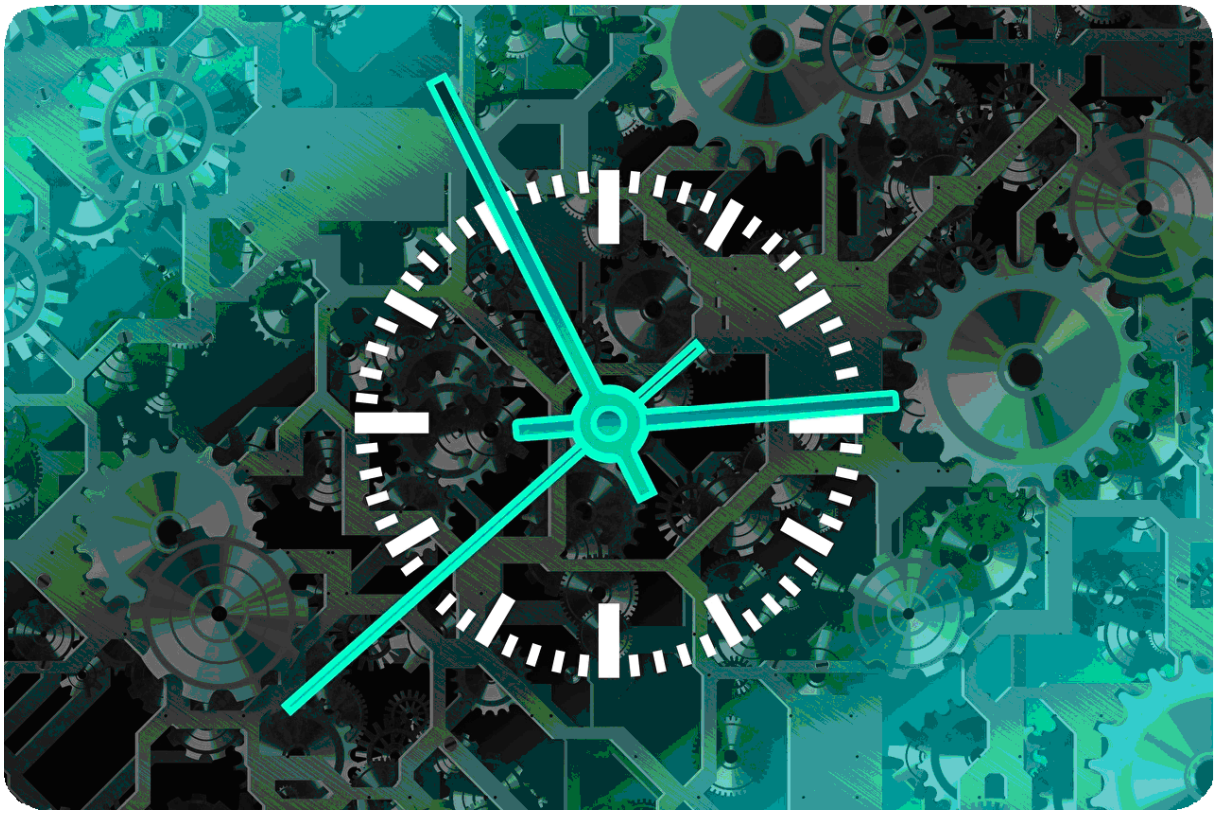


А.Н. ПОЛЕТАЙКИН

СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Учебное пособие

Краснодар
2020

Министерство науки и высшего образования
Российской Федерации
КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

А.Н. ПОЛЕТАЙКИН

СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Учебно-методическое пособие

Краснодар
2020

УДК 629.113(071):004.01:004.4
ББК 32.973 (018.2)
П 49

Рецензенты:
Доктор технических наук, профессор
А.Н. Фионов

Кандидат физико-математических наук, доцент
В.В. Подколзин

Полетайкин, А.Н.

П 49 Системы реального времени: учеб. пособие / А.Н. Полетайкин. – Краснодар: Кубанский гос. ун-т, 2020. − 105 с. – 100 экз.
ISBN 978-5-8209-1757-8

Содержит теоретические положения и методические указания к выполнению лабораторных работ по учебной дисциплине «Системы реального времени».

Адресовано бакалаврам направления подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем». Может быть полезно студентам, изучающим язык программирования ассемблера и низкоуровневое программирование IBM PC.

УДК 629.113(071):004.01:004.4
ББК 32.973 (018.2)

ISBN 978-5-8209-1757-8

ОБЩИЕ ПОЛОЖЕНИЯ

Дисциплина «Системы реального времени» рассматривает структурные и функциональные особенности автоматизированных систем, работающих в режиме реального времени. Также изучаются: а) разные виды периферийного оборудования систем реального времени (СРВ), в том числе основные виды датчиков и исполнительных устройств в составе встроенных, технологических и интегрированных СРВ, которые выполняют контрольно-управляющие функции в режиме реального времени; б) подходы к созданию аппаратного и программного обеспечения разных измерений и выполнению технологических задач в условиях СРВ с целью выработки практических навыков технической реализации указанных СРВ и создания для них специального программного обеспечения.

В пособии рассматриваются конкретные примеры и классификация СРВ, понятия измерительных преобразователей и их параметры, понятие и устройство датчика и исполняющего устройства, классификация измерительных преобразователей, виды и технические характеристики исполнительных устройств, принципы организации обмена между датчиками и исполняющим устройством, принципы построения периферийных и человеко-машинных интерфейсов для функционирования СРВ, базовая архитектура микропроцессора *i*486 и структура программы на языке ассемблера микропроцессора *i*486, виды адресации и основные группы команд в языке ассемблера микропроцессора *i*486.

ОРГАНИЗАЦИЯ ЗАНЯТИЙ ПО КУРСУ

Дисциплина «Системы реального времени» читается студентам специальности направления подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем» на четвертом курсе обучения после изучения дисциплин «Информатика и программирование»,

«Математический анализ», «Дискретная математика и математическая логика», «Теория вычислительных процессов и структур», «Технология разработки ПО», «Метрология и качество ПО», «Микропроцессорная техника». Целью курса является формирование у студентов знаний, умений и практических навыков в сфере использования построения СРВ с применением датчиков и исполнительных устройств различного назначения и программирования обмена данными между регистрирующими и исполнительными компонентами СРВ и ядром СРВ на примере процессора *i*486. Полученные навыки и умения в дальнейшем используются при изучении дисциплин «Распределенные системы и алгоритмы», «Параллельное программирование», «Прикладное ПО», «Компьютерное моделирование», в материалах НИР и производственных практик, а также при курсовом и дипломном проектировании по направлению подготовки 02.03.03 «Математическое обеспечение и администрирование информационных систем».

Изучаемый курс предусматривает выполнение пяти лабораторных работ, рабочая схема выполнения которых представлена в таблице.

Рабочая схема выполнения лабораторных работ

№ п/п	Тема лабораторной работы: материал для изучения и характер выполняемых работ	Объем, ч
1	Представление целых чисел в памяти ЭВМ: составление простейшей программы на языке ассемблера для пересылки данных с использованием различных способов адресации операндов	2
2	Изучение команд арифметических и логических операций микропроцессора *i486*: команды арифметических и логических операций языка ассемблера; использование команд сдвига для редукции по модулю 2	4
3	Составление алгоритмов для обработки одномерных массивов: прямая и косвенная адресация данных; организация ветвлений и	4

	циклов; обработка одномерных массивов на языке ассемблера	
4	Программирование ввода-вывода в режиме реального времени: организация и программирование ввода-вывода информации в СРВ; программная реализация временной задержки при обработке данных на языке ассемблера	4
5	Инфообмен между ЭВМ и периферийными устройствами: преобразования входного аналогового сигнала в двоичный код; дискретизация сигнала по времени; градуировочная характеристика датчика; разработка структурной схемы интерфейса связи УВМ с объектом; разработка алгоритма функционирования объекта и драйвера для связи УВМ с объектом	4
	Всего за семестр	18

По каждой лабораторной работе оформляется отчет. Отчеты сдаются на проверку руководителю в течение семестра по мере их выполнения и защищаются в установленном порядке. При положительном результате защиты студент получает дифференцированную оценку. При неудовлетворительной оценке знаний студента по теме данного отчета студент возвращается к повторному изучению соответствующих материалов, после чего допускается к повторной защите. Неудовлетворительно выполненный отчет также возвращается на доработку. Требования к содержанию отчетов о выполнении лабораторных работ представлены далее.

