаг квантования по времени):

$E(\tau)$ – текущее дискретное значение величины рассогласования на входе регулятора;

$E(\tau-i)$ – дискретное значение E за i шагов квантования назад;

$U_p(\tau)$ – текущее дискретное значение регулирующего воздействия;

$U_p(\tau-i)$ – дискретное значение U за i шагов квантования назад;

$a_0, a_1, \dots, a_m$ – коэффициенты знаменателя дискретной передаточной функции регулятора: $a_{m+1} = 0$ – фиктивный коэффициент, введенный для упрощения программы;

$b_0, b_1, \dots, b_n$ – коэффициенты числителя той же функции.

Все коэффициенты a_i и b_i являются функциями периода дискретности T , который обычно является параметром заданной цифровой системы управления.

Чтобы рассчитать текущее значение $U_p(\tau)$, необходимо, судя по формуле (5.17), предварительно рассчитать все произведения вида $b_{n-i} E(\tau-i)$ и – вида $a_{m-i+1} U_p(\tau-i)$, а затем их алгебраически сложить. Если расчет начинается с момента времени $\tau=0$, то должны быть заданы все начальные значения величин $U_p(\tau-i)$ и $E(\tau-i)$ и, конечно, все коэффициенты a_i и b_i . Задание этих величин в алгоритме, приведенном на рис.5.22, не показано, но оно подразумевается. Начало расчета помечено на рис.5.22 цифрой I, а конец расчета – цифрой II, обе в кружках. Расчет начинается с задания нового дискретного значения $E(\tau)$ задающей частью системы-управления, которой принадлежит данный регулятор. Поскольку все начальные условия расчета $U_p(\tau)$ учтены в формуле (5.17), то перед началом расчета прошлое значение $U_p(\tau)$ полагается равным нулю ($U_p(\tau)=0$). Обнуляется и переменная цикла расчетов ($i=0$). Это означает, что обнуляются ячейки ОЗУ, в которых хранятся текущее значение $U_p(\tau)$ и текущее значение переменной цикла. Первый сверху прямоугольник рассматриваемого алгоритма является, таким образом, заданием операций с ОЗУ. Далее в алгоритме (рис.5.22) организовано формирование $m+2$ произведений типа

$$U_a = -a_{m+1}U_p(\tau-1)$$

включая фиктивное $-a_{m+1}U_p(\tau)$ при $i=0$, которое является первым в цикле расчетов. Последнее произведение будет соответствовать $i = m + 1$ и даст $-a_0U_p(\tau-m-1)$. Дальнейшие расчеты произведений данного типа будут прекращены согласно записанному в верхнем ромбе условию $i > m + 1$. Если это условие соблюдается (вариант ДА), то совершается переход к расчету произведений типа

$$U_b = b_{n-i}E(\tau-i)$$

в обход расчета произведений первого типа. Такой переход называется *условным переходом* (переход в обход некоторой части программы в случае выполнения заданного условия). Расчет $n + 1$ произведений второго типа ограничен условием $i > n$, т.е. совершается до тех пор, пока имеет место $i < n$.

После каждого из расчетов заданных произведений полученное произведение суммируется в состав $U_p(\tau)$ по формуле

$$U_p(\tau) = U_p(\tau) + U_a$$

или по формуле

$$U_p(\tau) = U_p(\tau) + U_b.$$

Это означает, что каждый раз из ОЗУ извлекается накопленное значение $U_p(\tau)$ и к нему прибавляется либо U_a , либо U_b , пока не будут рассчитаны и суммированы в состав $U_p(\tau)$ все заданные произведения.

Если все произведения U_b подсчитаны, то по условию $i > n$ (вариант ДА) происходит переход к проверке, все ли расчеты U_a выполнены, через исполнение условия $i > m + 1$. Это необходимо в тех случаях, когда имеет место $m + 1 > n$. В таких случаях, хотя расчеты по условию $i > n$ будут завершены, условие $i > m + 1$ еще не будет выполняться, так что расчеты U_a нужно будет продолжить. Для продолжения расчетов значение i увеличивается на единицу блоком $i = i + 1$.

Выполнение одновременно условий $i > n$ и $i > m + 1$ (вариант А) свидетельствует об окончании расчетов текущего значения U_p , которое поступает на выход II (в кружке) и далее в систему управления силовой частью исполнительного устройства. В следующем такте T , после поступления нового значения рассогласования $E(\tau)$ начнется расчет нового значения U_p , а только что рассчитанное значение U_p станет прошлым, что выражается соотношением:

$$U_p(\tau-1) = U_p(\tau).$$

Последняя формула означает, что как только будет рассчитано значение $U_p(\tau)$ оно будет занесено в ячейку ОЗУ, где до того хранилось значение $U_p(\tau-1)$, то есть предыдущее значение U_p . Предварительно значение $U_p(\tau-i)$ переносится в ячейку ОЗУ, где хранилось значение $U_p(\tau-2)$ и т.д. Этот процесс называется сдвигом значений выходной

величины. Аналогично производится сдвиг входной величины E . Он продолжается до тех пор, пока не получит новое значение наибольшая по номеру из величин $U_p(\tau)$ или $E(\tau)$. Данные сдвиги отображены в самом нижнем прямоугольнике схемы алгоритма (рис.5.22).

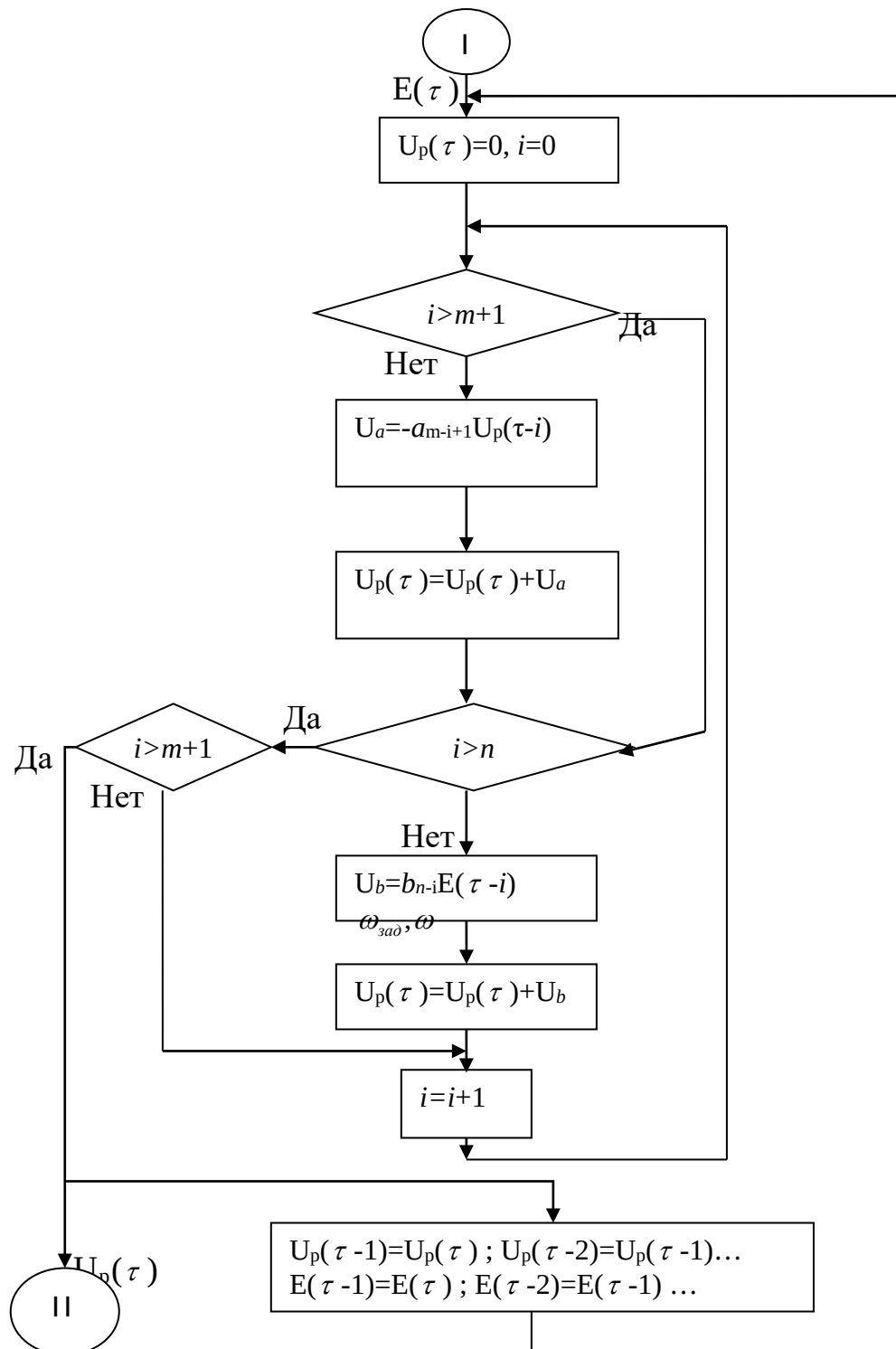


Рис. 5.22. Алгоритм цифрового регулятора

Время такта T обычно является жестко заданным (например, конструкцией исполнительного устройства) периодом дискретности, так что время выполнения расчетов величины $U_p(\tau)$ должно (с некоторым запасом) укладываться во время T . Отсюда возможны различные подходы к разработке алгоритма и управляющей программы регулятора. Если время расчетов достаточно мало, то алгоритм и программу можно разрабатывать исходя из условия ее минимальной длины (что и делалось), чтобы она занимала меньше места в ПЗУ. Если же время расчетов угрожающе велико, то программу разрабатывают исходя из требования обеспечения ее максимального быстродействия.

При рассмотрении системы автоматизации в целом обсуждать подробности алгоритмов отдельных регуляторов, входящих в ее состав, не представляется возможным. Схема общего алгоритма сложной системы обычно состоит из ряда прямоугольников, в которых размещаются названия разделов УП, и связывающих их ромбов основных логических условий функционирования системы. Такие алгоритмы необходимы как сжатое и удобное для обозрения описание системы автоматизации.

В качестве примера рассмотрим общий алгоритм микропроцессорной системы управления лифтом (рис.5.23). Функционирование системы управления в общих чертах становится понятным по надписям на схеме алгоритма. После включения в работу (блок 1 алгоритма) и в течение всего времени работы производится постоянный контроль цепи безопасности (ромб условного перехода 2). При размыкании этой цепи (вариант ДА) происходит аварийная остановка лифта (блок 3). Если кабина лифта остановилась на ловителях или от воздействия на конечные выключатели, то необходимо воспользоваться режимом деблокировки. В случае остановки по причине неисправности другого типа, поиск и устранение неисправностей (6) производится без вхождения в режим деблокировки. Ромбы 7, 8 и 9 определяют условия задания тех или иных режимов управления лифтом, реализуемых программными блоками 10, 11 и 12. При этом в режиме ревизии шахты лифта управление лифтом возможно только с поста ревизии на крыше кабины. Выполнение программы управления лифтом согласно алгоритму рис.5.23 продолжается до тех пор, пока не поступит команда остановки.

Для более подробного ознакомления с системой управления лифтом необходимо рассмотреть алгоритмы подпрограмм, входящих в состав общего алгоритма. Тогда окажется, что в них входят, в частности, алгоритмы управления режимами работы электропривода лифтового подъемника. И только при рассмотрении составов последних алгоритмов дело дойдет до рассмотрения алгоритмов функционирования регуляторов скорости и положения в составе цифровой системы управления электроприводом механизма подъема, аналогичных

рассмотренному выше цифровому регулятору. А в завершение должна быть рассмотрена реализация заданных алгоритмов управления средствами микропроцессорной техники.

При реализации заданных алгоритмов их иерархии приводится в соответствие иерархия управляющих микропроцессорных устройств, внутри которых с помощью системы приоритетов строится иерархия управляющих программ.



Контрольные вопросы к главе 5

1. Каковы особенности управления технологическим циклом?
2. В чем заключается различие между последовательностными и комбинационными автоматами?
3. Приведите примеры технической реализации дискретных двоичных сигналов.
4. Каковы основные этапы составления комбинационных схем управления?
5. Что такое таблица истинности и как ее составляют?
6. Как составляются логические выражения заданных логических (булевых) функций по таблице истинности?
7. Как производится упрощение логических выражений методом Квайна-Мак-Класки?
8. Как производится составление релейно-контактных схем с помощью логических выражений заданных логических функций?
9. Почему логическое произведение (функция И) реализуется последовательным соединением контактов реле, а логическое суммирование (функция ИЛИ) – параллельным соединением?
10. Каковы преимущества мостиковых релейных структур по сравнению с последовательно-параллельными соединениями?
11. Как избегать ложных цепей при составлении релейно-контактных схем управления?
12. Каковы основные разновидности и схемные обозначения логических элементов?
13. Как реализуются основные логические функции на элементах И-НЕ?
14. Как противодействовать гонке импульсов в логических схемах управления?
15. Приведите схему структуры последовательностного автомата и объясните, как формируются сигналы, отображающие внутреннее состояние автомата, и выходные сигналы.
16. Дайте пример составления схемы простейшего автомата с использованием таблицы переходов и выходов.
17. Перечислите основные виды запоминающих устройств, применяемых в логических автоматах с памятью.
18. Перечислите основные виды триггеров и приведите пример синхронного триггера.
19. Опишите логическую структуру и приведите схему универсального *JK*-триггера.
20. Как с помощью *JK*-триггера реализуются триггеры иных типов?
21. Перечислите основные разновидности регистров, приведите пример схемы регистра.

22. Как реализуются операции умножения и деления на регистрах
23. Каков принцип построения счетных регистров (счетчиков)?
24. Приведите схему реверсивного счетчика и объясните, как она производит счет импульсов.
25. Дайте пример схемы линейного дешифратора.
26. Объясните принцип построения ячейки сумматора двоичных чисел.
27. Как реализуются сумматором операции сложения и вычитания?
28. Приведите алгоритм цифрового регулятора в виде формулы для вычисления регулирующего воздействия и дайте объяснение порядка реализации заданных расчетов.
29. Приведите схему алгоритма цифрового регулятора и объясните, как она обеспечивает функционирование регулятора.
30. Приведите пример схемы общего алгоритма управления сложным ТО и объясните с его помощью порядок функционирования системы управления.

Глава 6. Особенности проектирования АСУТП

6.1. Основные задачи и принципы проектирования

АСУТП и управляемый ею технологический комплекс образуют единую производственную автоматизированную систему в составе ГПС (см. §7.1). При проектировании учитываются, прежде всего, многочисленные связи АСУТП с управляемым объектом, а также с верхним уровнем управления, представляющим собой то или иное сочетание САПР с системой планирования сменно-суточных заданий. Основные задачи, решаемые при проектировании АСУТП, – это техническое, программное и информационное обеспечение управления технологическим комплексом.

Техническое обеспечение включает в себя выбор комплекса типовых технических средств, необходимых для реализации заданных функций АСУТП, и проектирование необходимого специального оборудования. Виды технических средств, употребляемых при проектировании АСУТП, нетрудно проследить по структуре УВМ, представленной на рис.1.2. Это, прежде всего, комплектные управляющие устройства: промышленные компьютеры и др. Они обеспечивают прием управляющей и технологической информации, ее переработку в соответствии с заданной программой и формирования управляющих сигналов. Комплектование указанных управляющих устройств производится выпускаемыми серийно модулями и узлами микропроцессорной техники, как то:

- модули обработки технологической информации, обеспечивающей получение информации о технологических параметрах и состоянии технологического оборудования;
- типовые устройства связи с объектом, обеспечивающие передачу управляющей информации нижнему уровню управления, локальным УВМ, или непосредственно исполнительным устройствам;
- типовые запоминающие устройства, жесткие и гибкие магнитные диски с дисковыми, обеспечивающие нужный объем памяти для информационных систем АСУТП;
- модули цифрового управления от УВМ исполнительными устройствами, входящие в состав систем управления исполнительными устройствами;
- типовые устройства связи с обслуживающим персоналом: клавиатура, дисплеи, принтеры и пр.

Выбор технических средств состоит в сравнении различных вариантов технического обеспечения разрабатываемой АСУТП в зависимости от технических, экологических, производственных, эксплуатационных и иных требований. АСУТП являются сложными

техническими объектами, для которых непросто найти наиболее предпочтительный вариант технических средств. Поскольку общепринятые количественные критерии сравнения и общие формализованные процедуры выбора технических средств отсутствуют, то их выбор производится по преимуществу эвристическими методами и основан на опыте и инженерной интуиции разработчиков.

Процедура выбора технических средств состоит из двух этапов. Первым этапом является выбор наиболее подходящего комплекса технических средств из предлагаемых отечественными и зарубежными производителями. На втором этапе производят формирование конкретного набора технических средств для проектируемой АСУТП на базе ранее выбранного комплекта. Выбор базового комплекта технических средств определяется основными характеристиками управляемого технологического комплекса, из которых важнейшими являются:

- 1) тип технологического процесса (формообразования, изменения агрегатного состояния или химического состава продукции, транспортировки сырья и фабрикатов);
- 2) характер протекания процесса во времени (непрерывный или дискретный);
- 3) протяженность ТО (станок, технологическая линия, прокатный стан и пр.);
- 4) особенности исполнительных устройств (электропривод, нагревательные, электрохимические и пр. устройства);
- 5) степень пожаро- и взрывоопасности;
- 6) особенности окружающей среды (температура, влажность, агрессивность и пр.);
- 7) перспективы развития и модернизации техпроцесса.

Что касается выбора конкретного набора технических средств, то он более определяется специфическими требованиями к АСУТП:

- 1) уровень иерархии, зависящий от взаимодействия ЦУВМ с локальными УВМ и от развитости функций последних;
- 2) количество и характер измеряемых и регулируемых параметров, а также требования к точности измерений и качеству регулирования;

- 3) выполняемые функции (управление технологическим циклом, стабилизация параметров на заданном уровне, многосвязное автоматическое регулирование, автоматическая оптимизация и пр.);
- 4) связь с другими автоматизированными системами (ГПС, АСУП);
- 5) перспективы развития АСУТП данного типа.

Выбор технических средств не является одноразовым действием. Состав технических средств АСУТП многократно уточняется в процессе проектирования. Соответственно выясняется необходимость проектирования недостающей аппаратуры.

Программное обеспечение АСУТП подразделяется на системное, инструментальное и прикладное. *Системное* программное обеспечение составляют операционные системы и системы управления базами данных, поставляемые производителями аппаратной части УВМ. *Инструментальное* программное обеспечение является основным объектом усилий разработчиков АСУТП, так как оно служит основным инструментом создания управляющих программ (УП). Оно включает в себя ассемблеры, трансляторы, интерпретаторы, редакторы и другие программные средства, облегчающие создание УП. Эти средства в своем наиболее развитом виде объединяются в рамках систем автоматизированного программирования (САП) и входят в состав АСТПП. *Прикладное* программное обеспечение – это совокупность УП. Оно создается преимущественно на этапе эксплуатации АСУТП в порядке решения текущих производственных задач.

Информационное обеспечение – это единая система технологической, технико-экономической, справочной и оперативной информации, обеспечивающая нормальное функционирование АСУТП. Информационное обеспечение АСУТП реализуется, прежде всего, в электронном виде в памяти УВМ: базы данных, массивы УП, технические описания и инструкции и пр. Кроме того, часть информационного обеспечения подготавливается на бумажных носителях. Это технические описания и спецификации, инструкции по эксплуатации и другая эксплуатационная документация, призванная облегчить обучение ремонтно-эксплуатационного персонала, наладку, внедрение и нормальную эксплуатацию АСУТП.

При проектировании АСУТП обязательно применение комплексного системного подхода. В соответствии с ним АСУТП должна рассматриваться как сложная система, состоящая из взаимосвязанных частей, взаимодействие которых должно быть правильно и достаточно точно учтено. Общие свойства АСУТП реализуются через закономерное

взаимодействие всех ее узлов, блоков и модулей. Рассмотрим основные положения системного подхода при проектировании АСУТП.

- 1) Должна быть четко сформулирована целевая функция АСУТП, определены основное назначение и задачи, которые должна решать проектируемая АСУТП в общей структуре управления данным производством. В результате выявляется место АСУТП в иерархической структуре управления производством. Определяется совокупность управляющих воздействий, поступающих на вход АСУТП, и основные свойства управляемого технологического комплекса как объекта управления.
- 2) Прорабатывается иерархическая структура АСУТП, выделяются подсистемы, реализуемые на базе ЛУВМ, определяется их взаимодействие с ЦУВМ и с подчиненными им системами управления исполнительных устройств. Определяются взаимные связи между отдельными элементами управляемого техпроцесса и соответственно – между системами управления исполнительных устройств, управляющих ходом техпроцесса. Определяются возможности декомпозиции общей задачи реализации целевой функции АСУТП на частные задачи управления взаимосвязанными комплексами исполнительных устройств. Это приводит к иерархическому принципу построения АСУТП (см. §1.2) как неотъемлемому признаку системного подхода.
- 3) Прорабатываются возможности построения АСУТП на базе комплектного, выпускаемого серийно оборудования. Разработка оригинальных устройств, узлов и модулей специального назначения целесообразно лишь в тех случаях, когда их применение в составе АСУТП значительно повышает экономическую эффективность или другие существенные показатели эффективности эксплуатации автоматизируемого комплекса.
- 4) Проектирование АСУТП должно производиться поэтапно, пройдя ряд стадий от составления технического задания до создания рабочего проекта, со все большей детализацией.

6.2. Этапы разработки и внедрения АСУТП

Создание АСУТП – это весьма сложный процесс, который проходит через пять стадий: разработка технического задания, выполнение технического проекта, создание рабочего проекта, внедрение в производство и анализ функционирования АСУТП в производственных условиях. Проектирование АСУТП производится

коллективом специалистов: технологов, электромехаников, программистов, экономистов, в котором специалисты по электроприводу играют заметную роль.

Разработка *технического задания* связана с целым рядом предварительных исследований, связанных с обследованием автоматизируемого технологического комплекса и с проведением (при необходимости) предпроектных научно-исследовательских работ (НИР). Предметом НИР являются, прежде всего, анализ техпроцесса как объекта управления с целью определения критериев функционирования системы управления с учетом имеющихся ограничений. В рамках предпроектных НИР разрабатываются предварительные варианты математических моделей техпроцесса и целевой функции системы управления с использованием теоретических и экспериментальных методов синтеза алгоритмов функционирования и управления.

На основе результатов предварительных исследований проводится эскизная проработка проекта будущей АСУТП. При этом разрабатываются основы функционально-алгоритмической структуры системы, производится синтез основных алгоритмов управления и их экспериментальная проверка. Затем обосновывается выбор основных технических средств системы и предлагается их номенклатура, определяются задачи модернизации технологического оборудования в связи с совершенствованием техпроцесса благодаря его автоматизации, производится сравнение разрабатываемой АСУТП и ее ближайших аналогов с целью более полного обоснования принятых технических решений. В завершение предварительных исследований и разработок формируются основные положения технического задания. Они включают в себя: полный перечень функций, реализуемых проектируемой системой; технико-экономическое обоснование ее создания; виды и объемы проектных и научно-исследовательских работ по проектированию, внедрению и анализу функционирования создаваемой АСУТП; план-график работ по монтажу, наладке и пуску автоматизированного комплекса.

На стадии выполнения *технического проекта* сначала производится системотехнический синтез АСУТП: разрабатывается функциональная структура системы с определением круга решаемых при проектировании задач, формируется перечень воспринимаемых входными модулями АСУТП сигналов и определяются их метрологические характеристики, формируются технические требования к устройству контроля и регулирования технологических параметров, разрабатывается информационная и организационная структура системы. Затем производится окончательный выбор технических средств, посредством которых строится АСУТП: определяется

аппаратная часть центральной и локальных УВМ; выбираются измерительная аппаратура и приборы, предназначенные для производства измерений технологических параметров; разрабатываются устройства локальной автоматики и определяется состав исполнительных устройств технологического комплекса. В связи с этим выдаются задания на модернизацию отдельных технологических узлов автоматизируемого комплекса и на строительно-монтажные работы.

В порядке технического проектирования производится подготовка заявок на стандартное и нестандартное оборудование, составляется смета на технические средства, необходимые для комплектования проектируемой АСУТП. Производится также расчет ожидаемой технико-экономической эффективности и сравнительный анализ проектируемой АСУТП в сопоставлении с известными аналогами. Составляется ее патентный формуляр.

На заключительной стадии технического проектирования проектируется специальное программное и информационное обеспечение АСУТП. В частности, разрабатываются полные наборы алгоритмов нормального функционирования системы в режиме реального времени и производится их экспериментальная проверка. Осуществляется разработка схем сбора и преобразования технологической информации. С другой стороны, разрабатывается инструментальное программное обеспечение, автоматизирующее создание прикладных управляющих программ вплоть до уровня САП.

Рабочее проектирование производится по материалам технического проекта. Его основной частью является разработка рабочей документации технического обеспечения АСУТП. При этом, помимо рабочих чертежей составляется техническая документация: технические описания системы в целом и ее составляющих частей, спецификации, инструкции по эксплуатации и другая документация, являющаяся неотъемлемой частью информационного обеспечения АСУТП.

В состав рабочего проекта входит также рабочая документация на программное обеспечение АСУТП, включая необходимое системное программное обеспечение и образцы управляющих программ. Программное обеспечение сопровождается соответствующим информационным обеспечением. Это инструкции по программированию, описания алгоритмов специального математического обеспечения, перечни и описания специализированных программ. В рамках рабочего проекта предоставляется полный комплект нормативно-справочной документации.

Внедрение новой АСУТП в производство происходит в следующем порядке.

- 1) Аппаратная часть АСУТП устанавливается вблизи управляемого технологического комплекса и соединяется с ним линиями связи.
- 2) Проводится комплекс организационно-технических мероприятий, в рамках которых производится необходимая модернизация технологического оборудования в целях адаптации его к автоматическому управлению, а также обучение ремонтно-эксплуатационного персонала.
- 3) Производится наладка АСУТП, начиная с наладки ее технических средств. Затем отрабатываются алгоритмы управления отдельными узлами и производственными модулями автоматизируемого комплекса, после чего совершается переход к отладке решения отдельных функциональных задач на базе всего технологического комплекса.
- 4) Осуществляется опытная эксплуатация АСУТП. В этот период определяются эксплуатационные характеристики автоматизированного комплекса, производится дополнительная отладка технических средств и программного обеспечения, соответственно корректируется техническая документация.

В процессе наладки и опытной эксплуатации внедрение АСУТП в производство осуществляется поэтапно. Вначале АСУТП работает в *режиме сбора данных*, а управление технологическим процессом производится оперативным персоналом. В это время производится дополнительное изучение автоматизируемого техпроцесса и уточнение его математической модели. Затем АСУТП переводится в *режим советчика*. При рабочем режиме советчика задействованы алгоритмы управления техпроцессом, но вырабатываемые управляющие сигналы не реализуются, а выводятся в качестве рекомендаций оператору, управляющему техпроцессом. В этом режиме производится дополнительное изучение эффективности алгоритмов управления, реализуемых АСУТП, и их уточнение. При достижении достаточной надежности функционирования алгоритмов управления, АСУТП переводится в *режим непосредственного управления техпроцессом* и опытная эксплуатация реализуется в полной мере.

Опытная эксплуатация завершается *приемо-сдаточными испытаниями*, которые производятся по заранее разработанной и утвержденной в установленном порядке программе. По результатам опытной эксплуатации и приемо-сдаточных испытаний производится доработка АСУТП перед передачей ее в постоянную эксплуатацию.

Последующий анализ функционирования АСУТП в производственных условиях позволяет определить реальную технико-экономическую эффективность автоматизированного технологического комплекса и наметить пути совершенствования АСУТП и автоматизированного техпроцесса.

Контрольные вопросы к главе 6

1. Каков состав технического обеспечения АСУТП?
2. Каким образом производится выбор технических средств АСУТП?
3. Какие требования к управляемому ТО и к АСУТП учитываются при выборе технических средств?
4. В чем заключается программное и информационное обеспечение АСУТП?
5. В чем состоит системный подход к проектированию АСУТП?
6. Каковы основные этапы разработки АСУТП?
7. Как производится разработка технического задания на АСУТП?
8. В чем состоит выполнение технического проекта АСУТП?
9. Как производится рабочее проектирование АСУТП?
10. Каков порядок внедрения новой АСУТП в производство?

Глава. 7. Системы программного управления производственными установками

7.1. Назначение и общая структура

Системы программного управления (СПУ) – это АСУТП, которые обеспечивают управление технологическими объектами по

заранее подготовленной и введенной в УВМ на специальном программоносителе программе. Первичные программоносители, с которых программы вводятся в СПУ, – это электронные ОЗУ УВМ и магнитные диски (дискеты). Основу СПУ составляют УВМ различной сложности на базе микропроцессорных систем: микроконтроллеры, программируемые контроллеры и промышленные компьютеры.

Наиболее сложные СПУ в машиностроении имеют четырехступенчатую иерархическую структуру (см. рис. 7.1) и совместно с управляемым технологическим комплексом и управляющими структурами САПР составляют гибкую производственную систему (ГПС), выходящую за рамки АСУТП. Целью ГПС является ведение техпроцесса по безлюдной технологии, т.е. предполагается, что ГПС позволяет вести техпроцесс в течение рабочей смены или нескольких смен без вмешательства технологического персонала и работников ремонтно-наладочных служб. В состав ГПС обычно входят следующие подсистемы (см. рис.7.1).

- Подсистема планирования, составляющая совместно с САПР и АСТПП верхний уровень управления. Ее назначением является планирование сменно-суточных заданий, подготовка проектной документации и материально-технического снабжения, включая поставку сырья, материалов, инструмента и пр.

- Подсистема диспетчеризации, которая обеспечивает оперативное управление технологическим комплексом, порядок выполнения производственной программы (верхний уровень АСУТП).

- Транспортно-накопительная подсистема, включающая в себя автоматизированный склад и устройства для укладки, временного накопления, разгрузки и доставки обрабатываемых изделий, инструмента и технологической оснастки.

- Гибкие производственные модули (ГПМ), оснащенные устройствами ЧПУ (УЧПУ). ГПМ – это единица технологического оборудования с программным управлением, предназначенная для производства некоторого класса изделий как в составе ГПС, так и в автономном режиме (например, станки с ЧПУ).

- Подсистема автоматизированного контроля, важнейшими функциями которой являются диагностика неисправностей и контроль качества продукции.

Структура оборудования, входящего в состав ГПС, показана на рис. 7.1. Высшим уровнем управления в ГПС является ЦУВМ (два промышленных компьютера, см. рис. 2.6), которая получает от АСУ предприятия технологические программы, а также месячный план работы ГПС и его корректировку. Первый компьютер реализует

функции АСТПП и планирования, а второй – функции оперативного управления техпроцессом. ЦУВМ производит сменно-суточное планирование на базе полученного месячного плана с учетом фактического состояния производства (обеспеченность заготовками, инструментом, технологической информацией, выполнение предыдущего сменно-суточного задания с учетом брака и незавершенного производства). Сменно-суточное задание поступает в центральную диспетчерскую систему, которая осуществляет оперативное управление технологическим комплексом. Управление отдельными видами обработки изделий производится в рамках ГПМ (на рис.7.1 представлено N ГПМ). В состав ГПМ могут входить помимо станков с ЧПУ обрабатывающие центры, промышленные роботы, оснащенные средствами активного контроля и адаптивного управления и пр. Материальные потоки ГПС обслуживаются автоматизированным складом, связанным с ГПМ транспортными устройствами. Системы управления ГПМ, складом и транспортом являются ЛУВМ (см. рис.1.1) в составе ГПС. Обмен информацией ЦУВМ с локальными СПУ ГПМ реализуется информационной магистралью типа Ethernet или Profibus (см. §2.6).

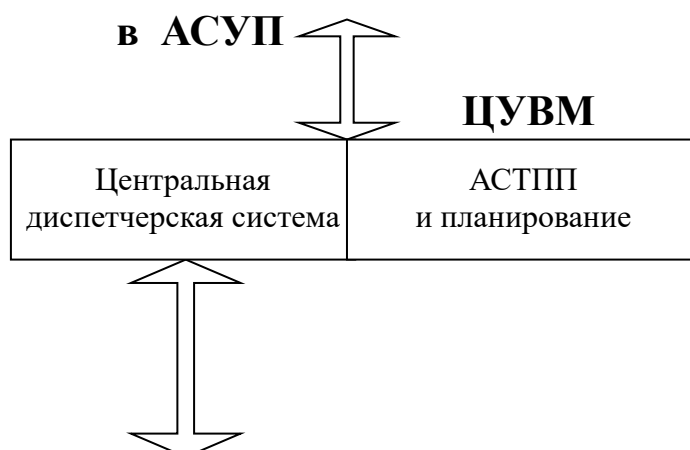


Рис. 7.1. Структура оборудования ГПС

7.2. Локальные системы программного управления

Локальные СПУ – это системы, служащие для управления отдельными ГПМ, транспортными устройствами, роботами. Основные функции локальных СПУ:

- *Централизованный контроль*: сбор данных о технологическом объекте и о работе АСУТП с помощью разного рода датчиков и преобразователей; оценка отклонений данных от заданных значений и расчет основных параметров техпроцесса; анализ срабатывания блокировок и защит, диагностика состояния оборудования.

- *Регулирование техпроцесса*: стабилизация параметров на заданном уровне, программное управление изменением параметров по заданному закону, автоматическая оптимизация параметров техпроцесса.

- *Программно-логическое управление*: выполнение различных переключений по заданной программе и в зависимости от значений параметров техпроцесса, т.е. управление технологическим циклом работы оборудования.

- *Ввод и отображение информации*: ввод данных и команд вручную с помощью клавиатуры и разного рода переключателей; оперативное отображение вводимых данных и технологической информации на дисплеях и с помощью других средств индикации.

- *Обмен информацией*: подготовка данных и УП для передачи управляющим устройствам высшего и низшего уровней управления, выполнения протоколов обмена с рассредоточенными абонентами через последовательный интерфейс.

Локальные СПУ строятся на базе специализированных микропроцессорных устройств, выпускаемых серийно. В зависимости от характера и сложности выполняемых функций базовые управляющие устройства локальных УВМ выпускаются в двух видах:

- программируемые контроллеры (ПЛК), или программируемые логические контроллеры, основным назначением которых является программно-логическое управление технологическим циклом (цикловое управление);

- УЧПУ (устройства числового программного управления), или *промышленные контроллеры*, которые помимо программно-логического управления осуществляют сложные операции регулирования параметров, например – перемещение рабочих органов станка по заданной траектории.

При программно-логическом управлении с помощью ПЛК возможно и управление перемещением рабочих органов, но только по номерам рабочих точек в пространстве. Положения, соответствующие заданным рабочим точкам, должны быть отработаны заранее в процессе наладки. Современные ПЛК, в дополнение к основной функции управления технологическим циклом изготовления продукции, могут также осуществлять и несложное регулирование параметров техпроцесса, т.е. несложное числовое управление, но в большинстве случаев числового управления применяются УЧПУ. В состав современных УЧПУ и систем с ПЛК входит значительное количество различных микропроцессорных устройств и линий связи. Они в своей совокупности образуют информационную сеть типа Fieldbus («полевая» сеть, см. §2.6).

7.3. Программируемые контроллеры

7.3.1. Структура ПЛК

Программируемые контроллеры (зарубежное наименование – PLC, т.е. Programmable Logical Controller) – это выпускаемые серийно управляющие устройства, предназначенные преимущественно для управления технологическим циклом работы производственного оборудования. Типичная структура ПЛК показана на рисунке 7.2. Конструктивно ПЛК обычно оформлен в виде двух или трех блоков (см. рис.7.2). Обязательными являются блок управляющих модулей ПЛК (блок ПЛК) и блок программатора. Программатор – это программирующее устройство, с помощью которого управляющая программа (УП) вводится в блок ПЛК. Для разработки и ввода УП

может быть использован компьютер, оснащенный специализированным программным обеспечением. Одновременно предусматривается применение переносного клавишного программатора. Последний используется преимущественно для наладки системы управления, построенной на базе ПЛК. После ввода и отладки УП контроллер реализует заданные управляющие функции, работая в автономном автоматическом режиме.