ТРЕБОВАНИЯ К СОДЕРЖАНИЮ ОТЧЕТОВ О ВЫПОЛНЕНИИ ЛАБОРАТОРНЫХ РАБОТ

В процессе изучения курса «Системы реального времени» должны быть подготовлены и сданы 5 отчетов о выполнении

лабораторных работ (см. табл.). Лабораторные работы выполняются в соответствии с заданием (разд. 2) в рамках индивидуальной профессионально ориентированной темы.

Отчет должен содержать заголовок, тему, цель, задание, ход работы, результаты (теоретические и практические) и выводы. *Заголовок* обозначает род выполненной работы: Лабораторная работа №#. *Тема, цель* и *задание* копируются из соответствующего раздела методических указаний, включая общую и индивидуальные части. *Ход работы* должен содержать по пунктам информацию о выполнении каждого пункта задания. При необходимости ход работы следует дополнять иллюстрациями, таблицами и листингами программы. *Результаты* представляются после раздела *Ход работы* и сопровождаются краткими комментариями. В разделе *Выводы* по пунктам приводится краткая характеристика результатов (что сделано?), а также краткие сведения о методах и средствах их получения.

Титульный лист к отчету должен содержать сведения об образовательной организации и кафедре, заголовок отчета, название учебной дисциплины, номер лабораторной работы, фамилию, инициалы и должность преподавателя, фамилию, инициалы и номер группы исполнителя, город и год составления отчета.

ЛАБОРАТОРНЫЕ РАБОТЫ

Лабораторная работа 1
Представление целый чисел в памяти ЭВМ

Цель: изучение принципов представления числовой информации в памяти ЭВМ; приобретение практических навыков представления целых чисел со знаком в машинном формате.

Теоретические положения

Представление целых чисел со знаком

Для представления целых чисел со знаком используются три формата (табл. 1.1)

Таблица 1.1

Форматы представления целых чисел в памяти ЭВМ

Формат	Директива	Диапазон	Описание в сегменте данных
Байт	DB (define byte)	–128 ÷ +127	A DB 5, –8, 7, ...
Слово	DW (define word)	–32768 ÷ +32767	B DW 15, 8, –37, ...
Двойное слово	DD (define doubleword)	-2^{31} ÷ $+2^{31}-1$	C DD 25, –6, –27, ...

Целые числа со знаком представляются в дополнительном коде. Дополнительный код положительного числа есть само число в шестнадцатеричной системе счисления нужного формата.

Дополнительный код отрицательного числа формируется так:

1) модуль отрицательного числа определяется в двоичном виде;

2) слева дописываются нули до нужного формата;

3) полученное число инвертируется и к нему прибавляется 1;

4) результат разбивается на тетрады и записывается в шестнадцатеричном виде;

5) для форматов DW и DD в дампе памяти байты идут в обратном порядке.

Пример: число – 8 в формате DB, DW, DD:

Операция	Формат DB	Формат DW
1) модуль в двоичной форме	1000	1000
2) дописываем 0	00001000	0000 0000 0000 1000
3) инвертируем и прибавляем 1	11111000	1111 1111 1111 1000
4) в 16-ричном виде	F8	FF F8
5) в дампе памяти	F8	F8 FF

Формат DD	
1) модуль в двоичной форме	1000
2) дописываем 0	0000 0000 0000 0000 0000 0000 0000 1000
3) инвертируем и прибавляем 1	1111 1111 1111 1111 1111 1111 1111 1000
4) в 16-ричном виде	FF FF FF F8
5) в дампе памяти	F8 FF FF FF

Таким образом, DB в памяти компьютера так и останется как F8, а DW и DD в памяти будут соответственно F8 FF и F8 FF FF.

Базовая архитектура процессора

В программной модели процессора i486 имеется 31 регистр (рис. 1).

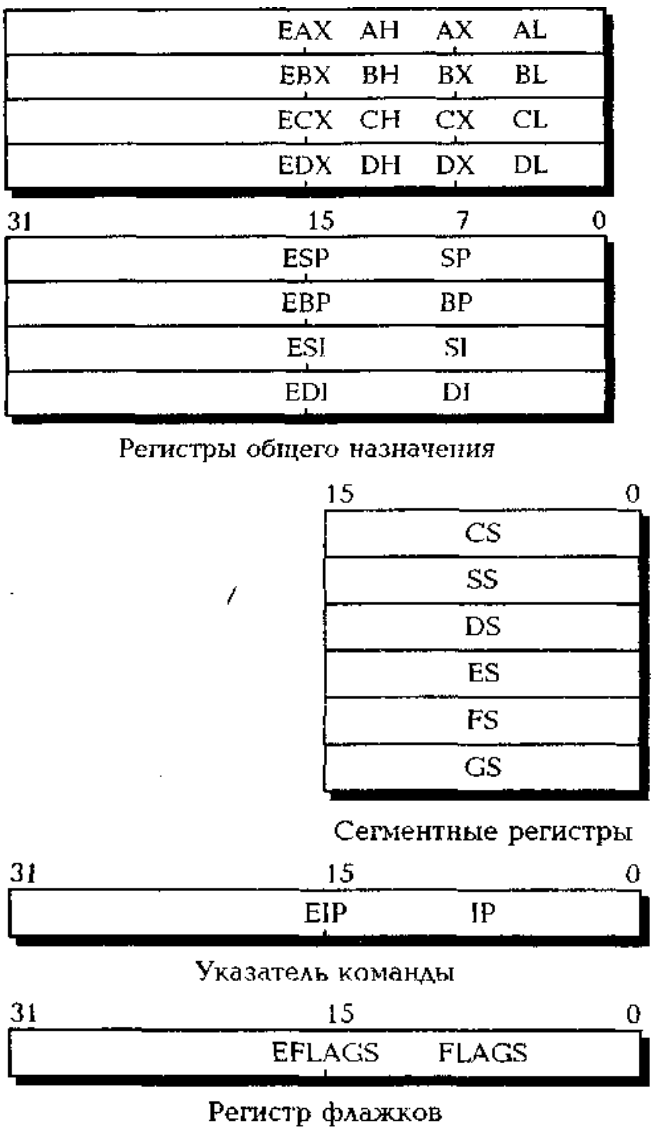


Рис. 1. Регистровая модель процессора i486

Регистры подразделяются на 16 регистров прикладного программиста (пользовательские регистры) и 15 регистров системного программиста (системные регистры). Функционально они подразделяются на несколько групп: регистры общего назначения, индексные и базовые регистры.

Системные регистры предназначены для управления режимом работы процессора. Они определяют системные таблицы, позволяют задавать контрольные точки при отладке программ и проверять функционирование внутренней кэш-памяти, а также кэш-буфера страничного преобразования.

2. Регистры общего назначения (РОН). Восемь 32-битных РОН применяются для хранения данных и/или адресов. Буква E в названии регистров означает «расширенный» (Extended). Отметим, что для всех РОН допускается указывать в командах их младшие 16-битные половины – регистры AX, BX, CX, DX, SP, BP, SI, DI. Кроме того, в первых четырех РОН отдельно адресуются старшие (High) и младшие (Low) байты – регистры AH, BH, CH, DH и AL, BL, CL, DL. Старшие половины РОН, т.е. биты 31–16, отдельно адресовать нельзя. Таким образом, первые четыре РОН позволяют процессору легко оперировать байтами, словами и двойными словами. Рассмотрим некоторые функции 32/16/8-битных РОН.