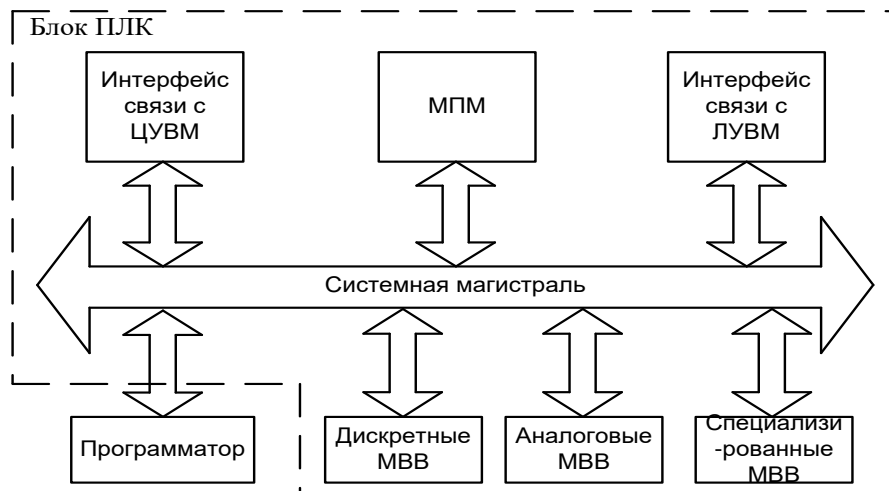


Рис. 7.2.

Структура программируемого логического контроллера

Процессорный блок ПЛК имеет модульную структуру, что позволяет формировать “корзину” ПЛК с большим или меньшим количеством модулей различного назначения в зависимости от сложности и характера решаемых задач управления. Обязательным модулем ПЛК, наряду с модулем питания, является модуль процессора МПМ (микропрограммный модуль, см. рис.7.2). В этом модуле находится микропрограммное устройство, осуществляющее переработку поступающей информации в соответствии с заданной УП и формирование управляющих сигналов. В нем же находится устройство памяти ПЛК, включая ПЗУ и ОЗУ. Для ввода информации, поступающей от управляемого объекта, и вывода сигналов управления служат модули ввода-вывода МВВ. Обычно в составе ПЛК имеется несколько модулей ввода дискретных сигналов и несколько модулей вывода дискретных сигналов. Габарит ПЛК определяется общим количеством входных и выходных сигналов, которые он может принимать и, соответственно, выдавать. Например, ПЛК на 256 входов-выходов – это ПЛК, в котором может быть установлено такое количество дискретных модулей ввода или вывода сигналов, что общее количество сигналов, обрабатываемых ПЛК, может достигать 256 при условии, что аналоговые и

специализированные модули отсутствуют. Каждый дискретный модуль ввода или вывода обычно рассчитан на 8, 12, 16 и до 32 дискретных сигналов.

Помимо модулей, обеспечивающих обработку дискретных сигналов, в блоке ПЛК могут быть установлены модули для ввода или вывода аналоговых сигналов. Модули ввода аналоговых сигналов – это аналого-цифровые преобразователи (АЦП), преобразующие непрерывные аналоговые сигналы в цифровые (числовые). Напряжению или току на входе АЦП соответствует число на выходе АЦП, пропорциональное данному напряжению или току. Модули вывода аналоговых сигналов – это цифро-аналоговые преобразователи (ЦАП), преобразующие заданное процессором число, поданное на данный выход, в пропорциональное ему напряжение или ток, формируемые на данном выходе. При необходимости в блоке ПЛК устанавливаются различные специализированные МВВ. Среди них отметим модули для приема импульсных сигналов датчиков перемещений, а также модули, предназначенные для управления следящими электроприводами. Особо следует упомянуть модуль для обмена информацией с центральной УВМ, который выполняется на базе последовательного интерфейса и устанавливается, если рассматриваемый ПЛК играет роль ЛУВМ. Последовательный интерфейс может применяться и для управления ведомыми устройствами по схеме ведущий-ведомый, в том числе (через УСО) – с интеллектуальными датчиками. Если же ПЛК играет роль ЦУВМ, то он оснащается модулем последовательного интерфейса для связи с локальными УВМ.

7.3.2. Языки программирования ПЛК

Большинство задач управления в рамках АСУТП требует не столько вычислительных арифметических операций (хотя бывают, необходимы и они), сколько логических операций, без которых не обходится ни один процесс управления. Логические операции совершаются над сигналами, которые либо приходят по каналам обратной связи от датчиков технологических параметров, либо вырабатываются различными управляющими устройствами в составе АСУТП. Сигналы логического типа (дискретные сигналы) принимают значения либо 0; либо 1. Они либо поступают в ПЛК через дискретные модули входов (сигналы обратной связи), либо выдаются из ПЛК через дискретные модули выходов на исполнительные устройства (управляющие сигналы). Получение информации о сигналах обратной связи называется *опросом входов*.

Поскольку информация о дискретном сигнале принимает значение либо 0, либо 1, то для ее хранения применяют ячейку памяти триггерного типа. При программировании каждую ячейку памяти

принято именовать *битом*. В системе логического управления каждый дискретный сигнал функционирует в виде логической переменной. Каждый бит в системе программирования ПЛК представляет такую логическую переменную или константу и может соответственно принимать значение либо 0, либо 1. При программировании имеют дело не с самими сигналами, а с битами, с помощью которых исходные сигналы представлены в памяти ПЛК. Бит – это элементарный объем адресации в стандартных системах программирования ПЛК. Возможны и другие объекты адресации, включая численные величины.

Порядок программирования ПЛК определяется международным стандартом IEC 61131-3. Помимо объектов – битов в нем предусмотрена возможность объединения битов в *слова*. Предусмотрены следующие форматы слов:

B – *байт*, состоящий из 8 бит и предназначенный для обозначения символов, применяющихся при программировании ПЛК;

W (Word) – *слово одинарной длины*, составляет 16 бит; такие слова применяются для записи совокупностей битов, а также чисел от -32768 до 32767;

D (Double) – *слово двойной длины*, т.е. длиной 32 бит; оно может быть использовано для записи алгебраических величин, как и слово одинарной длины;

F (Floating) – *слово с плавающей точкой* (оно же – с плавающей запятой) применяется в алгебраических и других математических операциях.

Для обозначения объектов адресации ПЛК приняты следующие символы:

I (Input) – бит входа (отображение дискретного входного сигнала);

Q (Output) – бит выхода (отображение дискретного выходного сигнала);

X – обозначение битового формата при адресации;

M (Memory) – бит памяти (промежуточная переменная) или начало обозначения величины, хранящихся в памяти ПЛК;

K – константа;

S – системный бит, контролирующий корректность функционирования ПЛК в процессе выполнения прикладной программы;

L (Label) – метка.

Прежде чем приступить к написанию текста прикладной программы, необходимо определить адреса ячеек памяти, в которых будут храниться значения переменных и постоянных величин, фигурирующих в составляемой программе. Адресация начинается знаком %, являющимся символом стандарта IEC 61131-3. Затем

записывается тип адресуемого объекта: *I* – вход, *Q* – выход, *M* – внутренняя (промежуточная) переменная и пр. После этого указывается формат объекта (бит или слово того или иного типа) и номер бита или регистра памяти, который этим объектом занят. Номера битов, предназначенных для хранения информации о состоянии дискретных входов и выходов, задаются в строго определенном порядке. При этом старшая цифра номера должна соответствовать номеру корзины (шасси, крейта), в которой установлен модуль, содержащий адресуемый вход или выход, следующая группа цифр должна соответствовать номеру модуля, последняя группа цифр – это номер входа (или выхода), предназначенного в данном модуле для подачи на него адресуемого сигнала. В памяти ПЛК последняя группа цифр, о которой упоминалось выше, является номером бита, в котором запоминается состояние адресуемого входа или выхода. Номер бита определяется от номера регистра точкой. Например, % I12.5 означает, что в битовую ячейку 5 регистра 12, расположенного в нулевой корзине (нули перед первой значащей цифрой опускаются), поступает сигнал о состоянии входа 5 модуля входов №12. Номера ячеек памяти, предназначенных для хранения внутренних переменных, распределяются по усмотрению программиста.

Стандартом IEC 61131-3 определены пять языков программирования ПЛК, опирающихся на общую систему адресации, на стандартизированные общие элементы: символы, форматы данных и переменных. Это позволяет программировать ПЛК на любом стандартном языке программирования в зависимости от характера решаемой задачи и вкусов проектировщика системы автоматизации. Расположив рассматриваемые языки программирования ПЛК в порядке сложности реализуемых ими функций, получим следующий ряд.

LD (Ladder Diagrams), – язык «лестничных диаграмм», или язык РКС. Он представляет собой графическую интерпретацию процесса разработки релейно-контактных схем (РКС) управления (см. §5.3). Управляющая программа на языке LD состоит из релейных управляющих цепочек, ступеней, отображаемых на дисплее компьютера, на котором производится программирование, вместе с ограничивающими их условными линиями электропитания. Получается изображение, несколько напоминающее лестницу. Это «лестничная» конструкция автоматически переводится на язык машинных кодов и транслируется в ПЛК. Язык LD удобен для программирования решения относительно несложных задач управления и является основным для многих ПЛК младших классов.

IL (Instruction List) – язык набора инструкций. В его основе лежит своеобразный метод отображение логических формул булевой

алгебры (см. §5.2), когда они выписываются в виде колонок инструкций (команд), разделенных на две части: слева записывается код операции в удобном для запоминания (мнемоническом) виде, а справа – адрес логической переменной, над которой должна быть совершена указанная операция. Язык IL – наиболее удобный и эффективный язык программирования низкого уровня. В основе его лежит язык STEP5 фирмы SIEMENS. Ниже язык IL будет рассмотрен более подробно.

FBD (Functional Block Diagrams) – язык функциональных блоков. Этот язык ведет свое происхождение от структурных и логических схем автоматического управления, знакомых каждому специалисту в области автоматизации. Техника проектирования на языке FBD сводится к отображению на дисплее компьютера системы управления в виде набора стандартных функциональных блоков (обычно в виде прямоугольников), соединенных между собой линиями, отображающими логические или иные связи между ними. Каждый блок обеспечивает реализацию той или иной логической или арифметической функции, функции сравнения, специальной функции управления (например, ПИД-регулятора, таймера) и пр. Внутри каждого блока располагается обозначение выполняемой им функции. Входные сигналы подводятся с левой стороны блоков, а выходные выводятся с правой стороны. Обязательно должны быть указаны источники сигналов для всех входов блока, а также приемник для выходного сигнала (не должно быть не присоединенных входов и выходов). Если выходной сигнал блока – промежуточная переменная, то выход такого блока соединяется со входом следующего блока в соответствии с логикой управления. Язык FBD удобен тем, что позволяет наглядными графическими средствами создавать управляющие программы практической любой сложности.

ST (Structured Text) – язык «структурированного текста». Язык ST относится к языкам высокого уровня вроде Basic и Pascal, но приспособлен для решения задач программного управления на базе ПЛК. Конструкции языка ST могут быть употреблены в составе языков LD и IL при решении сложных задач управления.

SFC (Sequential Function Chart) – язык последовательностных функциональных схем, или Графсет. Он предназначен для представления в понятном и заказчику, и исполнителю виде логики функционирования сложных систем автоматизации на уровне ЦУВМ. Язык SFC позволяет объединить в одном комплексе локальные управляющие программы и блоки. С его помощью достигается согласованность выполнения локальных программ, контроль состояния управляемого техпроцесса и обеспечение синхронизации по приему и обработке данных о ходе процесса. Данный язык позволяет графически отобразить управление техпроцессом в виде

укрупненных управляющих блоков, порождаемых ими действий и переходов. Переход от функциональных схем SFC к конкретным управляющим программам производится средствами описанных выше языков программирования ПЛК и, прежде всего с помощью языка ST.

7.3.3. Язык программирования IL

Язык IL предназначен для программирования ПЛК, опираясь непосредственно на логические (булевы) формулы, описывающие технологический цикл, реализуемый под управлением ПЛК (см. гл. 5). Программа, составленная на языке IL, состоит из последовательности инструкций. *Инструкцией* в данном случае называют команду, состоящую из двух частей: кода инструкции и адресной части (операнда). Код инструкции указывает, какую именно операцию следует совершить над содержимым ячейки памяти, адрес которой указан в адресной части инструкции. Система адресации языка IL, общая для всех языков программирования ПЛК, предусмотренных стандартом IEC 61131-3, описана выше. Инструкции языка IL делятся на инструкции проверки выполнения логических условий и на инструкции выполнения заданных действий.

Начнем рассмотрение проверочных инструкций с группы LD (LoaD). Они имеют следующие коды:

LD- загрузить бит, заданный в адресной части данной инструкции, в процессор;

LDN (LoaD Not)- загрузить инверсию бита, заданного в адресной части;

LDR- (LoaD Rise)- загрузить в процессор единицу, если в текущем цикле опроса значение бита, указанного в адресной части, изменилось от нуля к единице;

LDF- (LoaD Fall)- загрузить единицу, если в текущем цикле опроса значение бита, указанного в адресной части, изменилось от единицы к нулю.

Инструкция типа LD употребляется в начале последовательности инструкций, составляющей реализацию отдельной логической формулы.

Аналогично кодам инструкций группы LD формируются коды инструкций группы AND (логическое И) и группы OR (логическое ИЛИ):

AND- произвести логическое умножение результата предыдущих логических операций, хранящегося в процессоре, на бит, заданный в адресной части данной инструкции;

OR- произвести логическое сложение вышеуказанных величин;

ANDN (AND Not)- произвести логическое умножение результата в процессоре на инверсию бита, заданного в адресной части данной инструкции;

ORN- произвести логическое сложение вышеуказанных величин: бита в процессоре с инверсией бита, заданного в адресной части;

ANDR (AND Rise)- произвести логическое умножение результата в процессоре на единицу, если в текущем цикле опроса значение бита, указанного в адресной части, изменилось от нуля к единице.

Остальные коды: ORR, ORF и ANDF расшифровываются аналогично. Предусмотрены также аналогичные коды XOR, XORN, XORR и XORF для инструкций, реализующих исключающее ИЛИ.

Помимо инструкций, реализующих отдельные логические операции, язык IL предусматривает *проверочные блоки*. Каждый из проверочных блоков реализует целую группу логических операций. Типичными проверочными блоками являются *таймер* (timer), служащий для организации выдержек времени, и *счетчик* (counter), позволяющий организовать счет импульсов и технологических циклов.

Инструкции выполнения заданных действий (инструкции действия) инициируют подачу процессором управляющих сигналов, соответствующих сложившимся логическим условиям. Рассмотрим группу инструкций действия ST (STorage). Они имеют следующие коды:

ST- переслать по заданному в адресной части данной инструкции адресу значение бита (0 или 1), находящегося в процессоре (бит был сформирован в результате предыдущих проверочных операций);

STN (STorage Not)- совершить такую же операцию, что и по инструкции ST, но с предварительной инверсией значения бита, находящегося в процессоре;

S (Set)- установить бит, адрес которого указан в адресной части данной инструкции, в состояние 1, если в текущем цикле работы процессор находится в состоянии 1; состояние 1 адресуемого бита сохраняется и в дальнейшем, даже если процессор перейдет в состояние 0;

R (Reset)- сбросить бит, адрес которого указан в адресной части, в состояние 0, если в текущем цикле работы процессор находится в состоянии 1; состояние 0 адресуемого бита сохраняется, даже если процессор перейдет в состояние 0.

Инструкциями ST, STN, S и R задаются значения выходных и промежуточных дискретных управляющих сигналов ПЛК.

К инструкциям действия относятся также инструкции передачи управления JMP (JuMP) и END. Их коды:

JMP – безусловный переход на метку, номер которой указан в адресной части данной инструкции;

JMPC (JuMP on Condition) – условный переход на метку; переход реализуется, если в результате выполнения предыдущей логической операции процессор оказался в состоянии 1, а иначе выполнение программы продолжается обычным порядком;

JMPCN (JuMP on Condition Not) – условный переход на метку в случае, когда в результате выполнения предыдущей логической операции процессор оказывается в состоянии 0, а иначе выполнение программы продолжается обычным порядком.

Код END означает, что выполнение цикла управления завершено. После этого ПЛК переходит к началу нового цикла управления (к началу управляющей программы). Если же нужно остановить выполнение циклической программы, то используется инструкция HALT. У инструкций END и HALT нет адресной части (безадресные инструкции).

Если в логической формуле имеются скобки, то их следует учесть и при отображении данной логической формулы на языке IL. В качестве примера рассмотрим две логические формулы:

$$Q2.1 = I1.1 \cdot I1.2 + I1.3 \text{ и}$$

$$Q2.2 = I1.1 (I1.2 + I1.3).$$

Здесь мы полагаем, что информация о входных сигналах X_{11} , X_{12} и X_{13} попадает в ячейки памяти I1.1, I1.2 и I1.3, а формируемые в ПЛК выходные сигналы Y_{21} и Y_{22} направляются на выходы Q2.1 и Q2.2. Программный фрагмент, отображающий формулу для Q2.1, будет форматироваться без скобок:

```
LD  %I1.1
AND %I1.2
OR  %I1.3
ST  %Q2.1
```

В данном фрагменте в процессор загружается (LD) значение бита I1.1, которым и определяется после этого текущее состояние процессора (0 или 1). Затем производится логическое умножение (AND) данного состояния на значение бита I1.2. После этого производится логическое сложение (OR) нового состояния процессора с битом I1.3. Копия результата сложения (0 или 1), полученного в процессоре, пересылается (ST), в полном соответствии с формулой для Q2.1, на выход Q2.1.

Если необходимо реализовать функцию Q2.2, то нужно применить скобки:

```
LD  %I1.1
AND(%I1.2
OR  %I1.3
)
ST  %Q2.2
```

Открывающая скобка ставится непосредственно после кода операции, которую нужно реализовать над результатом действий в скобках, а закрывающая скобка ставится в виде отдельного оператора после завершения программирования всех действий в скобках.

Если необходимо произвести сложные логические, арифметические и др. математические операции, то можно воспользоваться конструкциями языка ST. Действия, программируемые на языке ST, следует заключать в квадратные скобки. Например:

[%MW4: = %MW1 + 10].