EAX/AX/AL – аккумулятор (Accumulator). Наиболее часто применяется для хранения промежуточных данных. Многие команды оперируют данными в аккумуляторе значительно быстрее, чем в других регистрах. Задействован в операциях умножения/деления и командах ввода-вывода с привлечением внешних портов. Вся десятичная арифметика выполняется только с участием аккумулятора.

EBX/BX – базовый регистр (Base). Применяется для указания базового (начального) адреса объекта данных в памяти.

ECX/CX/CL – регистр-счетчик (Counter). Участвует в качестве счетчика в некоторых командах, которые производят повторяющиеся операции, например сдвиги, манипуляции цепочками или операторы циклов.

EDX/DX – регистр данных (Data). Наиболее часто привлекается для хранения промежуточных данных, а также в командах умножения и деления (совместно с аккумулятором). В командах ввода-вывода регистр DX содержит адрес порта, к которому производится обращение.

ESP/SP – указатель стека (Stack Pointer). Регистр неявно используется в стековых операциях, в том числе при обработке прерываний, а также при вызове и возврате из подпрограмм. Регистр ESP (SP) адресует вершину стека в текущем сегменте.

EBP/BP – указатель базы (Base Pointer). Предназначен для удобного доступа к объектам данных, находящимся в памяти, например, к параметрам подпрограмм.

ESI/SI – индекс источника (Source Index). Выполняет функцию регистра адреса. При производстве цепочечных операций адресует текущий элемент цепочки-источника.

EDI/DI – индекс получателя или приемника (Destination Index). Выполняет функцию регистра адреса. При производстве цепочечных операций адресует текущий элемент цепочки-получателя.

3. Сегментные регистры. Процессор i486 имеет шесть 16-битных сегментных регистров, которые содержат селекторы сегментов (segment selector) и определяют текущие доступные сегменты памяти. В общем, каждый сегментный регистр определяет сегмент (т.е. блок смежных ячеек) памяти. В процессоре i486 имеет место сегментная организация памяти и содержимое сегментного регистра определяет сегмент косвенно, через дескрипторную таблицу (descriptor table). Сегментные регистры предназначены для идентификации тех текущих сегментов, к которым может обращаться выполняющаяся программа. Производить прямые обращения по физическим адресам памяти программист не может.

CS – сегментный регистр кода (Code Segment). Определяет текущий сегмент, который содержит машинные команды, образующие программу.

SS – сегментный регистр стека (Stack Segment). Задает текущий сегмент стека, т.е. той области памяти, в которой выполняются все стековые операции. Вершину стека адресует регистр ESP (SP).

DS – сегментный регистр данных (Data Segment). Идентифицирует текущий сегмент, содержащий данные текущей программы.

ES, FS, GS – дополнительные сегментные регистры. Предназначены для задания трех сегментов данных, доступных текущей программе, и не выполняют никаких специальных функций.

Записываемое в сегментный регистр слово называют селектором (selector), поскольку оно выбирает (селектирует) один конкретный сегмент из множества возможных. Каждый сегмент

имеет базовый адрес. Для адресации данных внутри сегмента к базовому адресу прибавляется 32-битное смещение (offset). После выбора сегмента (посредством загрузки селектора сегмента в сегментный регистр) программа должна определять только смещения. При этом явная спецификация сегмента необязательна. Если сегмент явно не указан с помощью префикса замены сегмента, процессор автоматически выбирает сегмент в соответствии с правилами, приведенными в табл. 1.2.

Таблица 1.2

Правила выбора сегментов по умолчанию

Тип обращения	Используемый сегмент, используемый регистр	Правило выбора сегмента по умолчанию
Команды	Сегмент кода Регистр CS	Автоматически при выборке команды
Стек	Сегмент стека Регистр SS	Все стековые операции. Все обращения через ESP и EBP
Локальные данные	Сегмент данных Регистр DS	Все обращения к данным, кроме стековых и цепочечных
Цепочка-получатель	Доп. сегмент Регистр ES	Получатель цепочечных команд

4. Регистр флагов. Регистр флагов EFLAGS/FLAGS содержит флаги, которые подразделяются на восемь флагов состояния и шесть флагов управления. Формат этого регистра приведен на рис. 2.

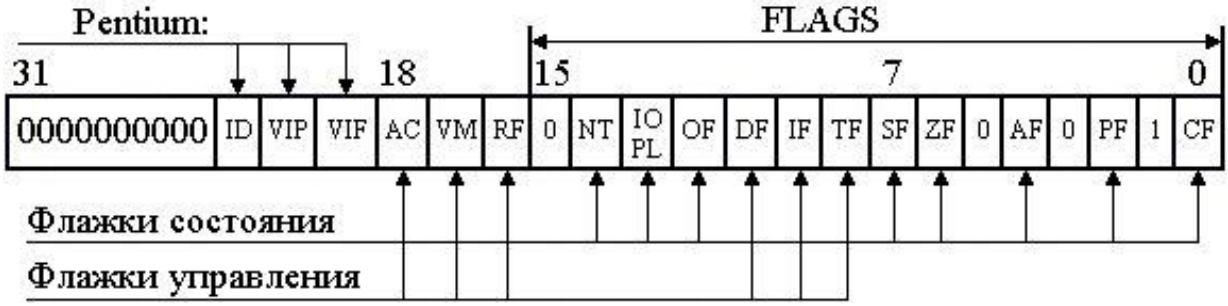


Рис. 2. Структура регистра флагов процессора i486

Флаги состояния в регистре EFLAGS сообщают об особенностях результата команды обработки данных. Команды простой пересылки данных (например, команда MOV) не воздействуют на флаги состояния. Команды условных переходов позволяют программе учесть состояния флагов и отреагировать на них. Например, когда счетчик управления циклом уменьшается до 0, изменяется состояние флага ZF, это изменение можно использовать для подавления перехода в начало цикла. К флагам состояния относятся следующие флаги.

0 CF (Carry Flag) – флаг переноса. Этот бит устанавливается в состояние 1, если арифметическая операция вызвала перенос (при сложении) или заем (при вычитании); в противном случае флаг CF сбрасывается в состояние 0.

2 PF (Parity Flag) – флаг паритета (четности). Флаг PF во многих командах устанавливается в состояние 1, если младшие восемь бит результата содержат четное число единичных бит; в противном случае флаг PF сбрасывается в состояние 0.

4 AF (Auxiliary carry Flag) – флаг вспомогательного переноса. Флаг AF устанавливается в состояние 1, если арифметическая операция вызвала перенос или заём в/из тетрады, т.е. бита 3, результата; в противном случае флаг AF сбрасывается в состояние 0.

6 ZF (Zero Flag) – флаг нуля. Флаг ZF большинством команд устанавливается в состояние 1, когда результат операции равен 0; в противном случае он сбрасывается в состояние 0.

7 SF (Sign Flag) – флаг знака. Большинство команд устанавливают флаг SF в то же состояние, в каком находится старший бит результата. В зависимости от размера операндов им будет бит 7, 15 или 31. При использовании целых знаковых чисел, представленных в дополнительном коде, флаг SF показывает знак результата.

11 OF (Overflow Flag) – флаг переполнения. Флаг OF устанавливается в состояние 1, если в арифметических операциях со знаковыми операндами результат превышает диапазон представимых чисел; в противном случае флаг OF сбрасывается в состояние 0. В зависимости от размера операндов старшим битом является бит 7, 15 или 31.

Первые шесть флагов называются арифметическими флагами, так как они отражают ту или иную особенность результата арифметической или логической команды.

12, 13 IOPL (input/Output Privilege Level) – уровень привилегий ввода-вывода. Это двухбитное поле является составной частью механизма защиты процессора. Оно показывает тот минимальный уровень привилегий выполняющейся задачи, на котором разрешается производство операций ввода-вывода.

14 NT (Nested Task) – флаг вложенной задачи. Флаг NT устанавливается в состояние 1, когда текущая задача производит переключение на другую задачу с помощью команды вызова CALL. С помощью этого флага организуется цепь вложенных задач аналогично вложению подпрограмм.

Флаги управления в регистре EFLAGS воздействуют на определенные аспекты работы процессора.

8 TF (Trap Flag или Trace Flag) – флаг покомандной работы (трассировки). Когда этот флаг находится в состоянии 0, процессор работает обычным образом, а при TF = 1 он переводится в режим покомандной работы, т.е. после выполнения каждой команды генерируется внутреннее прерывание с переходом к отладчику. Этот флаг является одним из средств, предназначенных для отладки программ.

9 IF (Interrupt Flag) – флаг прерывания. Разрешает (1) и запрещает (0) внешние прерывания на входе INTR.

10 DF (Direction Flag) – флаг направления. Этот флаг задает направление обработки цепочек строковыми командами.

Указатель команды. Регистр указателя команды EIP/IP (Instruction Pointer) предназначен для адресации команд внутри текущего сегмента кода. Обязательно участвует во всех командах передачи управления, к которым относятся команды условных и безусловных переходов, вызова подпрограмм, возврата из подпрограмм и др. В операциях с регистром EIP участвуют двойные слова, а в операциях с регистром IP – слова.

Структура программы на языке ассемблера. Программа на языке ассемблера состоит из предложений. Каждое

предложение занимает одну строку, которую условно можно разделить на четыре части:

Поле метки	*Поле команды/директивы*	*Поле операндов*	*Поле комментариев*

Поле метки отделяется от поля команды/директивы или пробелом, или двоеточием. Поле команды/директивы отделяется от поля операндов пробелом. Если поле операндов составляет два операнда, то они отделяются запятой. Поле комментариев начинается с точки с запятой (;). Условность распределения на указанные поля определяет, что какое-то из полей может отсутствовать. Например, есть команды, которые не имеют операндов (CBW). Также комментарии могут занимать всю строку.

Каждая программа состоит не менее чем из трех сегментов: кодовый сегмент, сегмент данных и стековый сегмент. Сегментов данных может быть больше одного. Информация о размещении сегментов в памяти компьютера сохраняется в соответствующих сегментных регистрах: CS, DS (ES), SS. В программе необходимо разместить специальные директивы, которые описывают сегменты программы. Эти директивы начинаются с точки и могут иметь или не иметь операнды.

.MODEL – директива определения используемой модели памяти; имеет операнд, означающий тип памяти, например .MODEL SMALL.

.STACK – директива определения размера стекового сегмента. Размер задается операндом и является числом байтов, отведенных под стек.

.DATA – директива определения сегмента данных. Не имеет операндов. После этой директивы можно размещать директивы описания меток с данными.

.CODE – директива определения кодового сегмента. После этой директивы надо размещать команды программы.

Каждая программа должна начинаться с двух команд, которые определяют сегментный регистр данных (DS):

```
MOV AX, @DATA
MOV DS, AX
```

Сегментные регистры стека и кода определяются операционной системой от времени запуска программы на выполнение.

Программный модуль надо заканчивать директивой END. Если директива не имеет операнда, то выполнение программы начинается с первой команды в кодовом сегменте. Операндом этой директивы может быть метка, которая указывает процессору команду, с которой надо начинать выполнение программы. Такой вариант директивы можно использовать, когда в начале кодового сегмента описаны процедуры.

Для корректного окончания программы ее надо заканчивать так:

```
MOV AH, 4Ch
INT 21h
```

Эти команды выполняют выход из программы в ОС.

Составление программы на языке ассемблера включает четыре этапа.

1. *Написание исходного кода программы*. Для этого используется любой редактор, который разрешает сохранять файл в формате «текст MS DOS». Файл сохраняется с расширением .ASM.

2. *Трансляция программы*. Для этого в командной строке надо набрать команду TASM и полный путь к файлу с исходным кодом программы, при этом надо находиться в папке, в которой находится программа TASM. Результатом трансляции является информация об ошибках и сообщениях в программе (или об их отсутствии). Строка с информацией включает номер строки программы с ошибкой (сообщением) и расшифровка ошибки (сообщение). Если в результате трансляции программы имеют место ошибки или/и сообщения, надо исправить ошибки и снова выполнить этап 2. Результатом трансляции является объектный файл с расширением .OBJ.

3. *Компоновка программы*. Для этого в командной строке надо набрать команду TLINK и полный путь к объектному файлу, при этом надо находиться в папке, в которой находится программа TLINK. Результатом трансляции является выполняемый файл с расширением .EXE.

4. *Выполнение программы*. Для этого в командной строке надо набрать команду TD и полный путь к файлу с расширением .EXE, при этом надо находиться в папке, в которой находится программа TD. TD – это программа-отладчик, с помощью которой можно выполнять программу по шагам.

Пример: написать и откомпилировать программу, определив заданные числа в сегменте данных

```
DOSSEG
.MODEL SMALL
.STACK 100h
.DATA
 A DB 90
 B DW 90, -172
 C DD 90, -172
.CODE
MOV AX, @DATA
MOV DS, AX
;       …программный код…
MOV AH, 4Ch
INT 21h
END
```

Методы адресации в микропроцессоре *i486*

– Непосредственная адресация. Значение операнда указывается в команде.

MOV AX, 10Ah

ADD BX, -532

– Регистровая адресация. Значение операнда сохраняется в одном из регистров общего назначения.

MOV DX, CX

SUB AX, BX

– Прямая адресация. Если известен адрес операнда, располагающегося в памяти, можно использовать этот адрес. Если операнд – слово, находящееся в сегменте, на который указывает ES, со смещением от начала сегмента 0001, то команда

MOV AX, ES:0001

поместит это слово в регистр AX.

В реальных программах обычно для задания статических переменных используют директивы определения данных, которые позволяют ссылаться на статические переменные не по адресу, а по имени. Тогда, если в сегменте, указанном в ES, была описана переменная word_var размером в слово, можно записать ту же команду как

MOV AX, ES:WORD_VAR

В таком случае ассемблер сам заменит слово «word_var» на соответствующий адрес. Если селектор сегмента данных находится в DS, имя сегментного регистра при прямой адресации можно не указывать, DS используется по умолчанию. Прямая адресация иногда называется адресацией по смещению.

– Косвенная адресация

В команде указывается общая формула, определяющая адрес операнда.

Варианты косвенной адресации с примерами представлены в табл. 1.3.

Таблица 1.3

Варианты косвенной адресации процессора i486

<table>
<tr><td colspan="5">БАЗА + ИНДЕКС + СМЕЩЕНИЕ</td></tr>
<tr><td colspan="5">В роли БАЗЫ могут использоваться регистры BX или BP
В роли ИНДЕКСА могут использоваться регистры SI и DI
В роли СМЕЩЕНИЯ можно использовать имя метки, константу или выражение</td></tr>
<tr><td>BX</td><td>SI</td><td>VAR</td><td colspan="2">имя переменной</td></tr>
<tr><td>BP</td><td>DI</td><td>102h</td><td colspan="2">константа</td></tr>
<tr><td></td><td></td><td>VAR+5</td><td colspan="2">выражение</td></tr>
<tr><td colspan="5">Примеры косвенной адресации</td></tr>
<tr><td colspan="2">INC [BX+VAR]</td><td colspan="3" rowspan="3">БАЗА + СМЕЩЕНИЕ</td></tr>
<tr><td colspan="2">INC [BP+CONST]</td></tr>
<tr><td colspan="2">INC [VAR]+[BP]</td></tr>
<tr><td colspan="2">INC [SI+VAR]</td><td colspan="3" rowspan="2">ИНДЕКС + СМЕЩЕНИЕ</td></tr>
<tr><td colspan="2">INC VAR[SI]</td></tr>
<tr><td colspan="2">INC WORD PTR [BX+SI]</td><td colspan="3" rowspan="2">БАЗА + ИНДЕКС</td></tr>
<tr><td colspan="2">INC WORD PTR [BP]+[DI]</td></tr>
</table>

INC [VAR+BX+DI]	БАЗА + ИНДЕКС + СМЕЩЕНИЕ
INC VAR[BP][SI]	
MOV AX,VAR[BP+DI]	
MOV AX,[BX]	БАЗА
MOV AX,[SI]	ИНДЕКС
MOV AX,[102h]	СМЕЩЕНИЕ (относится к прямой адресации)

Структура программы для процессора i686

При использовании инструментальной среды RadASM система команд и методы адресации будут те же. Однако структура программы будет иметь отличия. В данном случае программа начинается с директив, включающих архитектуру процессора i686 и необходимые библиотеки (например, ввода-вывода). Завершение программы осуществляется командой exit.