Здесь к содержимому регистра памяти MW1 прибавляется 10, а полученный результат запоминается в регистре MW4.

Корректно оформленный фрагмент программы представляет собой последовательность инструкций, расположенных в столбик, по одной инструкции в строке. Каждый автономный фрагмент начинается с восклицательного знака, вслед за которым может располагаться комментарий, ограниченный звездочками. При необходимости ниже комментария может располагаться метка (Label), которая имеет вид %Li, (где i может принимать значения от 0 до 999). Метки используются для организации переходов внутри программы. Первой инструкцией (вслед за меткой, если она имеется) должна быть инструкция типа LD или JMP. Каждая последовательность должна содержать одну или несколько проверочных инструкций и завершаться инструкцией действия.

При выполнении программы инструкции выполняются в том порядке, в котором они введены в ПЛК, но с учетом заданных внутри программы переходов. Нормальный режим работы ПЛК заключается в циклическом повторении управляющей программы. Поэтому после отработки инструкции END автоматически начинается выполнение управляющей программы с самого начала.

В качестве примера составим программу регулирования уровня жидкости путем автоматического включения и выключения электропривода насоса, подающего жидкость, в соответствии с формулой (5.10). Запишем ее в скобочной форме:

$$H^t = d_s (d_n + H^{t-1}).$$

Примем также, что сигналы датчиков d_s и d_n подаются на входы I1.2 и I1.1 соответственно, а выходной сигнал Н на включение насоса формируется на выходе Q1.1. Чтобы избежать скобок в программе, начнем с программирования выражения в скобках. Получим:

!

```
%L1
LD   %I1.1
OR   %Q1.1
```

AND %I1.2

ST %Q1.1

Метка L1 необходима, чтобы можно было организовать включение насоса из разных сегментов программы.

7.4. Устройства числового программного управления

УЧПУ – это комплектные системы управления с ЧПУ (зарубежное наименование ЧПУ – CNC, т. е. Computer Numerical Control) в машиностроении.

Числовым программным управлением (ЧПУ) называют управление техпроцессом, в котором величины перемещений рабочих органов и другие параметры техпроцесса задаются в числовой форме, в виде чисел. Выше мы выяснили, что при цикловом программном управлении (например с помощью ПЛК) траектория перемещения рабочих органов задается номерами точек в пространстве, а движение между заданными точками жестко определяется конструкцией оборудования. При числовом же программном управлении траектория движения задается численно либо координатами опорных точек (позиционное управление), либо в виде непрерывной кривой (контурное управление).

Основные задачи решаемые УЧПУ:

- программно-логическое управление технологическим циклом (аналогично ПЛК);
- следящее управление перемещением рабочих органов по заданной траектории;
- автоматическая оптимизация техпроцесса;
- обеспечение связи с оператором и ЦУВМ;
- диагностика УЧПУ и основного оборудования;
- подготовка УП с помощью инструментального программного обеспечения УЧПУ.

Аппаратной базой УЧПУ являются микропроцессорные комплекты и большие интегральные схемы (БИС). Центральным узлом УЧПУ является компьютер, который решает основные задачи управления, программирования, технологических расчетов, диагностики, связи с оператором и с ЦУВМ. Отсюда название CNC.

Структура аппаратной части УЧПУ может быть отображена блок-схемой, приведенной на рис. 6.2, при том отличии, что программатор УЧПУ может быть тем же компьютером, который является основой управляющей части УЧПУ. Программирование может также производиться ЦУВМ, снабжающей локальные УЧПУ управляющими программами через последовательный интерфейс. Специализированные модули УЧПУ зачастую строятся на базе специализированных микро-ЭВМ, например – для обеспечения цифрового управления следящими электроприводами в соответствии

с цифровыми уставками, формируемыми процессором УЧПУ согласно обрабатываемой УП. Алгоритмы, по которым формируются команды на перемещение рабочих органов, зависят от типа *интерполяции*, заданной в УП, и от способа учета геометрии инструмента при программировании его траектории.

Геометрическая информация включает в себя вид траектории (линейная, круговая и пр.), скорости и ускорения при обработке элементов траектории. Траектория перемещения рабочего органа формируется в виде совокупности опорных точек, перемещение между которыми производится по определенному закону. При этом учитывается, что около 90% траекторий, встречающихся при обработке резанием, являются либо прямолинейными, либо круговыми. Поэтому стандартными законами перемещения рабочих органов между опорными точками, заданными в УП, приняты перемещения по прямой линии либо по окружности. Если перемещение между опорными точками производится по прямой, то такой способ обработки координат называется *линейной интерполяцией*, а если по окружности – то *круговой интерполяцией*. Другие способы интерполяции употребляются редко, а лекальные кривые аппроксимируются отрезками прямых и дугами окружности. Чтобы обеспечить обработку отрезка прямой, задаются координаты его начала и конца, между которыми производится движение с помощью линейной интерполяции. Чтобы задать движение по окружности путем круговой интерполяции, необходимо помимо координат начала и конца программируемой дуги задать координаты центра окружности, часть которой составляет данная дуга.

Если данную траекторию совместить с перемещением центра инструмента, то при установке нового инструмента, имеющего другие габариты, обрабатываемая траектория будет искажена. Поэтому необходимо обеспечить автоматический расчет траектории движения инструмента с учетом его габаритов. Необходимо, чтобы обработка производилась путем перемещения инструмента по *эквидистантному* контуру, т.е. такому, при котором инструмент идет по заданному контуру на расстоянии, равным расстоянию от центра инструмента до его режущей кромки. Задача определения траектории центра инструмента с учетом габаритов последнего называется расчетом эквидистанты. В случае линейной или круговой интерполяции расчет эквидистанты наиболее прост, так как эквидистантными кривыми к прямой и дуге являются также прямая и дуга.

Особой задачей, решаемой в рамках УП, является обеспечение автоматической оптимизации, примеры которой даны в главе 4.

7.5. Программирование УЧПУ

Аппаратная часть УЧПУ изготавливается на специализированных предприятиях электронной промышленности. Проектирование УЧПУ сводится к выбору необходимого набора аппаратуры и разработки соответствующего программного обеспечения в объеме, достаточном для решения задач управления технологическим оборудованием. В заключение производится конструктивная разработка шкафа, в котором размещаются модули и узлы УЧПУ.

Программное обеспечение УЧПУ делится на *системное, инструментальное и прикладное* (см. §6.1). Управляющие программы (УП) создаются в процессе эксплуатации УЧПУ с помощью инструментального программного обеспечения, предусмотренного для него проектировщиками. В наиболее сложных случаях УП разрабатываются с помощью АСТПП. При этом используются системы автоматизированного программирования (САП), которые представляют собой проблемно-ориентированные языки, максимально приближенные к терминологии, используемой технологами, например – T-Flex-CAD, включая T-Flex-ЧПУ отечественной фирмы «Top Systems». САП входят в состав программного обеспечения центральных ЭВМ и сложных УЧПУ типа обрабатывающих центров. Для создания УП с помощью САП используются автоматизированные рабочие места (АРМ) технолога. При использовании АРМ программирование ведется в диалоговом режиме, технолог использует меню альтернативных решений и графические возможности САП для оформления документации техпроцесса. Разработанные таким образом УП составляют прикладное программное обеспечение УЧПУ. Оно хранится в памяти ЦУВМ и передается по мере необходимости в УЧПУ технологического оборудования. УП могут храниться также в устройствах внешней памяти, таких как магнитные диски и дискеты. Программа, записанная на дискете, вводится в УЧПУ с помощью дисковода. Разработка УП может производиться также оператором с пульта УЧПУ. Перед началом работы УЧПУ по УП, хранящейся в его памяти, производится наладка: вывод рабочих органов в исходное положение, ввод в память УЧПУ информации, связанной с ограничениями техпроцесса и т.п.

Наиболее распространенным языком программирования УЧПУ, лежащим в основе большинства САП, является код ISO-7, регламентированный международным стандартом ISO 6983, а также ГОСТ 20999-83, см. приложение 4. Этот код использует символы кода КОИ-8 по ГОСТ Р34.303-92 в соответствии с международным стандартом ISO4873-86 (см. приложение 1). Он содержит заглавные буквы английского алфавита, арабские цифры, знаки математических операций и ряд других символов. В коде ISO-7 символам кода КОИ-8 приданы значения, описывающие функционирование СПУ.

В коде ISO-7 УП записываются на программноноситель в виде последовательности кадров (рис.7.3).

%0	.	.	.	.	.	.	.	.	LF
(комментарий)									
:	.	.	.	.	.	.	.	.	LF
N1	.	.	.	.	.	.	.	.	LF
	.	.	.	.	.	.	.	.	LF
N	.	.	.	.	.	.	M02	.	LF
	.	.	.	.	.	.	.	.	LF
%1	.	.	.	.	.	.	.	.	LF
	.	.	.	.	.	.	.	.	LF
N	.	.	.	.	.	.	M02	M30	LF

Рис.7.3. Структура программы, записанной в коде ISO-7

Кадры состоят из информационных слов, расположенных в определенном порядке, а слова из символов. Первый символ слова является буквой латинского алфавита, а остальные символы являются арабскими цифрами и образуют число со знаком или целочисленный код. Содержанием кадра является последовательность слов, задающих одну рабочую операцию. Каждый кадр программы должен начинаться с номера кадра, который состоит из символа N и целочисленного кода. Кадры должны располагаться в порядке возрастания номеров, номера не должны повторяться. После номера кадра информационные слова должны записываться в следующей последовательности:

- N – число повторений участка программы (Heading);
- G – подготовительная функция (Geometry);
- D – величина коррекции (Displacement);
- X – перемещение (положение), заданное по оси X;
- Y – перемещение (положение), заданное по оси Y;
- Z – перемещение (положение), заданное по оси Z;
- U, V, W – перемещения (повторные) по осям X, Y, Z соответственно;
- P, Q – третичные перемещения по осям X, Y соответственно;
- I, J, K – параметры интерполяции или шаг резьбы параллельно осям X, Y, Z соответственно;
- A – угол поворота вокруг оси X ($A \rightarrow \alpha$);
- B – угол поворота вокруг оси Y ($B \rightarrow \beta$);
- C – угол поворота вокруг оси Z ($C \rightarrow \Gamma \rightarrow \gamma$);
- R – формальный параметр, служащий для хранения целых чисел со знаком (Register);
- F – величина подачи в мм/мин или в мм/об (Feed);

S – скорость главного движения в м/мин или в об/мин (Spinedle);
T – номер инструмента с указанием номера корректора или без него (Tool);
M – вспомогательная функция, задающая режим работы (Mode);
L – обращение к подпрограмме (Linkage).

Кадр завершается символом LF или ПС (перевод строки). Номер кадра (и символ ПС при вводе с перфоленты) являются обязательными в каждом кадре. Остальные символы используются по мере необходимости задания тех или иных элементов технологической операции.

Последовательности кадров объединяются в разделы УП, заголовки которых начинаются символами от % 0 до % 9. Каждый заголовок главы завершается символом ПС. Затем идут комментарии, заключенные в круглые скобки. Информация в скобках не используется для управления в УЧПУ. После комментариев, если они нужны, располагается главный кадр главы УП, который начинается с символа”:” (двоеточие). Главный кадр содержит информацию об общих условиях обработки. Он не имеет номера, а завершается, как и другие кадры, символом ПС.

Рассмотрим основные функции, реализуемые словами в составе кадра УП.

Подготовительные функции задаются символом G (Geometry) и двух или трехзначным десятичным числом. Они определяют геометрические параметры станка, например характер интерполяции (линейная или круговая), характер позиционирования, вид коррекции инструмента, вид системы координат и т.п.

Размерные перемещения X, Y, Z, U, V, W, R, Q, I, J, K, A, B, C, R – записываются целыми числами со знаком в виде абсолютных величин (положений) или в виде приращений координат в данном кадре.

Вспомогательные функции M (Mode) определяют команды для электроавтоматики станка и тем самым задают режим работы. Может быть задано до ста таких функций в пределах M00 – M99. Примерами вспомогательных функций являются M02 – команда выключения оборудования в конце раздела УП, M03 – включение шпинделя на вращение по часовой стрелке, M06 – смена инструмента и т.д.

В начале главы УП, после символа % и соответствующего номера, размещается формат слов кадра УП. В формате перечисляются все символы, используемые в данной УП, за каждым из которых идет набор цифр. Если в слове рассматриваемого типа можно опустить нули перед первой значащей цифрой, то первой цифрой формата является 0. Если же можно опустить нули после последней значащей цифры, то последней цифрой формата является 0. Если в данном слове фигурирует целое число, то в формате

записывается одна значащая цифра, определяющая максимально возможное количество разрядов записанного в слове числа. Если в слове может быть записано дробное число, то помимо начального или конечного нуля записываются 2 цифры, первая из которых обозначает максимальное количество разрядов целой части, а вторая – максимальное количество разрядов дробной части записываемого числа.

Пример формата: %2 N04 G02 D02 X043 Y043 и т.д. В указанном формате второго раздела (%2) УП N04 означает, что номер кадра может иметь до 4-х цифр (например, N8301) и нули перед первой значащей цифрой можно опустить, т.е. необязательно писать N0042, так как достаточно написать N42. В том же формате слово D02 означает, что может быть задано 99 вариантов коррекции траектории инструмента, от D01 до D99, причем вместо D01 можно писать D1. Далее в вышеприведенном формате фигурирует слово X043. Оно означает, что целая часть числа, задающего координату X, может содержать до 4 цифр, а дробная часть, поскольку нули опускаются только впереди, всегда содержит 3 цифры. Например, если в каком либо рабочем кадре имеется слово X23500, то оно означает задание координаты $X=23,5$ мм. А если задано X-235, это означает $X=-0,235$ мм.

В качестве примера рассмотрим формирование траектории перемещения рабочей точки механизма в соответствии с графиком, приведенным на рис. 7.4. Программу записываем в *абсолютных размерах*, т.е. отсчет перемещений во всех кадрах программы производится в одной и той же системе координат, показанной на рис. 7.4. При написании программы используем разработанный выше формат слов кадра. Программа отработки перемещений согласно графика, приведенного на рис. 7.4, получается такой (см. табл. П. 4.1):

%1						LF
N1	G90	G01	X21000	Y33000	F100	LF
N2		G01	X63500	Y60000		LF
N3		G01	X52500	Y14000		LF
N4		G01	X21000	Y33000		LF
N5		G01	X0	Y0	M02	LF

Здесь слово G90 в кадре N1 задает отсчет перемещений в абсолютных размерах, и это задание действует и в последующих кадрах, так же как и задание скорости подачи F100. Слово G01, задающее отработку перемещений с заданной скоростью по прямой (линейная интерполяция), должно быть задано в каждом кадре, в котором программируется перемещение по прямой. В кадре N1 запрограммировано перемещение из начала координат в точку P₁ с

координатами $X=21$ мм и $Y=33$ мм. Аналогично формируются перемещения в остальных кадрах, т.е. указываются координаты точки, в которую должен переместиться рабочий орган в конце кадра. В кадре N5 кроме того имеется слово M02, указывающее на завершение отработки и останов путем выключения производственного оборудования (см. табл. П4.2).

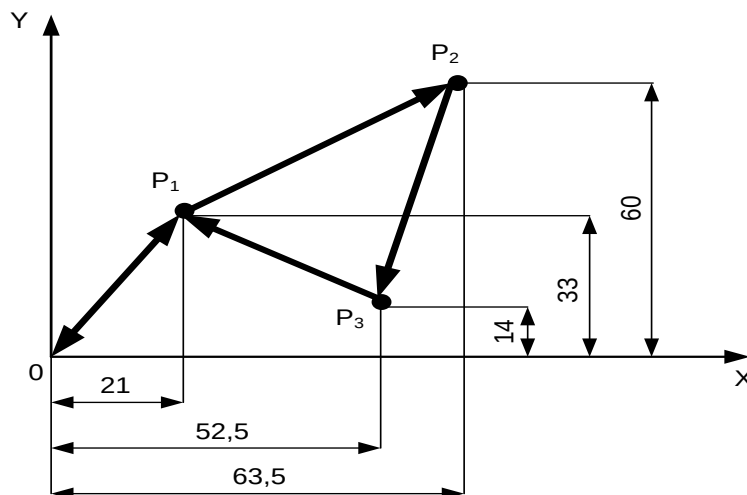


Рис. 7.4. Траектория движения расчетной рабочей точки

Та же программа, записанная в относительных размерах, т.е. когда начало координат устанавливается в точке, в которой находится механизм перед началом отработки кадра, имеет вид:

%1					LF
N1	G91	G01	X 21000	Y33000	F 100 LF
N2		G01	X 42500	Y27000	LF
N3		G01	X-11000	Y-46000	LF
N4		G01	X-31500	Y+19000	LF
N5		G01	X-21000	Y-33000	M 02 LF

Рассмотрим также примеры программирования перемещений по окружности, которые реализуются путем круговой интерполяции.

Для задания круговой интерполяции в кадре программы должны быть указаны (см. табл. П4.1):

- направление обхода дуги по или против часовой стрелке (G02 или G03);
- плоскость обработки (G17, G18, G19);
- проекции на оси плоскости обработки от радиуса - вектора кругового участка, направленного из начальной точки P_0 в центр

круга O' (см. рис.7.5 и 7.6); если плоскость обработки – XY, то проекции радиуса - вектора – IJ; проекции задаются в относительных размерах;

- координаты конечной точки P_k в относительных (G91) или абсолютных (G90) размерах.

Если круговая интерполяция в плоскости XY программируется с заданием координат конечной точки в относительных размерах, а направление обхода контура по часовой стрелке (рис. 7.5), то данный кадр программы должен выглядеть, при условии проставления номера кадра, так:

N... G91 G17 G02 X44000Y20000 I+30000J-23000 LF

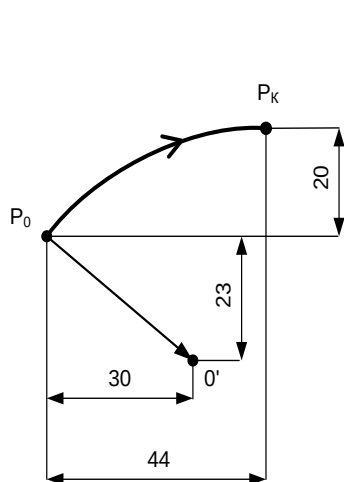


Рис. 7.5 Круговая интерполяция с заданием координат конечной точки в относительных размерах

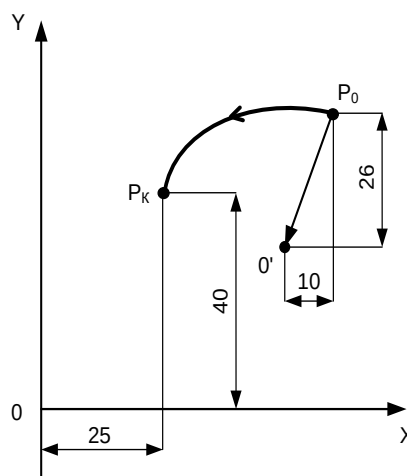


Рис. 7.6 Круговая интерполяция с заданием координат конечной точки в абсолютных размерах

При необходимости задания круговой интерполяции против часовой стрелки с программированием координат конечной точки в абсолютных размерах (рис. 7.6), соответствующий кадр программы должен иметь такой вид:

N...G90 G17 G03 X25000Y40000 I-10000J-26000 LF

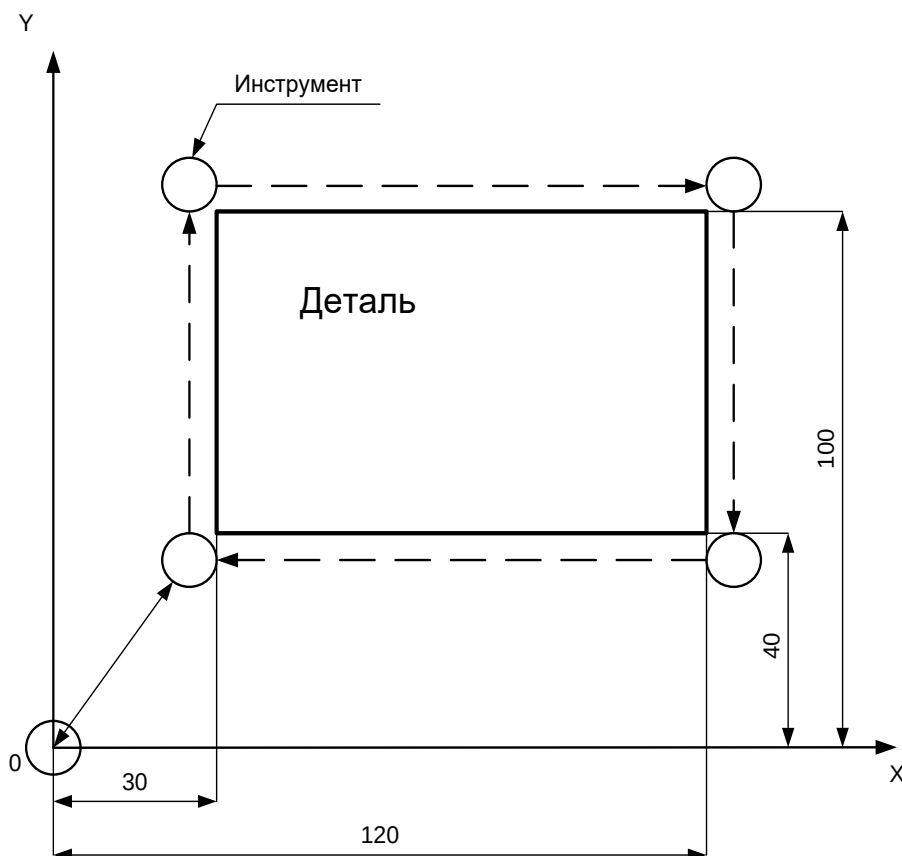


Рис. 7.7 Обработка с учётом коррекции на радиус фрезы

В УП траектория центра режущего инструмента (эквидистанта) должна быть сдвинута относительно заданного в чертеже контура детали на величину радиуса фрезы или радиуса закругления режущей кромки резца. Поэтому в управляющую программу надо вводить коррекцию (т.е. поправку) на радиус инструмента, что позволяет составлять УП непосредственно по чертежу независимо от размеров инструмента.

Коррекция положительная (отрицательная) задается функцией G43 (G44) и словом D, задающим величину коррекции, которые записываются в указанной выше последовательности перед каждой корректируемой координатой.

Отмена коррекции производится функцией G40 или заданием D00.

Функция G40 отменяет все виды коррекции по всем координатам, а слово D00 отменяет коррекцию только по той координате, перед которой оно задано, и не аннулирует задание G43 (G44).

Коррекция отрицательна (G44), если координата центра инструмента меньше координаты контура. В противном случае устанавливают положительную коррекцию G43. Приведем пример

программирования обработки с учетом коррекции на радиус инструмента в соответствии с рис. 7.7 (см. табл. П4.1):

N...00	G90	G00	G44D01X30000	G44D01Y40000	LF
N...01		G01	G43	Y100000 F1000	LF
N...02		G01	G43	X120000	LF
N...03		G01	G44	Y40000	LF
N...04		G01	G44	X30000	LF
N...05		G00	D00 X0	D00 Y0	LF

Коррекция на радиус инструмента D01=5000.

7.6. Исполнительные устройства УЧПУ

Исполнительные устройства СПУ реализуют согласованное движение рабочих органов, заданное управляющей программой, а также различные переключения исполнительных механизмов. Системы управления исполнительными механизмами делятся, в зависимости от характера и способа задания реализуемого движения, на цикловые, позиционные и контурные.

Цикловые системы управления обеспечивают лишь включение и выключение исполнительных механизмов в зависимости от номера реализуемого такта в составе технологического цикла и от номера рабочей точки, по достижении которой совершаются заданные переключения. Приход в заданную точку фиксируется электромеханическими упорами (путевыми переключателями), информация от которых поступает в управляющее устройство и определяет дальнейшее распределение управляющих сигналов в рамках текущего такта управления. При чисто цикловом управлении положение рабочих органов фиксируется лишь в отдельных точках траектории их движения, различаемых по номерам этих точек. При необходимости фиксируется также и время пребывания исполнительных механизмов в заданном положении. Траектории перемещения рабочих органов от одной фиксированной точки к другой при цикловом управлении жестко определяются конструкцией исполнительных устройств и не могут быть изменены произвольно.

Позиционные системы управления обеспечивают помимо всего того, что реализуют цикловые системы управления, также и вывод рабочих органов в произвольную точку пространства в зоне обслуживания и фиксирование (позиционирование) рабочих органов в заданной точке с заданной точностью. При позиционировании траектория перемещения рабочих органов в пределах зоны обслуживания может быть задана произвольно, а точность отработки перемещения задается лишь в окрестности точки позиционирования.

Контурные системы управления, помимо реализации описанных выше функций обеспечивают перемещение рабочих органов по

заданному контуру с заданной точностью обработки контура. Для этого контур намечается в виде совокупности опорных точек, а перемещение рабочих органов реализуется от точки к точке путем линейной или круговой интерполяции (см. §7.4). Для исполнительных устройств УЧПУ характерна реализация как позиционного, так и контурного управления.

В качестве силовой части исполнительных устройств УЧПУ применяются электрические, гидравлические и пневматические приводы, следящие или шаговые, причем применение электроприводов непрерывно расширяется. Это происходит благодаря улучшению их массогабаритных показателей, лучшим эксплуатационным характеристикам и КПД и созданию электромеханизмов, способных с высокой эффективностью воспроизводить заданные пространственные траектории. Наибольшее распространение среди исполнительных устройств УЧПУ получили следящие электроприводы (СЭП) постоянного и переменного тока.

В электродвигателях постоянного тока, использующих беспазовые малоинерционные двигатели и имеющих повышенные требования к качеству воспроизведения движения, широко применяются транзисторные широтно-импульсные преобразователи. Применение следящих приводов этого типа характерно для роботов-манипуляторов и станков малой мощности. Функциональная схема такого привода приведена на рис. 7.8.

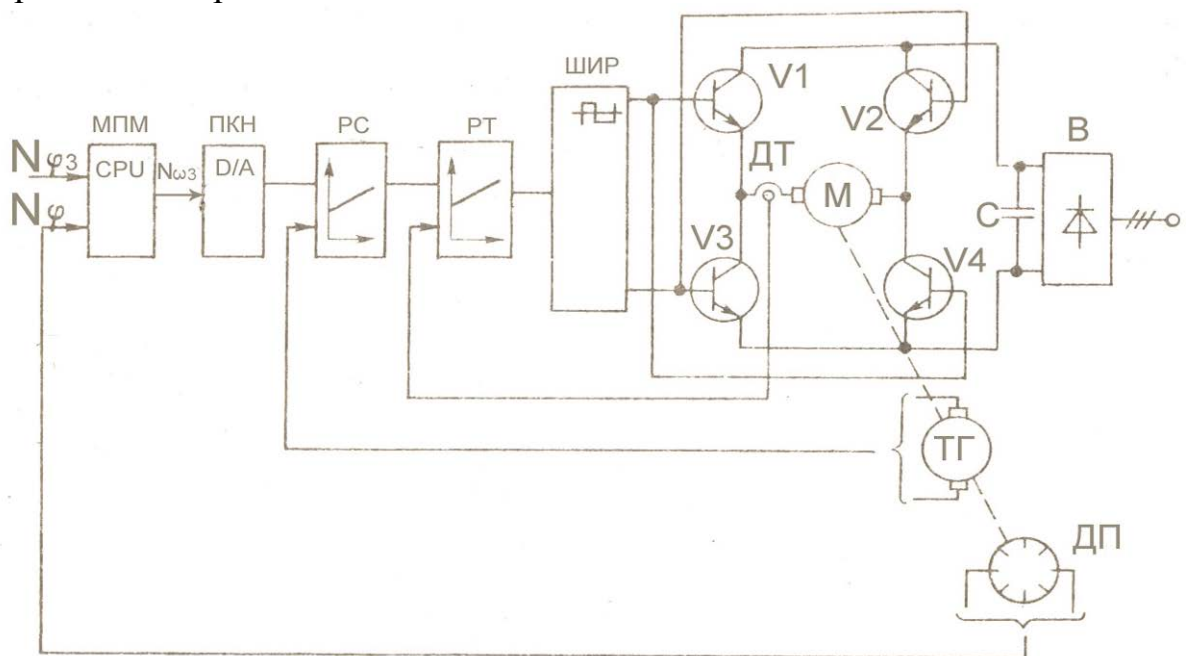


Рис.7.8. Следящий электропривод постоянного тока для роботов

Здесь транзисторные модули V1- V4, питаемые от нерегулируемого выпрямителя В с емкостным фильтром С, составляют силовую часть широтно-импульсного преобразователя, питающего

двигатель постоянного тока М. Коммутация вентилях V1- V4 определяется частотой и скважностью импульсов, поступающих с широтно- импульсного регулятора ШИР. Скважность импульсов на выходах ШИР определяется уровнем напряжения на его входе и задает скважность напряжения, поступающего на двигатель М. Чем больше скважность, тем меньше средняя величина напряжения, питающего якорь двигателя М.

Система управления приводом включает в себя цифровой контур регулирования положения (причем регулятор положения запрограммирован в микропроцессорном модуле МПМ) и аналоговые контуры регулирования скорости и тока с датчиком тока ДТ и с тахогенератором ТГ в качестве датчика скорости. Таким образом, мы имеем трехконтурную следящую систему с подчиненным регулированием скорости и тока якоря. В качестве регуляторов скорости и тока применены аналоговые пропорционально-интегральные регуляторы РС и РТ. Могут быть применены и цифровые регуляторы, причем их алгоритмы строятся по схеме, представленной на рис. 5.22. В случае применения цифровых регуляторов сигналы скорости и тока должны быть преобразованы в числовую форму с помощью АЦП.

В качестве датчика положения ДП в схеме рис.7.8. использован фотоэлектрический датчик, работающий в коде Грея, причем в датчик ДП встроен преобразователь кода Грея в двоичный арифметический код микропроцессорного модуля МПМ. В модуле МПМ число N_{ϕ} , определяющее положение датчика ДП, сравнивается с числом $N_{\phi з}$, определяющим заданное положение. Величина рассогласования обрабатывается в соответствии с программой регулятора положения и в результате определяется в числовом виде величина уставки скорости $N_{\omega з}$. Далее данная уставка должна поступить на вход комплектного регулируемого привода, имеющего, как показано на рис. 7.8, аналоговую систему управления. Поэтому она преобразуется цифро-аналоговым преобразователем ПКА (преобразователь код-напряжение) в аналоговую форму.

Следящие электроприводы постоянного тока имеют высокие показатели: диапазон регулирования скорости до 10^4 при относительной неравномерности движения не хуже 0,2, полосу пропускания замкнутого контура >20 Гц, высокую идентичность приводов по различным управляемым осям. Они выпускаются комплектно в аналоговом и цифровом исполнениях. Однако в настоящее время они успешно вытесняются следящими электроприводами переменного тока, двигатели которых имеют лучшие массогабаритные и стоимостные показатели.

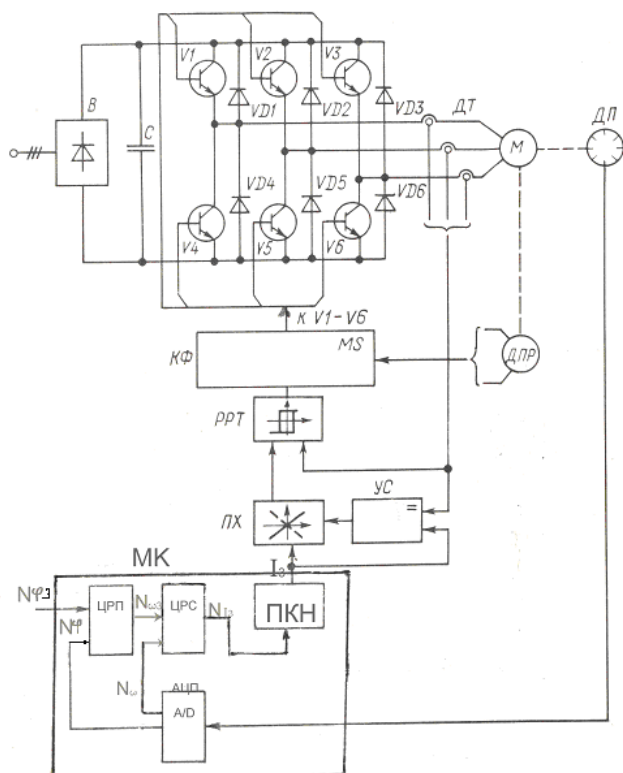


Рис.7.9. Следящий электропривод переменного тока на базе вентильного двигателя

На рис. 7.9 приведена функциональная схема СЭП переменного тока, построенного на базе вентильного двигателя, с управлением от микроконтроллера МК. Вентильные двигатели широко используются в приводах подач станков с ЧПУ и в приводах звеньев промышленных роботов. Вентильный двигатель, представленный на схеме рис. 7.9, построен на базе трехфазной синхронной машины М переменного тока с возбуждением от постоянных магнитов на роторе и на базе вентильного коммутатора, построенного на транзисторных модулях V1- V6 с подключенными антипараллельно обратными диодами VD1-VD6. Двигатель М совместно с коммутатором эквивалентен машине постоянного тока. Он получает питание от неуправляемого выпрямителя В с емкостным фильтром С на выходе. Вентильный коммутатор переключается коммутатором фаз КФ в зависимости положения ротора М сигналами датчика положения ротора ДПР. Порядок переключения вентилей V1- V6 устанавливается таким, чтобы двигатель М мог развивать максимальный для текущего положения его ротора момент. Релейный регулятор РРТ обеспечивает регулирование тока статора путем сравнения уставки заданного значения тока, поступающей от переключателя характеристик ПХ, с сигналом обратной связи по току, поступающим от датчиков фазных токов ДТ. Устройство сравнения УС и переключатель характеристик ПХ обеспечивают соответствие знаков уставки тока и фактического значения сигнала обратной связи по току.

Поскольку момент вентильного двигателя пропорционален амплитуде первой гармоники тока статора, то сигнал, задающий ток статора, тем самым задает и момент двигателя.

В СЭП по схеме рис. 7.9 сигнал обратной связи по положению формируется импульсным датчиком перемещения ДП, каждый импульс которого соответствует одной дискрете перемещения датчика. Величина перемещения N_ϕ фиксируется в числовом виде на выходе АЦП. На другом выходе АЦП формируется также в числовом виде величина скорости N_ω , полученная путем дифференцирования сигналов датчика ДП. Сигнал положения N_ϕ поступает на вход цифрового регулятора положения ЦРП, а сигнал скорости N_ω – на вход цифрового регулятора скорости ЦРС. Их алгоритмы составляются по схеме, представленной на рис.5.22. На выходах указанных регуляторов формируются в числовом виде уставка скорости $N_{\omega з}$ и уставка тока $N_{i з}$. Поскольку уставка тока должна поступать на переключатель ПХ в аналоговом виде, то на выходе ЦРС установлен цифро-аналоговый преобразователь ПЧН.

Исполнительные устройства на базе электроприводов с шаговыми двигателями реализуются в двух вариантах: с приводом от силового шагового двигателя (ШД) и с приводом от гидромотора (ГМ), управляемого от ШД и являющегося, по существу, усилителем момента (см. рис. 7.10).