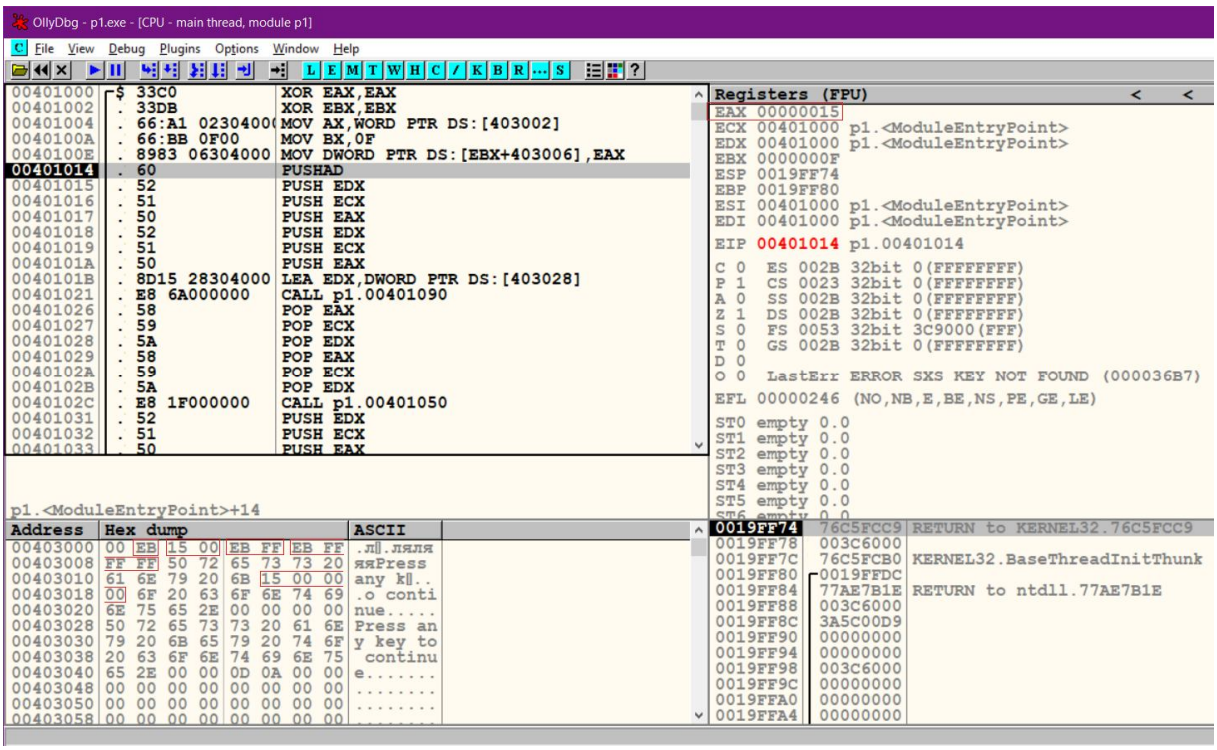
Пример кода, реализующего типовое задание на лабораторную работу №1:

```
.686     ; Архитектура процессора i686
include /masm32/include/io.asm  ; Библиотека ввода-вывода

.data                           ; Сегмент данных
    AA DB -21
    A DW 21
    B DW -21
    CD DD -21

.code ; Сегмент кода
start:
   ;обнуление регистров
   xor eax,eax
   xor ebx,ebx
   mov AX, A    ; загрузка числа в аккумулятор
   mov BX, 0fh ; загрузка элемента формулы косвенной адресации
   mov [ebx]+offset CD, EAX ; пересылка данных посредством
                            ; косвенной адресации
   inkey ; Ждем нажатия любой клавиши
   exit  ; Завершаем работу программы
end start ; Объявлвем точку входа в программу
```

После сборки (Build) и пошагового выполнения в отладчике данный код и состояние дампа памяти выглядит так:



На рисунке показана программа в режиме отладки на этапе завершения выполнения программы, когда все операции выполнены. В нижней левой части отладчика отображается дамп памяти, где рамками выделены значения объявленных переменных, а также значение содержимого аккумулятора, переданное в память со смещением на F байт относительно метки переменной CD.

Дополнительная информация

– В двухоперандных командах операнды должны быть одной длины.

– Длина операндов, задействованных в команде, может определяться длиной одного из них. Если надо изменить размерность операнда, используются префиксы:

word ptr (длина операнда равняется слову);

byte ptr (длина операнда равняется байту).

Например: mov word ptr A, ax.

– В роли непосредственного операнда можно использовать адрес метки, которая определяется при помощи префиксов:

offset (смещение метки в сегменте);

seg (адрес сегмента, в котором размещенная метка).

Например: mov ax, offset A.

Адрес метки можно сохранить как значение другой переменной, указав имя метки в разделе данных. При этом необходимо помнить, что длина смещения метки равняется слову.

Например:

```
.DATA
A DB 90
B DW 90, -172, C   ; значение байтов B + 4
                   ; и B + 5
; равняется значению смещения метки C
```

Если при тестировании программы будет определена ошибка о неизвестности формата данных, то необходимо перед операндом с непосредственной адресацией поставить соответствующий префикс длины операнда: word ptr или byte ptr.

Более подробную информацию по созданию программ при помощи инструментальной среды RadASM см. в учебном пособии [2].

Задание

1. Представить целые числа I_1 и I_2 (табл. 1.4) в формате DB, DW, DD.
2. Составить и откомпилировать программу, определив число I_1 в форматах DB, DW, DD, а число I_2 в форматах DW, DD.
3. Задать такие операции пересылки данных:

– загрузить регистр R1 числом I2 из сегмента данных;

– с использованием заданного варианта косвенной адресации записать содержимое R1 в сегмент данных со смещением на I1 байт относительно метки I2 (число I1 предварительно загрузить в соответствующий базовый или индексный регистр, **при наличии других операндов в формуле адресации определить их произвольно**).

4. Проверить результаты расчетов и пересылок в дампе памяти.
5. Сделать расчет времени выполнения программы (методика расчета представлена в прил. А).

Таблица 1.4

Варианты индивидуальных заданий

№ вар.	I1	I2	R1	Косвенная адресация
1	149	–74	CX	[BX]
2	203	–502	DX	[BP]
3	246	–1917	AX	[BX+CONST]
4	224	–47	BX	[BP+CONST]
5	140	–6205	CX	[SI]
6	129	–132	DX	[DI]
7	151	–7814	AX	[BX+SI]
8	124	–5665	BX	[BP+DI]
9	45	–98	CX	[BX+SI]
10	75	–457	DX	[BP+DI]
11	21	–124	AX	[BX][SI]
12	205	–1055	BX	[BP][DI]
13	80	–395	CX	[BX+CONST]
14	177	–207	DX	[BP+CONST]
15	169	–1134	AX	[SI]
16	122	–167	BX	[DI]
17	227	–1705	CX	[BX+SI]
18	94	–29	DX	[BX+DI]
19	241	–1778	AX	[BP+SI]
20	107	–200	BX	[BP+DI]
21	200	–61	CX	[BX][SI]
22	153	–557	DX	[BX][DI]
23	140	–1909	AX	[BP][SI]
24	70	–2504	BX	[BP][DI]
25	87	–908	CX	[BX+CONST][SI]
26	23	–1323	DX	[BP+CONST][DI]
27	59	–1530	AX	[BX+CONST][DI]
28	190	–440	BX	[BP+CONST][SI]
29	202	–130	BX	[BP]
30	190	–1072	AX	[BX]
31	103	–552	DX	[BP]
32	146	–2917	AX	[BX+CONST]
33	124	–247	BX	[BP+CONST]
34	240	–3205	CX	[SI]
35	229	–1032	DX	[DI]

Лабораторная работа 2
Изучение команд арифметических и логических операций

Цель: изучение команд арифметических и логических операций, приобретение практических навыков при их использовании в различных вычислительных задачах.

Теоретические положения

Команды арифметических операций и их особенности

ADD – выполняет сложение двух операндов, заданных в команде. Микропроцессор помещает результат на место первого. Второй операнд не изменяется. Команда корректирует регистр флагов в соответствии с результатом сложения (был ли результат нулевым, отрицательным, имел ли четность, перенос или переполнение).

Формат команды: ADD Операнд1, Операнд2

Пример: ADD AX, BX.

Операнд1 может размещаться в регистре общего назначения или памяти; Операнд2 – в регистре общего назначения, памяти или представляться константой. Оба операнда не могут размещаться в памяти.

ADC – команда сложения с учетом переноса, т.е. к сумме двух операндов прибавляется флаг переноса. Для любой *формы команды* **ADD** существует сравнимая с ней команда **ADC**.

SUB – аналогична по формату команде сложения **ADD**. Выполняет вычитание из операнда приемника операнда источника, результат операции сохраняется на месте операнда приемника, устанавливает флаги состояния в соответствии с результатом операции, причем флаг переноса означает заем.

Формат команды: SUB Операнд-приемник, Операнд-источник.

Пример: SUB AX, BX.

SBB – команда вычитания с заемом решает задачи вычитания повышенной точности. Учитывает флаг заема при

вычитании, т.е. значение заема вычитается из результата, полученного при нормальном вычитании.

NEG – команда смены знака операнда.

Формат команды: NEG Операнд.

Пример: NEG AX.

INC – команда увеличения. Прибавляет 1 к операнду.

Формат команды: INC Операнд.

Пример: INC AX.

DEC – команда уменьшения. Отнимает 1 от операнда.

MUL, IMUL – команды умножения. По команде **MUL** умножаются два целых числа без знака, при этом получается число без знака. По команде **IMUL** умножаются целые числа со знаком. При умножении целых чисел в качестве операндов используются числа, представленные в дополнительном коде.

По длине операндов, участвующих в операции, команду делят на две группы:

Умножение байт. Значение формата байт, размещенное в регистре AL, умножается на операнд формата байт, указанный в команде. Результат операции формата слово сохраняется в регистре AX.

Умножение слов. Значение формата слово, размещенное в регистре AX, умножается на операнд формата слово, указанный в команде. Результат операции формата двойное слово сохраняется в паре регистров AX, DX. При этом младшие 16 бит результата – в регистре AX, а старшие – в регистре DX.

Формат команды: MUL Операнд.

Операнд может размещаться в регистре или памяти.

Тип команды определяется по длине операнда, указанного в команде.

Пример: MUL BX (поскольку в качестве операнда используется регистр BX, следовательно, умножаются слова).

DIV, IDIV – команды деления. Команда деления **DIV** выполняет деление без знака и дает как частное, так и остаток. Деление целых чисел со знаком **IDIV** отличается от команды **DIV** только тем, что оно учитывает знаки обоих операндов. Если результат положителен, все происходит так же, как и у **DIV**, за исключением того, что максимальное значение частного

соответственно равно 127 и 32767 для байтов и слов. Если результат отрицателен, то остаток имеет тот же знак, что и частное. Минимальные значения частных для отрицательных результатов –128 и –32768 для байтов и слов соответственно.

Формат команды: DIV Операнд.

Операнд является делителем и может размещаться в регистре или памяти. По его длине команду делят на две группы.

Деление слова на байт. Значение формата слово, размещенное в регистре AX, делится на операнд формата байт, указанный в команде.

Результат операции: частное формата байт сохраняется в регистре AL, остаток от деления формата байт хранится в регистре AH.

Деление двойного слова на слово. Значение формата двойное слово, размещенное в паре регистров DX, AX (младшие 16 бит – в регистре AX, а старшие 16 бит – в регистре DX), делится на операнд формата слово, указанный в команде.

Результат операции: частное формата слово сохраняется в регистре AX, остаток от деления формата слово хранится в регистре DX.

Пример: DIV BL (поскольку в команде указан операнд формата байт, то выполняется деление содержимого регистра AX на содержимое регистра BL).

CBW – команда расширения байта до слова. Значение, хранимое в байте AL, преобразуется в слово и сохраняется в регистре AX.

CWD – команда расширения слова до двойного слова. Значение, хранимое в байте AX, преобразуется в двойное слово и сохраняется в паре регистров DX, AX.

Команды логических операций и их особенности: AND, OR, XOR и TEST. Используются для сброса и установки бит и для арифметических операций в коде ASCII. Все эти команды обрабатывают один байт или одно слово в регистре или в памяти и устанавливают флаги CF, OF, PF, SF, ZF.

AND: если оба из сравниваемых битов равны 1, то результат равен 1; во всех остальных случаях результат равен 0.

OR: если хотя бы один из сравниваемых битов равен 1, то результат равен 1; если сравниваемые биты равны 0, то результат равен 0.

XOR: Если один из сравниваемых битов равен 0, а другой равен 1, то результат равен 1; если сравниваемые биты одинаковы (оба – 0 или оба – 1), то результат равен 0.

TEST: действует как AND – устанавливает флаги, но не изменяет биты.

Первый операнд в логических командах указывает на один байт или слово в регистре или в памяти и является единственным значением, которое может изменяться после выполнения команд.

Для следующих несвязанных примеров предположим, что AL содержит 11000101, а BH содержит 01011100.

1. AND AL,BH ;Устанавливает в AL 0100 0100.
2. OR BH,AL ;Устанавливает в BH 1101 1101.
3. XOR AL,AL ;Устанавливает в AL 0000 0000.
4. AND AL,00 ;Устанавливает в AL 0000 0000.
5. AND AL,0FH ;Устанавливает в AL 0000 0101.
6. OR CL,CL ;Устанавливает флаги SF и ZF.

Примеры 3 и 4 демонстрируют способ очистки регистра. В примере 5 обнуляются левые четыре бита регистра AL. Хотя команды сравнения CMP могут быть понятнее, можно применить команду OR для следующих целей:

```
OR CX,CX ; Проверка CX на нуль.
JZ ... ; Переход, если нуль.
OR CX,CX ; Проверка знака в CX.
JS ... ; Переход, если отрицательно.
```

Команда TEST действует аналогично команде AND, но устанавливает только флаги, а операнд не изменяется. Приведем несколько примеров.

```
TEST BL,11110000B ; Любой из левых бит в BL.
JNZ ... ; равен единице?
TEST AL,00000001B ; Регистр AL содержит
JNZ ... ; нечетное значение?
TEST DX,OFFH ; Регистр DX содержит
JZ ... ; нулевое значение?
```

Еще одна логическая команда NOT устанавливает обратное значение бит в байте или в слове, в регистре или в памяти: нули становятся единицами, а единицы – нулями. Если, например, регистр AL содержит 1100 0101, то команда NOT AL изменяет это значение на 0011 1010. Флаги не меняются. Команда NOT не эквивалентна команде NEG, которая меняет значение с положительного на отрицательное и наоборот, посредством замены бит на противоположное значение и прибавления единицы.

Команды сдвига и циклического сдвига, которые представляют собой часть логических возможностей компьютера, имеют следующие свойства:

– обрабатывают байт или слово;
– имеют доступ к регистру или к памяти;
– сдвигают влево или вправо;
– сдвигают на величину до 8 бит (для байта) и 16 бит (для слова);
– сдвигают логически (без знака) или арифметически (со знаком).

Значение сдвига на 1 может быть закодировано как непосредственный операнд, значение больше 1 должно находиться в регистре CL.

Команды сдвига

При выполнении команд сдвига флаг CF всегда содержит значение последнего выдвинутого бита. Существуют следующие команды сдвига:

SHR ; Логический (беззнаковый) сдвиг вправо.
SHL ; Логический (беззнаковый) сдвиг влево.
SAR ; Арифметический сдвиг вправо.
SAL ; Арифметический сдвиг влево.