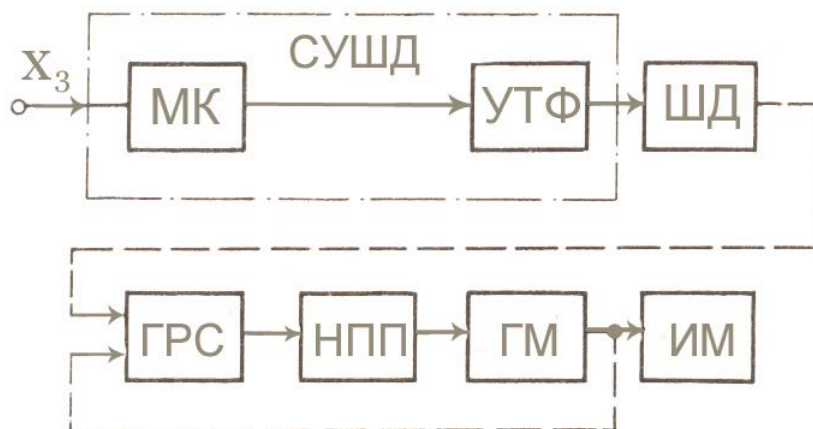


Рис.7.10. Шаговый электропривод с гидроусилителем момента

Система управления шаговым двигателем (СУШД) состоит из двух функциональных узлов: микроконтроллера МК и совокупности усилителей-регуляторов токов фаз УТФ. Микроконтроллер выполняет функции интерполятора и коммутатора фаз. Как интерполятор, он преобразует число X_z , поступившее от УЧПУ и задающее количество импульсов, поступающих на обмотки статора ШД, в реальные импульсы соответствующей частоты, причем количество импульсов равно заданному числу X_z , а их частота определяется заданной скоростью

вращения. Как коммутатор фаз, МК распределяет импульсы по фазам ШД. Перед подачей на ШД импульсы усиливаются усилителями УТФ. Каждому импульсу соответствует одна дискрета (минимальная величина) поворота вала ШД. Если применен силовой ШД, то он непосредственно вращает исполнительный механизм ИМ. В противном случае ШД поворачивает ведущий элемент гидрораспределителя ГРС, что приводит к изменению давления в полостях насоса переменной производительности НПП и гидромотора ГМ. Выходной вал ГМ поворачивает ИМ и одновременно – ведомый элемент ГРС. Это приводит к тому, что после поворота вала ГМ на заданный угол ГРС оказывается в нейтральном положении и вал ГМ останавливается.

Контрольные вопросы к главе 7

1. Что такое СПУ? Из чего складывается аппаратная часть СПУ?
2. Что такое ГПС и из каких подсистем она состоит?
3. Какова иерархическая структура ГПС?
4. Каковы основные функции локальных СПУ?
5. Каково назначение программируемых логических контроллеров (ПЛК) и УЧПУ (промышленных контроллеров)?
6. Приведите схему типичной структуры ПЛК и объясните назначение типовых модулей ПЛК.
7. Какими средствами производится программирование ПЛК?
8. Как производится адресация при программировании ПЛК по стандарту IEC 61131-3?
9. Каковы основные характеристики языков программирования ПЛК, входящих в состав стандарта IEC 61131-3?
10. Каковы основные приемы программирования на языке IL?
11. Что такое числовое программное управление (ЧПУ) и какие основные задачи решают УЧПУ?
12. Какова структура аппаратной части УЧПУ?
13. Из чего состоит и как формируется геометрическая информация в УЧПУ?
14. Что такое линейная интерполяция, круговая интерполяция и перемещение инструмента по эквидистантному контуру?
15. Как проектируется аппаратная часть УЧПУ?
16. Какова структура инструментального программного обеспечения УЧПУ и как оно используется для создания УП?
17. Как хранятся и как вводятся в действие готовые УП?
18. Каков порядок записи на программноносителе программы в коде ISO-7?
19. В каком порядке записываются информационные слова кода ISO-7 в составе кадра программы?
20. Какие функции управления реализуют информационные слова кода ISO-7, начинающимися буквами G и M?

21. Как задаются размерные перемещения в коде ISO-7?
22. Как задается формат слов УП в коде ISO- 7?
23. Приведите пример УП, задающий линейную траекторию рабочего органа в коде ISO-7.
24. Приведите пример УП, задающей круговую траекторию рабочего органа в коде ISO-7.

Литература

1. Автоматизация типовых технологических процессов и установок: Учебник для вузов.- М.Энергоатомиздат, 1988.
2. Автоматизированный электропривод промышленных установок/ Под общ. Ред. Онищенко Г.Б.: Учебное пособие для вузов.- М.:РАСХН, 2001.
3. Анашкин А. С., Кадыров Э. Д., Харазов В. Г. Техническое и программное обеспечение распределенных систем управления. Учебное пособие для вузов.- СПб.: «Р-2», 2004.
4. Борисов А.М., Лях Н.Е. Автоматизация технологических процессов: Учебное пособие для вузов. Ч. 1-4.- Челябинск: Изд. ЧГТУ, 1996.
5. Гмурман В.Е. Теория вероятностей и математическая статистика: Учебное пособие для вузов.- М.: Высшая школа, 2001.
6. Гук М. Ю. Аппаратные средства локальных сетей: Энциклопедия.- СПб.: Питер, 2002.
7. Коровин Б.Г., Прокофьев Г.И., Рассудов Л.Н. Системы программного управления промышленными установками и робототехническими комплексами: Учебное пособие для вузов.- Л.: Энергоатомиздат, 1990.
8. Матвейкин В.Г. и др. Применение SCADA- систем при автоматизации технологических процессов: Учебное пособие для вузов.- М.: Машиностроение, 2000.
9. Акулич И.Л. Математическое программирование в примерах и задачах: Учебное пособие для вузов.- М. Высшая школа, 1993.
10. Обработка нечеткой информации в системах принятия решений/Борисов А.Н. и др.- М.: Радио и связь, 1989.
11. Оливер В. Г., Оливер Н. А. Компьютерные сети.- СПб.: Питер, 2005.
12. Основы автоматизации машиностроительного производства / Под ред. Соломенцева Ю.М..- М. Высшая школа, 1999.
- 13.Петров В.В., Гордеев А.А. Нелинейные сервомеханизмы.- М.: Машиностроение, 1979.
14. Толковый словарь по вычислительным системам/ Под ред. Масловского Е.К..- М.: Машиностроение, 1989.

Приложение 1. Таблицы кодов обмена информацией в АСУТП

Таблица П1.1. Двоичное кодирование символов ASCII

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

Таблица П1.2.Кодирование символов КОИ-8 кодами, записанными 16-ричными цифрами

Младшие цифры	Старшие цифры кода															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	ПУС	AP1	Пробел	0	@	P	/	p	ВЦФ			ю	п	Ю	П	
1	НЗ	СУ1	!	1	A	Q	a	q	НЗН			а	я	А	Я	ё
2	НТ	СУ2	"	2	B	R	b	r	РП	УУК		б	р	Б	Р	
3	КТ	СУ3	#	3	C	S	c	s				ц	с	Ц	С	
4	КП	СТП	\$	4	D	T	d	t	БК	ВКП		д	т	Д	Т	
5	КТМ	НЕТ	%	5	E	U	e	u	НС	ОСУ		е	у	Е	У	
6	ДА	СИН	&	6	F	V	f	v	НП	ВП		ф	ж	Ф	Ж	
7	ЗБ	КБ	'	7	G	W	g	w	ОЖД			г	в	Г	В	
8	ВШ	АН	(	8	H	X	h	x				х	ь	Х	Ь	
9	ГТ	КН	)	9	I	Y	i	y				и	ы	И	Ы	
A	ПС	ЗН	*	:	J	Z	j	z	УР			й	з	Й	З	
B	ВТ	AP2	+	;	K	[	k	{	СП2	СП3		к	ш	К	Ш	
C	ПФ	РФ	,	<	L	\	l			ВЫП		л	э	Л	Э	
D	ВК	РГ	-	=	M	]	m	}		ВСТ		м	щ	М	Щ	
E	ВЫХ	РЗ	.	>	N	^	n	~	НРВ			н	ч	Н	Ч	
F	ВХ	РЭ	/	?	O	_	o	ЗБ	СП1			о		О	ЗБ	

Таблица П1.3. Управляющие сигналы в составе кодов КОИ-8 и ASCII

Код	Символы		Наименование символа	Код	Символы		Наименование символа
	КОИ-8	ASCII			КОИ-8	ASCII	
0	ПУС	NUL	Пустой символ	1A	ЗМ	SUB	Замена
1	НЗ	SOH	Начало заголовка	1B	AP2	ESC	Авторегистр 2 (возврат)
2	НТ	STX	Начало текста	1C	РФ	FS	Раздел. файлов
3	КТ	ETX	Конец текста	1D	РГ	GS	Раздел. групп
4	КП	EOT	Конец передач	1E	РЗ	RS	Разд. надписей
5	КТМ	ENQ	Кто там?	1F	РЭ	US	Разд. элементов
6	ДА	ACK	Подтверждение	7F	ЗБ	DEL *	Забой, стирание
7	ЗВ	BEL	Звонок (сигнал)	80	ВЦФ	DS	Выбор цифры
8	ВШ	BS	Возврат на шаг	81	НЗН	SOS	Нач. значимост
9	ГТ	HT	Горизонт. таб.	82	РП	FS	Раздел. полей
0A	ПС	LF	Перевод строки	84	БК	BYP	Блокировка
0B	ВТ	VT	Вертик. таб.	85	НС	NL	Новая строка
0C	ПФ	FF	Перевод формата	86	НП	LC	Нижнее положение
0D	ВК	CR	Возвр. каретки	87	ОЖД	IL	Ожидание
0E	ВЫХ	S0	Выход (на верхн. регистр)	8A	УР	SM	Установление режима
0F	ВХ	SI	Вход (на нижн. регистр)	8B	СП2	CU2	Символ пользователя 2
10	AP1	DLE	Авторегистр 1 (переключение)	8E	НРВ	SMM	Начало ручного ввода
11	СУ1	DC1	Сигнал управления 1	8F	СП1	CU1	Символ пользователя 1

12	СУ2	DC2	Сигнал упр. 2	92	УУК	CC	Управл. указат.
13	СУ3	DC3	Сигнал упр. 3	94	ВКП	PN	Вкл. перфорат.
14	СТП	DC4	Стоп (Сигнал управления 4)	95	ОСУ	RS	Остановка считывания
15	НЕТ	NAK	Отрицание подтверждения	96	ВП	UC	Верхнее положение
16	СИН	SYN	Синхронизация	9B	СПЗ	CU3	Симв. польз. 3
17	КБ	ETB	Конец блока	9C	ВЫП	PF	Выключение перфоратора
18	АН	CAN	Аннулирование	9D	ВСТ	RES	Восстановление
19	КН	EM	Конец носителя				

* Символы, помещенные в данной колонке ниже символа DEL, не входят в состав ASCII

П.2.1. Случайные события и их вероятность

Во многих случаях происходящие события представляются нам случайными. К ним относятся: разброс показаний измерительной аппаратуры, показателей качества продукции, отказ оборудования и прочее. Простейшая оценка случайного события производится путём определения его *относительной чистоты*:

$$h(E)=n_E/n, \quad (\text{П2.1})$$

где n_E – число реализаций исходов события E за n испытаний. Если в различных сериях испытаний значения $h(E)$ получаются близкими друг к другу, то говорят, что величина $h(E)$ обладает *статистической устойчивостью*. Относительные частоты $h(E)$ статистически устойчивых серий событий группируются около определённого числа, которое называется *вероятностью события E* и обозначается, как $P(E)$. Если количество испытаний можно рассматривать как неограниченные, то вероятность события E можно представить в виде предела отношения

$$P(E)= \lim_{n \rightarrow \infty} n_E/n. \quad (\text{П2.2})$$

События, не обладающие статистической устойчивостью, теорией вероятностей не рассматриваются.

Наличие статистической устойчивости случайных событий может быть проверено только опытным путём.

П.2.2. Основные свойства вероятностей

Из соотношений (П2.1) и (П2.2) с учётом того, что $0 \leq n_E \leq n$, следует

$$0 \leq P(E) \leq 1,$$

т.е. *вероятность случайного события заключена между 0 и 1.*

Если всегда имеет место $n_E = n$, то $P(E)=1$. Событие, вероятность которого равна единице, будем называть *достоверным*, а событие, вероятность которого равна нулю ($P(E)=0$, $n_E = 0$), будем называть *невозможным*.

*Суммой событий A и B называют такое событие C , которое состоит в осуществлении хотя бы одного из них в одном и том же испытании. Если события A и B не могут произойти вместе при одном и том же испытании, то они называются *несовместимыми*. Вероятность суммы n попарно несовместимых событий равна сумме вероятностей этих событий:*

$$P(A_1 + A_2 + \dots + A_n) = P(A_1) + P(A_2) + \dots + P(A_n). \quad (\text{П2.3})$$

Случайные события образуют *полную группу*, если они попарно несовместимы, но при каждом повторении испытаний обязательно происходит одно из них. Последнее ведёт к тому, что сумма вероятностей событий, образующих полную группу, равна единице.

Два случайных события называются *противоположными*, если реализация одного из них гарантирует неосуществление другого.

Нетрудно заметить, что противоположные события образуют полную группу, откуда следует:

$$P(\bar{A}) = 1 - P(A) \quad (\text{П2.4})$$

где $P(\bar{A})$ – вероятность события $\bar{A}$, противоположного A .

Произведением двух случайных событий называется произведение вероятности одного из них на условную вероятность другого, причём *условной вероятностью* $P(B|A)$ называют вероятность события B при условии осуществления события A . Формула произведения вероятностей, которое трактуется как *вероятность совмещения* двух событий, выглядит так:

$$P(AB) = P(A)P(B|A). \quad (\text{П2.5})$$

Следствием формулы (П2.5) является следующая формула вероятности суммы двух *совместимых* событий A и B , т.е. такие, у которых $P(A) \neq 0$ и $P(B|A) \neq 0$:

$$P(A+B) = P(A) + P(B) - P(AB). \quad (\text{П2.6})$$

Если условные вероятности случайных событий равны их безусловным вероятностям, то такие события называются *независимыми*. Вероятность совмещения (или совпадения) независимых событий определяется произведением вероятностей составляющих событий:

$$P(ABC...N) = P(A)P(B)P(C)...P(N). \quad (\text{П2.7})$$

П.2.3. Дискретные случайные величины

При оценке интересующих нас событий часто оказывается, что их исходы характеризуются различными числами, являющимися *значениями* интересующих нас величин. Однако, эти значения либо равны, либо отличаются друг от друга на величину, которая не может быть признана бесконечно малой, так как ограничена снизу дискретностью шкалы измерений, или каким-либо иным способом. Так, значения очков, подсчитываемых при выбрасывании игральных костей шестигранной формы, не могут быть меньше единицы на одной грани игральной кости. Равным образом, при дискретности измерений в 1В далеко не всегда разность напряжений в 1В может быть признана бесконечно малой и имеющей пренебрежимо малую вероятность реализации. Подобные рассуждения приводят к определению:

величина X есть дискретная случайная величина, если множество её значений образует последовательность различных чисел $x_1, x_2, \dots, x_i, \dots, x_n$ и есть каждое событие $X=x_i$ является случайным событием, характеризующимся вероятностью p_i .

События $X=x_i$ будем называть *составляющими событиями*, а *закон распределения* случайной величины X будем называть *правилом*, позволяющее находить любые вероятности $p_i(x_i)$. Если составляющие события образуют *полную группу*, то имеем:

$$\sum_{i=1}^n p_i = 1 \quad (\text{П2.8})$$

Закон распределения дискретной случайной величины зависит от характера событий, в процессе которых она выявляется. Если случайная величина X выявляется только в процессе заранее запланированных испытаний таких, как выборка шаров того, или иного цвета из урны, или взятие проб с целью определения качества изделий, то получается *биномиальное распределение*. Если же случайная величина связана с простейшим потоком случайных событий, то получается *распределение Пуассона*.

П.2.4. Биномиальное распределение

Биномиальное распределение связано с n -кратным повторением одного и того же испытания с целью обнаружения случайного события E , имеющего вероятность реализации $P(E)=p$. Полагают, что в результате проведения испытаний величина P не изменяется. Порядок проведения испытаний, который обеспечивает неизменность P , называют *независимыми испытаниями по схеме Бернулли*, или *схема повторной выборки*. Скажем, если из урны вынимают наугад один шар и проверяют на наличие какого-либо признака, то по схеме Бернулли шар после проверки следует вернуть в урну, а затем все шары следует тщательно перемешать. Вероятность того, что событие E не реализуется, определяется как

$$q=(1-p). \quad (\text{П2.9})$$

После проведения каждой серии из n испытаний событие E может быть фактически реализовано X раз, причём величина X может принимать значения от нуля до n . Биномиальное распределение даёт закон распределения случайной величины X после каждой серии из n выше описанных испытаний. При этом полагают, что сложное событие $X=m$ состоит в том, что за n испытаний случайное событие E наступит ровно m раз, а противоположное событие P наступит $n-m$ раз. Биномиальное распределение описывается соотношением

$$P(X=m)=C_n^m p^m q^{n-m}, \quad (\text{П2.10})$$

где $C_n^m=n!/m!(n-m)!$ – число сочетаний из n элементов по m , а $m=0,1,2,\dots,n$.

Данный закон распределения называется биномиальным, так как вероятности подсчитаны по формуле (П2.10) и соответствуют членам разложения бинома

$$(q+p)^n \quad (\text{П2.10}')$$

по степеням p .

Заметим, что если количество объектов, из которых производится выборка, много больше n , то распределение вероятностей X достаточно близко к биномиальному распределению и в случае *бесповторной выборки*, когда используемые объекты (изделия) после проведения испытаний не возвращаются в состав проверяемой партии.

П.2.5. Распределение Пуассона

Распределение Пуассона связано с процессами, характеризующимися простейшим потоком событий. *Потоком событий* будем называть последовательность событий, наступающих одно за другим в случайные моменты времени. Примерами потоков событий могут служить потоки заявок в системах массового обслуживания: на телефонной станции, заявки на ремонт, заказы на покупку товаров и прочее, а также поток бракованных изделий, или поток отказов при эксплуатации оборудования. *Простейший поток событий* отличается отсутствием последдействия, а именно:

а) вероятность наступления тех, или иных событий в простейшем потоке событий за любой промежуток времени зависит только от длительности этого промежутка (а не от начала отсчёта);

б) указанная вероятность не зависит от того, какое число событий имело место до начала рассматриваемого промежутка времени.

Кроме того, в простейшем потоке событий вероятность наступления очередного события в течение малого промежутка времени можно считать пропорциональной длительности этого промежутка с коэффициентом пропорциональности λ . Коэффициент λ является параметром потока. Его обычно определяют как изменение вероятности наступления события в единицу времени. Учитывая, что вероятность реализации события определяется соотношением

$$P = r / N_{N \rightarrow \infty}, \quad (\text{П2.11})$$

где N – общее количество объектов, с которыми могут происходить интересующие нас события;

r – количество фактически происшедших событий в рассматриваемый промежуток времени, то параметр λ может быть определен как:

$$\lambda = \Delta r / (N \Delta t), \quad (\text{П2.12})$$

причем Δr – есть количество событий, происшедших в течение времени Δt . Величина λ есть *удельная частота* реализации интересующих нас событий.