Следующий фрагмент иллюстрирует выполнение команды SHR:

```
MOV CL, 03
MOV AX, 10110111b
SHR AX, 1 ; 01011011 ; Сдвиг вправо на 1
SHR AX, CL ; 00001011 ; Сдвиг вправо на 3
```

Первая команда SHR сдвигает содержимое регистра AX вправо на 1 бит. Выдвинутый в результате один бит попадает в флаг CF, а самый левый бит регистра AX заполняется нулем. Вторая команда сдвигает содержимое регистра AX еще на три бита. При этом флаг CF последовательно принимает значения 1, 1, 0, а в три левых бита в регистре AX заносятся нули.

Рассмотрим действие команд арифметического сдвига вправо SAR:

```
MOV CL,03 ; AX
MOV AX,10110111B ; 10110111
SAR AX,1 ; 11011011 ; Сдвиг вправо на 1
SAR AX,CL ; 11111011 ; Сдвиг вправо на 3
```

Команда SAR имеет важное отличие от команды SHR: для заполнения левого бита используется знаковый бит. Таким образом, положительные и отрицательные величины сохраняют свой знак. В приведенном примере знаковый бит содержит единицу.

При сдвигах влево правые биты заполняются нулями. Таким образом, результат команд сдвига SHL и SAL индентичен.

Сдвиг влево часто используется для удваивания чисел, а сдвиг вправо – для деления на 2. Эти операции осуществляются значительно быстрее, чем команды умножения или деления. Деление пополам нечетных чисел (например, 5 или 7) образует меньшие значения (2 или 3 соответственно) и устанавливает флаг CF в 1. Кроме того, если необходимо выполнить сдвиг на 2 бита, то использование двух команд сдвига более эффективно, чем использование одной команды с загрузкой регистра CL значением 2.

Для проверки бита, занесенного в флаг CF, используется **команда JC (переход, если есть перенос).**

Команды циклического сдвига. Циклический сдвиг представляет собой операцию сдвига, при которой выдвинутый бит занимает освободившийся разряд. Существуют следующие команды циклического сдвига:

```
ROR ; Циклический сдвиг вправо
ROL ; Циклический сдвиг влево
RCR ; Циклический сдвиг вправо с переносом
RCL ; Циклический сдвиг влево с переносом
```

Следующая последовательность команд иллюстрирует операцию циклического сдвига ROR:

```
MOV CL, 03
MOV BX, 10110111B
ROR BX, 1 ; 11011011 ; Сдвиг вправо на 1
ROR BX, CL ; 01111011 ; Сдвиг вправо на 3
```

Первая команда ROR при выполнении циклического сдвига переносит правый единичный бит регистра BX в освободившуюся левую позицию. Вторая команда ROR переносит таким образом три правых бита.

В командах RCR и RCL в сдвиге участвует флаг CF. Выдвигаемый из регистра бит заносится в флаг CF, а значение CF при этом поступает в освободившуюся позицию.

Рассмотрим пример, в котором используются команды циклического и простого сдвига. Предположим, что 32-битовое значение находится в регистрах DX:AX так, что левые 16 бит находятся в регистре DX, а правые – в AX. Для умножения на 2 этого значения возможны следующие две команды:

```
SHL AX,1 ; Умножение пары регистров
RCL DX,1 ; DX:AX на 2
```

Здесь команда SHL сдвигает все биты регистра AX влево, причем самый левый бит попадает в флаг CF. Затем команда RCL сдвигает все биты регистра DX влево и в освободившийся правый бит заносит значение из флага CF.

Задание

1. Составить программу для расчета заданного арифметического выражения (табл. 2.1). Длину и значение переменных A, B, C выбрать самостоятельно. Константы, заданные в выражении, использовать в кодовом сегменте. Программа должна корректно работать при любых допустимых значениях переменных.
2. Описать команды умножения и деления, используемые в программе на предмет длины операндов, участвующих в операции. Охарактеризовать длину результата и место его хранения.

3. Получить загрузочный модуль и протестировать выполнение программы в отладчике.

4. На основе составленной программы выполнить следующие действия:

– загрузить в аккумулятор маскирующее слово, позволяющее определить заданную характеристику содержимого регистра DX;

– выполнить заданную проверку и ее результат сохранить в переменной RESULT, объявленной в сегменте данных.

5. Перекомпилировать загрузочный модуль и протестировать выполнение программы в отладчике.

6. Произвести расчет времени выполнения программы.

7. Сделать выводы.

Таблица 2.1

Варианты индивидуальных заданий

Номер варианта	Заданные операции	
	Арифметическая	Логическая
1	A/(976+B)–80·(C–15)	Определение наличия 0 в разряде 0
2	A+992/B·17–C/69	Определение наличия 1 в разряде 1
3	A·843+(B–33·C)/58	Определение наличия 0 в разряде 2
4	(A+22)·B–18/C+74	Определение наличия 1 в разряде 3
5	A+(981–B)/78/C+55	Определение наличия 0 в разряде 4
6	(A–255)/(B–164)–C·14	Определение наличия 1 в разряде 5
7	A·64/(B+81)+C/42	Определение наличия 0 в разряде 6
8	(A–481/B)/85·(C–44)	Определение наличия 1 в разряде 7
9	A·91+B/(54+C)·7	Определение наличия 0 в разряде 8

10	A/255–B/(7+C)·78	Определение наличия 1 в разряде 9
11	10A/16+B/(4·C–25)–144	Определение наличия 0 в разряде 10
12	A·(83·B–156)/(C+82)	Определение наличия 1 в разряде 11
13	(A+(98–B·122))/C·55	Определение наличия 0 в разряде 12
14	A/(34·B+1000/(C+21))	Определение наличия 1 в разряде 13
15	A/31·(B·12–C)+39	Определение наличия 0 в разряде 14
16	A–622/B·(87·C+69)	Определение наличия 1 в разряде 15
17	A·(302·B+137)/C	Определение наличия 0 в разряде 0
18	A–300+(B+74·C)/54	Определение наличия 1 в разряде 1
19	A–(867/B·(104–C·28))	Определение наличия 0 в разряде 2
20	(A–(60·B–175)/C)/7	Определение наличия 1 в разряде 3
21	(A–(607+B·92)/C)·50	Определение наличия 0 в разряде 4
22	(A+426)/(B·194+C)·11	Определение наличия 1 в разряде 5
23	A·44–(B+88·C)/34	Определение наличия 0 в разряде 6
24	(A+891/(B·59+C))·76	Определение наличия 1 в разряде 7
25	(A+75)/(B·25–C)·70	Определение наличия 0 в разряде 8
26	A·10+(B+C/41)·D–100	Определение наличия 1 в разряде 9
27	(2·A+145)/5+C/(35·B–11)	Определение наличия 0 в разряде 10

28	A·(150–18·B)/(308+C)	Определение наличия 1 в разряде 11
29	(A+216)/(12·B–4·C)+133	Определение наличия 0 в разряде 12
30	A/(529–B/195+C)10	Определение наличия 1 в разряде 13
31	(A+144)·(B·C–558)/2	Определение наличия 0 в разряде 14
32	(A–322)/B+(8·C–160)	Определение наличия 1 в разряде 15
33	A·(B+17·C)/(C–15)	Определение наличия 0 в разряде 0
34	A+100–(74·B·C)/5	Определение наличия 1 в разряде 1
35	A+(B·(104–C·19)/20)	Определение наличия 0 в разряде 2

Пример выполнения задания варианта № 30

Длину и значение операндов выбираем следующим образом:

```
A   DB  55  ; при умножении на 10 получится слово
B   DW  1950    ; при делении на байт получится байт
C   DB  31  ; к слагаемому формата байт прибавляется байт
```

Код на языке ассемблера:

```
MOV AL, 10
MUL A           ; числитель в аккумуляторе
PUSH AX
MOV AX, B
MOV BL, 195
DIV BL
ADD AL, C
XOR AH, AH      ; расширение полученной суммы до слова
```

```
    MOV BX, 529
    SUB BX, AX
    POP AX
    XOR DX, DX      ; расширение числителя до
двойного слова
    DIV BX      ; результат в аккумуляторе
```

Лабораторная работа 3
Организация обработки числовых одномерных массивов

Цель: изучение команд организации циклов и способов косвенной адресации данных памяти в микропроцессоре *i486*; приобретение практических навыков составления программ обработки одномерных массивов, освоение методов анализа трудоемкости и ресурсной сложности алгоритмов обработки одномерных числовых массивов.