Если промежуток Δt настолько мал, что за это время может произойти только одно событие ($\Delta r = 1$), то вероятность однократной реализации события будет равна:

$$P_1 = \lambda \Delta t + \theta_1, \quad \text{где } \lim (\theta_1 / \Delta t) \rightarrow 0 \text{ при } \Delta t \rightarrow 0.$$

Соответственно, отсутствие реализации данного события в том же промежутке t выразится соотношением:

$$P_0(\Delta t) = 1 - \lambda \Delta t + \theta_0, \quad \text{где } \lim (\theta_0 / \Delta t) = 0 \text{ при } \Delta t \rightarrow 0. \quad (\text{П2.13})$$

Приняв во внимание, что отсутствие реализации события в промежутке времени $t + \Delta t$ означает совпадение отсутствия реализации его как в промежутке t , так и за промежуток Δt , по правилу умножения вероятностей получим:

$$P_0(t + \Delta t) = P_0(t) P_0(\Delta t),$$

что с учетом (П2.13) дает:

$$P_0(t + \Delta t) = P_0(t) (1 - \lambda \Delta t + \theta_0) \text{ и} \\ (P_0(t + \Delta t) - P_0(t)) / \Delta t = P_0(t) (-\lambda + \theta_0 / \Delta t).$$

Далее, переходя к пределу при $\Delta t \rightarrow 0$, приходим к дифференциальному уравнению

$$dP_0/dt = -\lambda P_0,$$

решением которого (при начальном условии $P_0(0)=1$) является соотношение

$$P_0(t) = e^{-\lambda t}. \quad (\text{П2.14})$$

Уравнение (П2.14) является уравнением вероятности отсутствия интересующих нас событий, в частности, вероятности безотказной работы оборудования в течении времени t при простейшем потоке отказов, когда параметр потока стационарен ($\lambda = \text{const}$).

Если же необходимо определить вероятность сложного события X , заключающегося в том, что за промежуток времени t произойдет именно m интересующих нас событий в составе простейшего потока, то применяют *формулу распределения Пуассона*:

$$P(X=m) = (a^m / m!) e^{-a}, \quad (\text{П2.15})$$

в которой a – это среднее число событий, приходящихся в течении времени t на один обслуживаемый (или наблюдаемый) объект.

Если известен параметр потока λ , то имеем

$$a = \lambda t, \quad (\text{П2.15'})$$

так что показательное распределение (П2.14) есть частный случай распределения Пуассона при $m=0$.

В заключение напомним, что, в отличие от распределения Пуассона, биномиальное распределение характеризуется задаваемым заранее параметром n - количество проводимых испытаний. Однако, если значение n достаточно велико ($n > 20$), а значение p , фигурирующее в формуле (П2.10), достаточно мало ($p < 0,1$), то можно применить формулу распределения Пуассона, считая, что за время проведения n испытаний получится

$$a = np. \quad (\text{п 2.15"})$$

В этих условиях (и при умеренных значениях a , скажем, $a < 4$) биномиальное распределение приближается к распределению Пуассона ("асимптотическое приближение") и формула (П2.10) может быть заменена более простой формулой (П2.15).

П.2.6. Непрерывные случайные величины. Плотность распределения вероятностей

□□□ Расчет вероятности случайной величины (события) E базируется на подсчете числа реализаций данного события в процессе испытаний (см. §П2.1). Критерии оценки реализации того или иного события всегда дискретны, так как ограничены снизу дискретностью принятой шкалы измерений (см. §П2.3). В тех случаях, когда можно пренебречь дискретностью измерений интересующей нас случайной величины X , удобно представлять величину X как изменяющуюся непрерывно в некотором интервале (α, β) . Закон распределения вероятностей непрерывной случайной величины X должен позволять находить вероятность ее попадания в любой интервал (x_1, x_2) внутри интервала (α, β) .

Для облегчения дальнейших рассуждений введем понятие *функции случайной величины* X . Это такая случайная величина Y , которая принимает значение $y=f(x)$ всякий раз, когда величина X принимает значение x . Вероятность попадания величины X в интервал (x_1, x_2) обозначим через $P(x_1 < X < x_2)$.

Далее будем считать, что вероятность попадания значения X в интервал $(x, x+\Delta x)$ малой длины $\Delta x > 0$ пропорциональна Δx с точностью до бесконечно малых более высокого, чем Δx , порядка, т.е.:

$$P(x < X < x + \Delta x) = p(x)\Delta x + \theta(\Delta x),$$

или $dP = p(x)dx$,

причем $\theta(\Delta x)/\Delta x \rightarrow 0$, $\Delta x \rightarrow dx$ при $\Delta x \rightarrow 0$.

Отсюда, переходя к пределу при $\Delta x \rightarrow 0$, находим точное значение функции $p(x)$:

$$p(x) = \lim_{\Delta x \rightarrow 0} (P(x < X < x + \Delta x) / \Delta x) \quad (\text{П2.16})$$

Функция $p(x)$ есть плотность распределения вероятностей случайной величины X .

Исходя из выражения (П2.16), путем интегрирования элементов вероятности $p(x)dx$ находим значение вероятности попадания случайной величины X в любой интервал (x_1, x_2) :

$$P(x_1 < X < x_2) = \int_{x_1}^{x_2} p(x)dx. \quad (\text{П2.17})$$

Существование интеграла (П2.17) является доказательством непрерывности случайной величины X . Если известно распределение $P(x)$, то, согласно (П2.17), плотность распределения может быть найдена путем дифференцирования функции $P(\alpha < X < x)$, где α - минимально возможное значение X :

$$p(x) = dP/dx. \quad (\text{П2.18})$$

Функцию $P(\alpha < X < x)$ обычно отображают в виде $P(X < x)$ и называют *функцией распределения вероятности*.

Если случайная величина X существует лишь в интервале (α, β) то естественным нормирующим условием для функции $p(x)$ является

$$\int_{\alpha}^{\beta} p(x)dx = 1 , \quad (\text{П2.19})$$

что соответствует достоверности ограничения значений величины X интервалом (α, β) .

П.2.7. Числовые характеристики распределения вероятностей

Числовые характеристики распределения случайной величины подразделяются на *характеристики положения*, определяющие характерные значения случайной величины, и *характеристики рассеяния*. Из характеристик положения наиболее интересным для нас является среднее значение дискретной случайной величины, формируемое через распределение ее вероятностей аналогично среднему арифметическому. Оно задается следующим определением: *математическим ожиданием (или средним значением) MX дискретной случайной величины X называется сумма произведений всех ее возможных значений x_i на их вероятности p_i :*

$$MX = \sum_{i=1}^n x_i p_i . \quad (\text{П2.20})$$

Аналогично, *математическим ожиданием (или средним значением) MX непрерывной случайной величины X называется интеграл от произведения ее значений x на плотность распределения вероятностей $p(x)$:*

$$MX = \int_{\alpha}^{\beta} xp(x)dx , \quad (\text{П2.21})$$

где α и β – начальное и конечное значения X на интервале (α, β) , на котором сосредоточены все возможные значения величины X .

Аналогия формул (П2.20) и (П2.21) становится очевидной, если принять во внимание, что здесь элемент вероятности $p(x)dx$ аналогичен вероятности p_i , а интеграл (П2.21) – это сумма бесконечно малых слагаемых $x p(x)dx$.

Соответственно, математическое ожидание функции $Y = f(x)$ от случайной величины X выражается через закон распределения самой величины X :

а) для дискретной величины X :

$$Mf(x) = \sum_{i=1}^n f(x_i) p_i ; \quad (\text{П2.20'})$$

в) для непрерывной величины X :

$$Mf(x) = \int_{\alpha}^{\beta} f(x) p_x(x) dx . \quad (\text{П2.21'})$$

Математическое ожидание называют также *центром распределения случайной величины*, так как ее рассеяние определяется относительно математического ожидания.

Основные свойства математических ожиданий:

1) постоянный множитель можно выносить за знак математического ожидания:

$$MCX = CMX;$$

2) математическое ожидание суммы случайных величин равно сумме их математических ожиданий

$$M(x_1 + x_2 + \dots + x_n) = MX_1 + MX_2 + \dots + MX_n;$$

3) математическое ожидание произведения независимых случайных величин равно произведению их математических ожиданий:

$$M(XY) = MX MY.$$

Важнейшими числовыми характеристиками рассеяния случайной величины являются *дисперсия* и *среднеквадратическое отклонение*. Последнее определяется по формуле:

$$\sigma(X) = \sqrt{M(x-a)^2}, \quad (\text{П2.22})$$

где a – математическое ожидание (среднее значение) случайной величины X ;

$M(X-a)^2$ – среднее значение квадрата отклонения X от среднего значения a .

Подкоренное выражение в формуле (П2.22) часто употребляется самостоятельно и потому получило название *дисперсии случайной величины X* :

$$DX = M(X-a)^2. \quad (\text{П2.23})$$

Выражение (П2.23) можно также записать в форме

$$DX = MX^2 - (MX)^2. \quad (\text{П2.23'})$$

Согласно соотношения (П2.20') дисперсия дискретной величины равна

$$DX = \sum_{i=1}^n (x_i - a)^2 p_i, \quad (\text{П2.24})$$

а согласно соотношения (П2.21') дисперсия непрерывной величины равна

$$DX = \int_{\alpha}^{\beta} (x-a)^2 p(x) dx, \quad (\text{П2.25})$$

где α и β – крайние точки интервала существования X .

Основные свойства характеристик рассеяния:

1) у линейной функции случайной величины X :

$$Y = kX + b,$$

дисперсия увеличивается в k^2 раз, а среднеквадратическое отклонение – в $|k|$ раз;

2) если случайные величины X и Y независимы, то дисперсия их суммы равна сумме их дисперсий:

$$D(X+Y) = DX + DY$$

и соответственно –

$$\sigma(X+Y) = \sqrt{\sigma^2(X) + \sigma^2(Y)}.$$

В частности, дисперсия среднего арифметического n случайных величин, имеющих одинаковую дисперсию, составляет σ^2/n , так что среднеквадратическое отклонение в это случае равно:

$$\sigma((X_1 + X_2 + \dots + X_n)/n) = \sigma/\sqrt{n}. \quad (\text{П2.26})$$

При испытаниях, заключающихся в проверке наличия или отсутствия какого-либо признака и ведущих к *биномиальному распределению* (см. §П2.4), наличие признака можно обозначить как $x=1$, а его отсутствие, как $x=0$. Если вероятность наличия признака равна p , то математическое ожидание при биномиальном распределении окажется равным, согласно (П2.20),

$$Mx = np. \quad (\text{П2.26}')$$

Обозначив вероятность отсутствия признака через q и исходя из соотношения (П2.24), получим в случае однократной проверки наличия признака дисперсию:

$$DX_1 = (1-p)^2 p + (0-p)^2 q = q^2 p + p^2 q = pq.$$

В общем случае биномиального распределения (когда производится n -кратная выборка), получим:

$$DX = npq. \quad (\text{П2.27})$$

Если имеется простейший поток событий, ведущий к распределению Пуассона (см. §П2.5), формула (п 2.15), то его математическое ожидание можно вывести непосредственно из формулы (П2.20); полагая, что $X=m$, $m \geq 0$. Получим:

$$MX = \sum_{m=0}^{\infty} m \frac{a^m}{m!} e^{-a} = e^{-a} a \sum_{m=1}^{\infty} \frac{a^{m-1}}{(m-1)!} = a.$$

Если $a=\lambda t$ при $\lambda = \text{const}$, то a – это среднее число событий простейшего потока, наступающих за заданное время t и приходящихся на один наблюдаемый объект, что было показано с других позиций уже в §П2.5. Аналогичными рассуждениями можно показать, что дисперсия распределения Пуассона совпадает с его математическим ожиданием:

$$DX = a.$$

П2.8. Непрерывные законы распределения и их числовые характеристики

Обратимся, прежде всего, к *показательному распределению*, которое мы уже начали обсуждать (см. формулу (П2.14) в §П2.5) при рассмотрении распределения Пуассона, оценивающего вероятности событий, происходящих в простейшем потоке событий. Эти события происходят дискретно, но промежуток времени T между двумя соседними событиями в простейшем потоке можно считать непрерывной величиной. Отсюда возникает задача определения вероятности того, что значение $T < t$, где $t > 0$ – время, прошедшее

после предыдущего события. Из соотношения (П2.14) следует, что эта вероятность равна

$$P(T < t) = 1 - e^{-\lambda t}, \quad (\text{П2.28})$$

где λ – параметр потока, см. формулу (П2.12).

Функцию (П2.28) называют функцией показательного распределения вероятностей. Если интересующие нас события являются отказами оборудования, то $P(T < t)$ является функцией распределения вероятностей отказа, а величина T – текущим значением времени наработки до отказа. Дифференцируя функцию $P(T < t)$, представленную соотношением (П2.28), находим плотность распределения вероятностей величины T :

$$P(t) = \lambda e^{-\lambda t} \quad \text{при } t > 0. \quad (\text{П2.29})$$

Математическое ожидание величины T можно вычислить непосредственно по формуле (П2.21):

$$MT = \int_0^{\infty} t \lambda e^{-\lambda t} dt = \frac{1}{\lambda}. \quad (\text{П2.30})$$

Если нас интересуют отказы оборудования, то формула (П2.30) дает среднюю наработку до отказа при условии, что λ – это удельная частота простейшего потока отказов.

Дисперсия показательного распределения, согласно соотношения (П2.23'), равна

$$DT = \frac{1}{\lambda^2}, \quad (\text{П2.31})$$

а, следовательно, среднеквадратичное отклонение при показательном распределении равно математическому ожиданию.

Другим важнейшим законом распределения вероятностей является *нормальный закон распределения*. При нормальном распределении плотность распределения случайной величины X рассчитывается по формуле:

$$\varphi(x) = \frac{1}{\sigma \sqrt{2\pi}} \exp \left[-\frac{(x-a)^2}{2\sigma^2} \right], \quad (\text{П2.32})$$

где a – центр распределения (математическое ожидание),

σ – среднеквадратичное отклонение.

С помощью подстановки значений

$$u = \frac{x-a}{\sigma} \quad \text{и} \quad du = \frac{dx}{\sigma}$$

соотношение (П2.32) приводится к виду *стандартного нормального распределения* с параметрами $a = 0$ и $\sigma = 1$:

$$\varphi(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}. \quad (\text{П2.33})$$

Чтобы найти функцию нормального распределения, необходимо взять интеграл от $\varphi(u)$. Поскольку он не выражается через элементарные функции, то в расчетах, связанных с нормальным

распределением, используются таблицы кратной данному интегралу функции

$$\Phi(u) = \frac{2}{\sqrt{2\pi}} \int_0^u e^{-\frac{u^2}{2}} du, \quad (\text{П2.34})$$

называемой *интегралом вероятностей* (см. табл. П2.1). Ввиду нечетности функции $\Phi(u)$, имеет место $\Phi(-u) = -\Phi(u)$, что облегчает составление справочных таблиц. Согласно (П2.34) вероятность попадания случайной величины U в любой интервал (u_1, u_2) составляет

$$P(u_1 < U < u_2) = \frac{1}{2} [\Phi(u_2) - \Phi(u_1)], \quad (\text{П2.35})$$

причем значения u вычисляются в соответствии с заданными пределами изменения исходной случайной величины X :

$$u_1 = \frac{x_1 - a}{\sigma} \quad \text{и} \quad u_2 = \frac{x_2 - a}{\sigma}. \quad (\text{П2.35'})$$

Нетрудно заметить, что для симметричного интервала $(-u, u)$ получается:

$$P(-u < U < u) = \Phi(u).$$

Нормальный закон распределения характерен для процессов, в которых разброс параметров вызывается действием большого количества факторов, воздействие каждого из которых в отдельности трудно учесть ввиду его незначительности. Нормальное распределение характерно для распределения случайных ошибок измерений, разброса параметров производимой продукции и для многих других процессов производственного и непроизводственного характера. В частности, при испытаниях по схеме Бернулли, ведущих к биномиальному распределению (см. §П2.4), биномиальное распределение стремится к нормальному с увеличением n , числа испытаний. Это позволяет в практических расчетах, в том числе в расчетах надежности, применять формулы нормального распределения и тем значительно упростить расчеты.

Таблица П2.1

$$\text{Интеграл вероятностей} \quad \Phi(u) = \frac{2}{\sqrt{2\pi}} \int_0^u e^{-\frac{u^2}{2}} du$$

u	Φ(u)	u	Φ(u)	u	Φ(u)
0,00	0,0000 399	1,00	0,6827 236	2,00	0,9545 51
0,05	0,0399 398	1,05	0,7063 224	2,05	0,9596 47
0,10	0,0797 395	1,10	0,7287 212	2,10	0,9643 41
0,15	0,1192 393	1,15	0,7499 200	2,15	0,9684 38
0,20	0,1585 389	1,20	0,7699 188	2,20	0,9722 34
0,25	0,1974 384	1,25	0,7887 177	2,25	0,9756 30
0,30	0,2358 379	1,30	0,8064 166	2,30	0,9786 26
0,35	0,2737 371	1,35	0,8230 155	2,35	0,9812 24
0,40	0,3108 365	1,40	0,8385 144	2,40	0,9836 21
0,45	0,3473 356	1,45	0,8529 135	2,45	0,9857 19
0,50	0,3829 348	1,50	0,8664 125	2,50	0,9876 16
0,55	0,4177 338	1,55	0,8789 115	2,55	0,9892 15
0,60	0,4515 328	1,60	0,8904 107	2,60	0,9907 13
0,65	0,4843 318	1,65	0,9011 98	2,65	0,9920 11
0,70	0,5161 306	1,70	0,9109 90	2,70	0,9931 9
0,75	0,5467 296	1,75	0,9199 82	2,75	0,9940 9
0,80	0,5763 284	1,80	0,9281 76	2,80	0,9949 7
0,85	0,6047 272	1,85	0,9357 69	2,85	0,9956 7
0,90	0,6319 260	1,90	0,9426 62	2,90	0,9963 5
0,95	0,6579 248	1,95	0,9488 57	2,95	0,9968

Примечание к таблице П2.1. Числа, напечатанные между строками таблицы, являются разностями между соседними значениями функции $\Phi(u)$. Они приведены для облегчения интерполяции. При линейной интерполяции ошибка не превышает 0,0002 в интервале $0,4 < u < 1,8$ и 0,0001 в остальной части таблицы. Ошибка при расчете вероятностей в целом не превышает 0,0004 .

Приложение 3. Элементы булевой алгебры формальной логики

ПЗ.1. Объекты булевой алгебры

Булева алгебра представляет в символическом виде, принятом в математике, законы и соотношения формальной логики. Булева алгебра названа по имени английского математика Джорджа Буля (G. Boole, 1815-64), разрабатывавшего вопросы математической логики.

Все операции булевой алгебры производятся над математическими объектами, которые могут принимать только два значения. В общем виде эти два значения обозначают как 1 (Integer – нечто целое) и 0 (ноль – отсутствии чего-либо). При логическом анализе символы 1 и 0 применяются в смысле утверждения истинности (ДА) или ложности (НЕТ) некоторого высказывания. В технических применениях они часто указывают на наличие или отсутствие некоторого признака, а также на осуществление или не осуществление какого-либо интересующего нас события. Символы 1 и 0 называют *логическими булевыми значениями*, причем символ 1 чаще всего заменяют символом 1. В этом случае единица является не единицей счета, а обозначением одного из двух возможных значений объектов булевой алгебры.

Объекты булевой алгебры, логические переменные, принято обозначать латинскими буквами, как и в обычной алгебре. Над ними определены операции *сложения*, *умножения* и *дополнения*, чаще называемого *инверсией*.

П3.2. Операции сложения и умножения

Сложение и умножение производятся обычным порядком, но с учетом того, что в булевой алгебре употребляются только два значения алгебраических величин (два «числа»: 1 и 0).

Перечислим свойства *сложения* и *умножения* в рамках булевой алгебры.

$$\begin{aligned} 1) \text{ Коммутативность: } & A+B=B+A; \\ & AB=BA. \end{aligned} \quad (\text{П3.1})$$

$$\begin{aligned} 2) \text{ Ассоциативность: } & A+(B+C)=(A+B)+C; \\ & A(BC)=(AB)C. \end{aligned} \quad (\text{П3.2})$$

$$\begin{aligned} 3) \text{ Дистрибутивность: } & A(B+C)=AB+BC; \\ & A+BC=(A+B)(A+C). \end{aligned} \quad (\text{П3.3})$$

Второе свойство дистрибутивности не присуще обычной алгебре, но оно вытекает из свойств идемпотентности и поглощения, описанных ниже.

$$\begin{aligned} 4) \text{ Идемпотентность (равносильность):} \\ & A+A=A=AA \end{aligned} \quad (\text{П3.4})$$

Для обоснования идемпотентности, обратим внимание на то что $0+0=0$, но также должно быть $1+1=1$, так как в булевой алгебре нет чисел, больших единицы. Кроме того очевидно, что $0 \cdot 0=0$ и $1 \cdot 1=1$.

$$5) \text{ Операции с константами:} \\ A+0=A, \quad A \cdot 1=A, \quad A \cdot 0=0 \quad \text{и} \quad A+1=1.$$

Последняя операция обосновывается тем, что $1+1=1$.

$$\begin{aligned} 6) \text{ Поглощение:} \\ & A(A+B)=A+AB=A. \end{aligned} \quad (\text{П3.5})$$

Свойство поглощения становится понятным в следующей цепочке преобразований:

$$A(A+B)=AA+AB=A+AB=A(I+B)=A.$$

При описании операций сложения и умножения логических переменных иногда вместо знака плюс употребляют символ $\vee$, а в качестве знака умножения – символ $\wedge$.

ПЗ.3. Операция инверсии и законы Де Моргана

Инверсной (дополнением) логической переменной A является логическая переменная $\bar{A}$, равная 1, когда $A=0$, и нулю, когда $A=1$. Логическая переменная и ее инверсия связаны соотношениями *склеивания*:

$$A+\bar{A}=1 \text{ и } A\bar{A}=0.$$

Кроме того, применение операции инверсии ведет к следующим следствиям:

$$\overline{\bar{A}}=A, \quad \bar{1}=0, \quad \bar{0}=1.$$

Важнейшее значение в технических приложениях булевой алгебры имеют законы двойственности, или законы Де Моргана (А. Де Морган, 1806-71):

- 1) инверсия суммы равна произведению инверсий слагаемых, входящих в состав данной суммы:

$$\overline{A+B}=\bar{A}\bar{B};$$

- 2) инверсия произведения равна сумме инверсий сомножителей, входящих в состав данного произведения:

$$\overline{AB}=\bar{A}+\bar{B}.$$

Законы Де Моргана верны для любого количества слагаемых в составе исходной суммы и для любого количества сомножителей в составе исходного произведения.

Пример ПЗ.1.

$\overline{X_1 X_2 + \overline{X_1} \overline{X_2}} = \overline{X_1 X_2} \square \overline{\overline{X_1} \overline{X_2}} = (\overline{X_1} + \overline{X_2})(X_1 + X_2) = \overline{X_1} X_1 + \overline{X_1} X_2 + X_1 \overline{X_2} + \overline{X_2} X_2 = \overline{X_1} X_2 + X_1 \overline{X_2}$
, поскольку $X_1 \overline{X_1}$ и $\overline{X_2} X_2$ тождественно равны нулю.

Примем также во внимание часто встречающиеся тождественные преобразования:

$$A + \overline{A} B = A + AB + \overline{A} B = A + B(A + \overline{A}) \equiv A + B ;$$

а также

$$AB + AC + B \overline{C} = AB(C + \overline{C}) + AC + B \overline{C} = ABC + AC + AB \overline{C} + B \overline{C} = AC(B + 1) + B \overline{C} (A + 1) \equiv AC + B \overline{C} .$$

ПЗ.4. Булевы функции

Функцией логических переменных $X_1, X_2, \dots, X_n$ (булевой функцией) называется выражение

$$Y = f(X_1, X_2, \dots, X_n),$$

полученное путем инверсии, сложения и умножения исходных логических переменных.

Для каждого $n \geq 0$ может быть получено ровно $2^{(2^n)}$ различных булевых функций. Так, функцией одной переменной ($n=1$) являются всего четыре:

$$Y_0=0, \quad Y_1=X, \quad Y_2=\overline{X}, \quad Y_3=1,$$

а в табл.ПЗ.1 представлены все функции двух переменных. Из них наиболее важными для технических применений являются функции отрицания произведения (И-НЕ), отрицания суммы (ИЛИ-НЕ)*, логического произведения (И), логического сложения (ИЛИ), инверсии (НЕ) и повторения. Перечисленные функции реализуются в серийно выпускаемых логических микросхемах малой степени интеграции.

*С точки зрения общепринятой в математике терминологии следовало бы применять обозначение НЕ-ИЛИ вместо ИЛИ-НЕ и НЕ-И вместо И-НЕ.

Таблица П3.1

X_0 X_1	1 0 1 0 1 1 0 0	Наименование функций	Алгебраические выражения функций
Y_0	0 0 0 0	Константа нуль	$Y_0=0$
Y_1	0 0 0 1	Отрицание суммы (функция Пирса ИЛИ-НЕ)	$Y_1 = \overline{X_1 + X_0}$
Y_2	0 0 1 0	Запрет по X_1	$Y_2 = \overline{X_1} X_0$
Y_3	0 0 1 1	Инверсия X_1 (функция НЕ)	$Y_3 = \overline{X_1}$
Y_4	0 1 0 0	Запрет по X_0	$Y_4 = X_1 \overline{X_0}$
Y_5	0 1 0 1	Инверсия X_0 (функция НЕ)	$Y_5 = \overline{X_0}$
Y_6	0 1 1 0	Неэквивалентность	$Y_6 = X_1 \overline{X_0} + \overline{X_1} X_0$
Y_7	0 1 1 1	Отрицание произведения (функция Шеффера И-НЕ)	$Y_7 = \overline{X_1 X_0}$
Y_8	1 0 0 0	Логическое произведение (конъюнкция, функция И)	$Y_8 = X_1 X_0$
Y_9	1 0 0 1	Эквивалентность	$Y_9 = X_1 X_0 + \overline{X_1} \overline{X_0}$
Y_{10}	1 0 1 0	Повторение X_0	$Y_{10} = X_0$
Y_{11}	1 0 1 1	Импликация X_1 и X_0	$Y_{11} = \overline{X_1} + X_0$
Y_{12}	1 1 0 0	Повторение X_1	$Y_{12} = X_1$
Y_{13}	1 1 0 1	Импликация X_0 и X_1	$Y_{13} = X_1 + \overline{X_0}$
Y_{14}	1 1 1 0	Логическое сложение (дизъюнкция, ИЛИ)	$Y_{14} = X_1 + X_0$
Y_{15}	1 1 1 1	Константа единица	$Y_{15} = 1$

Булевы функции заданы в табл. П.3.1 двумя способами: с помощью таблиц истинности и с помощью алгебраических выражений. В

таблицах истинности (называемых также таблицами задания) каждому набору аргумента X_0 и X_1 задано значение функции Y_i . В общем случае таблицы истинности полностью определяют значение любой булевой функции, поскольку в них указываются значения функции для всех 2^n возможных наборов аргументов (n – количество аргументов заданной функции). Однако при $n \geq 5$ таблицы истинности становятся громоздкими и употребляются редко. Алгебраические выражения позволяют получить компактную запись булевых функций с помощью операций инверсии, сложения и умножения. Переход от таблицы истинности к алгебраическому выражению может быть осуществлен путем формирования алгебраического выражения в виде суммы произведений аргументов или их инверсий, называемого дизъюнктивной нормальной формой (ДНФ) булевой функции. Для получения ДНФ необходимо для каждого единичного значения функции, заданного в таблице истинности, сформировать произведение аргументов и их инверсий по правилу: в формируемое произведение ставится сам аргумент, если его значение в таблице истинности равно единице, но ставится инверсия, если значение аргумента в таблице истинности равно нулю. ДНФ формируется в виде суммы полученных описанным образом произведений.

Пример ПЗ.2. Записать по таблице истинности выражение функции Y_9 из табл. 3.1.

Решение. Единичным значениям функции Y_9 соответствуют два набора аргументов:

- 1) $X_0=1, X_1=1$;
- 2) $X_0=0, X_1=0$.

Следовательно, ДНФ функции Y_9 должна иметь вид

$$Y_9 = X_1 X_0 + \bar{X}_1 \bar{X}_0.$$

Полученные ДНФ следует по возможности упростить, применяя операции склеивания и поглощения. Рекомендуем получить по таблице истинности выражение функций Y_5 и Y_{14} (см. табл. ПЗ.1.) и упростить их до выражений, приведенных в таблице ПЗ.1.

Не следует забывать, что любой аргумент булевой функции сам может быть сколь угодно сложной булевой функцией.

Приложение 4

Символы и функции стандартного кода ISO–7 для ЧПУ (ГОСТ 20999–83)

- A – угол поворота вокруг оси X ($A \rightarrow \alpha$);
- B – угол поворота вокруг оси Y ($B \rightarrow \beta$);
- C – угол поворота вокруг оси Z ($C \rightarrow \Gamma \rightarrow \gamma$);
- D – величина коррекции (Displacement);
- E – номер кадра перехода;
- F – величина подачи (Feed);
- G – подготовительная функция (Geometry);
- H – число повторений участка программы, заголовков (Heading);
- I, J, K – параметры интерполяции или шаг резьбы параллельно осям X, Y, Z соответственно;
- L – обращение к подпрограмме (Linkage);
- M – вспомогательная функция, функция задания управления электроавтоматикой (Mode);
- N – номер кадра (Number);
- P, Q – третьи функции перемещений, параллельных осям X, Y соответственно;
- R – формальный параметр, регистр (Register);
- S – скорость вращения шпинделя главного движения (Spindle);
- T – функция инструмента (Tool);
- U, V, W – вторые функции перемещений, параллельных осям X, Y, Z соответственно;
- X, Y, Z – первичные перемещения ((или заданные положения) относительно осей X, Y, Z соответственно).

Если символы A, B, C, D, E, P, Q, R, U, V, W не используются в значениях, указанных выше, они становятся неопределенными и могут быть использованы для специальных значений.

Управляющие символы и специальные знаки кода ISO–7

LF (ПС) – конец кадра, разделяет на перфоленте кадры программы обработки (Line Feed);

% – начало программы, обозначает начало программы;

(– круглая скобка, обозначает, что следующая за ней информация не предназначена для считывания и отработки УЧПУ;

) – круглая скобка правая, обозначает, что следующая за ней информация должна считываться и обрабатываться УЧПУ;

+ – плюс, алгебраический знак;

- – минус, алгебраический знак;

/ – пропуск кадра, обозначает, что кадр программы управления может обрабатываться или нет, в зависимости от положения органа управления на пульте управления;

: – главный кадр, обозначает главный кадр программы УЧПУ;

HT (ГТ) – горизонтальная табуляция, управляет перемещением позиции печати к следующей предварительно установленной позиции вдоль строки (Horizontal Tab);

CR (BK) – возврат каретки (Carriage Return);

@ – операция над параметрами или переход к технологической программе, означает, в зависимости от значения следующих за данным знаком двух цифр, либо изменение последовательности выполнения кадров технологической программы, либо выполнение двухадресной операции над параметрами.

Таблица П4.1. Подготовительные функции G (Geometry)

Номер группы	Код	Комментарии
I	G00	Быстрое позиционирование, т. е. перемещение на быстром ходу в заданную точку с торможением до станочной константы. Предварительно запрограммированная скорость перемещения игнорируется, но не отменяется
	G01'	Линейная интерполяция – перемещение с запрограммированной скоростью по прямой
	G02	Круговая интерполяция – перемещение с запрограммированной скоростью по дуге окружности по часовой стрелке, если смотреть в сторону положительного направления оси плоскости траектории движения
	G03	То же, но против часовой стрелки
	G33	Нарезание резьбы с постоянным шагом
	G60	Точное позиционирование в пределах одной или двух определенных зон допусков
II	G28	Автоматический выход в исходную точку через промежуточную точку
	G29	Автоматический выход из исходной точки через промежуточную точку

III	G04	Пауза – выдержка в обработке на время, заданное в кадре
IV	G40	Отмена всех коррекций
	G41	Коррекция на радиус инструмента левая. Используется, когда инструмент находится слева от обрабатываемой поверхности, если смотреть от режущего инструмента в направлении его движения относительно изделия
	G42	Коррекция на радиус инструмента правая
V'	G45	Обеспечивает автоматическое включение сопрягающей дуги между данным и предыдущим кадрами
VI'	G53	Отмена линейного сдвига. Используется при работе в станочной системе координат
VII	G54'	Линейный сдвиг – смещение начала координат на величины, заданные в массиве смещения 1. Используется при работе в абсолютных величинах, при работе в относительных величинах игнорируется
	G55	То же применительно к массиву 2
VIII	G80'	Отмена постоянного цикла
	G81	Постоянный цикл 1
	G82	Постоянный цикл 2
	G83	Постоянный цикл 3
	G84	Постоянный цикл 4
	G85	Постоянный цикл 5
IX	G89	Постоянный цикл 9
	G90"	Задание перемещений в абсолютных величинах, т.е. координаты всех точек заданной траектории отсчитываются от нулевой точки одной и той же заданной системы координат
	G91	Задание перемещений в приращениях, т.е. отсчет перемещений производится относительно предыдущей запрограммированной точки, которая считается началом координат
X'	G92	Установка новой рабочей системы координат
XI'	G09	Торможение в конце кадра; автоматическое уменьшение скорости до станочной константы торможения
XII	G94'	Подача, мм/мин
	G95	Подача, мм/об
XIII	G17	Плоскость обработки XY
	G18	Плоскость обработки ZX
	G19	Плоскость обработки YZ

XIV	G96	Постоянная скорость резания: число, следующее за адресом в слове “скорость шпинделя”, равно скорости резания в м/мин; скорость шпинделя регулируется автоматически
	G97"	Отмена функции G96, восстановление задания скорости в об/мин
XV	G40"	Отмена всех коррекций
	G43	Коррекция инструмента положительная
	G44	Коррекция инструмента отрицательная
XVI	G20	Масштабирование (только для 2C85)
	G21	Отмена масштаба
XVII	G59	Программируемый дополнительный сдвиг нуля станка

' Функция действует в одном кадре

" Функция устанавливается автоматически при включении УЧПУ.

Таблица П4.2. Вспомогательные функции M (Mode)

Код	Комментарии	Условие
M00	Программируемый останов: производится останов шпинделя, подачи, выключение охлаждения. Дальнейшая работа возобновляется нажатием кнопки “Пуск”	П
M01	Останов с подтверждением – аналогично M00, но выполняется при предварительном подтверждении с пульта оператора	
M02	Конец раздела программы: указывает на завершение отработки текущего раздела УП. Производится останов шпинделя, подачи, выключение охлаждения, автоматический переход на начало программы. Запуск программы производится по нажатию кнопки “Пуск”	
M03	Включение шпинделя на вращение по часовой стрелке	Д
M04	Включение шпинделя на вращение против часовой стрелки	
M05	Останов шпинделя – производится останов шпинделя и выключение охлаждения	П
M06	Смена инструмента	Выполняется в кадре

M07, M08	Включение охлаждения №2 и №1 соответственно	Д
M09	Отключение охлаждения	П
M10, M11	Зажим и разжим в приспособлениях	Д
M13, M14	То же, что и M03, M04, но с включением охлаждения	
M15, M16	Перемещение “+” и “-“, используется для задания соответственно положительного и отрицательного направления перемещения, запрограммированного в кадре	
M17	Выход из подпрограммы	П
M19	Останов шпинделя в заданной позиции	
M20	Конец повторяющегося участка программы	
M30	Конец массива программ с переходом на начало массива (конец ленты)	
M36	Отмена M37	Д
M37	Уменьшение подачи в 100 раз	

Примечание. П – выполнение после отработки кадра, Д – выполнение до отработки кадра.