Теоретические положения

Организация циклов в программе

Цикл можно организовать с использованием команд условного перехода.

Пример: код циклического увеличения содержимого регистра BX от нуля до установленного предела.

```
        XOR BX, BX ; Обнуление регистра BX
Label1:    ADD BX, 20d ; Увеличение содержимого BX на 20d
     CMP   BX, 443d   ; Сравнение содержимого BX с константой
     JL    Label1     ; Jump if Low
```

На выходе из данной программы содержимое BX равно 460d.

Цикл можно организовать с помощью команд цикла **LOOP** (LOOPE/LOOPNE). Команда LOOP использует регистр CX в качестве счетчика цикла, она уменьшает регистр CX на 1 и передает управление на метку, если содержимое CX не равно 0. Если вычитание 1 из регистра CX привело к его обнулению, команда LOOP не делает перехода и выполняется следующая команда.

Пример:

```
MOV CX, LOOP_COUNT
```

```
Label: ; …тело цикла
       LOOP Label
```

Как видно из примера, в начале программы необходимо выполнять инициализацию регистра CX, загружая в него число повторений цикла.

Если LOOP_COUNT был равен 0, то цикл выполнится 65 536 раз.

Команда **LOOPE** – выходит из цикла, если флаг нуля = 1 или если в регистре CX образовался 0. Тем самым становится возможным двойственное завершение цикла.

Команда **LOOPNE** – выходит из цикла, если флаг нуля <> 0 или если в регистре CX образовался 0. Цикл можно организовать с помощью команды **JCXZ** – переход, если CX = 0. Команда не проверяет ни одного флага и не влияет ни на один из них.

Примечание. Эта команда обычно применяется в начале цикла (LOOP), чтобы пропустить тело цикла, если переменная счетчика (CX) равна нулю.

Организация ветвления по заданному условию

Для организации ветвления по условию используется комбинация команд **CMP** с одной из команд условного перехода. Команда **CMP** сравнивает два операнда и воздействует на флаги AF, CF, OF, PF, SF, ZF (см. разд. 1). Однако нет необходимости проверять все эти флаги по отдельности. В следующем примере проверяется: содержит ли регистр BX нулевое значение:

```
CMP BX, 00 ; Сравнение BX с нулем
JZ  M1     ; Переход на  M1 если нуль
   ; (действия при не нуле)
 M1: ...       ; Точка перехода при BX=0
```

Если BX содержит нулевое значение, команда **CMP** устанавливает флаг нуля ZF в единичное состояние, и возможно изменяет (или нет) другие флаги. Команда **JZ** (переход, если нуль) проверяет только флаг ZF. При единичном значении ZF, обозначающем нулевой признак, команда передает управление на адрес, указанный в ее операнде, т.е. на метку M1.

Следует пояснить характер использования команд условного перехода. Типы данных, над которыми выполняются арифметические операции и операции сравнения, определяют, какими командами пользоваться: беззнаковыми или знаковыми.

Беззнаковые данные используют все биты как биты данных; характерным примером являются символьные строки: имена, адреса и натуральные числа.

В знаковых данных самый левый бит представляет собой знак, причем если его значение равно нулю, то число положительное, и если единице, то отрицательное. Многие числовые значения могут быть как положительными так и отрицательными.

В качестве примера предположим, что регистр AX содержит 11000110, а BX – 00010110. Команда **CMP AX, BX** сравнивает содержимое регистров AX и BX. Если данные беззнаковые, то значение в AX больше, а если знаковые – то меньше (табл. 3.1).

Таблица 3.1

Переходы для беззнаковых данных

Мнемоника	Описание	Проверяемые флаги
JE/JZ	Переход, если равно/нуль	ZF
JNE/JNZ	Переход, если не равно/не нуль	ZF
JA/JNBE	Переход, если выше/не ниже или равно	ZF, CF
JAE/JNB	Переход, если выше или равно/не ниже	CF
JB/JNAE	Переход, если ниже/не выше или равно	CF
JBE/JNA	Переход, если ниже или равно/не выше	CF, AF

Любую проверку можно кодировать одним из двух мнемонических кодов. Например, **JB** и **JNAE** генерирует один и тот же объектный код, хотя положительную проверку **JB** легче понять, чем отрицательную **JNAE** (табл. 3.2).

Таблица 3.2

Переходы для знаковых данных

Мнемоника	Описание	Проверяемые флаги
JE/JZ	Переход, если равно/нуль	ZF
JNE/JNZ	Переход, если не равно/не нуль	ZF
JG/JNLE	Переход, если больше/не меньше или равно	ZF, SF, OF
JGE/JNL	Переход, если больше или равно/не меньше	SF, OF
JL/JNGE	Переход, если меньше/не больше или равно	SF, OF
JLE/JNG	Переход, если меньше или равно/не больше	ZF, SF, OF

Команды перехода для условия равно или ноль (**JE**/**JZ**) и не равно или не ноль (**JNE**/**JNZ**) присутствуют в обоих списках для беззнаковых и знаковых данных (табл. 3.3). Состояние равно/нуль происходит вне зависимости от наличия знака.

Таблица 3.3

Специальные арифметические проверки

Мнемоника	Описание	Проверяемые флаги
JS	Переход, если есть знак (отрицательно)	SF
JNS	Переход, если нет знака (положительно)	SF
JC	Переход, если есть перенос (аналогично JB)	CF
JNC	Переход, если нет переноса	CF
JO	Переход, если есть переполнение	OF
JNO	Переход, если нет переполнения	OF
JP/JPE	Переход, если паритет четный	PF
JNP/JP	Переход, если паритет нечетный	PF

Организация последовательного перебора элементов одномерного массива

Для организации последовательного доступа к элементам массива (находящегося в сегменте данных памяти) необходимо использовать один из следующих методов косвенной адресации:

– базовая – в регистр BX загружается адрес начала массива, затем в цикле значение этого регистра увеличивается на 2, если длина элемента массива равна слову, или на 1, если – байту;

– индексная – аналогично предыдущему, только адрес начала массива загружается в один из индексных регистров;

– базовая со смещением – в регистр BX загружается 0. При адресации к элементу указываем [BX + A], где A – имя массива. Затем в цикле значение регистра BX увеличивается на 2, если длина элемента массива равна слову, или на 1, если байту;

– индексная со смещением – аналогично предыдущему, только 0 загружается в один из индексных регистров.

Пример: подсчитать количество отрицательных элементов в массиве A(20), где каждый элемент имеет разрядность 2 байта, т.е. слово.

```
        Dosseg
        .model small
        .stack      100h
        .data
A       dw   7, –3, 5, 2, 4, 8, –11,   ;объявляем массив чисел
27, 6, 4, 8, –9, 5, 3, 11, 1, 12, –7,
14, 9
count       db   0                     ;объявляем переменную
        .code                           счетчик
        mov  ax,@data
        mov  ds,ax                      ; устанавливаем регистр DS
        mov  cx,20                      ; в начало сегмента данных
        sub  si,si                      ; заносим длину массива
L1:     mov  bx, [a+si]                 ; обнуляем SI
        cmp  bx, 0                      ; заносим в BX значение из
```

```
        JGE  L2          массива
        inc  count       ; сравниваем с 0
L2:     add  si,2        ; если >=0, то переход на
        loop L1          метку L2
        mov  ah, 4ch     ; увеличиваем счетчик
        int  21h         ; переходим к следующему
        end              элементу
                         ; пока не конец – на L1
                         ; 4Ch – функция выхода из
                         ; программы по 21h
                         прерыванию
```

Задание

1. На основании индивидуального задания (табл. 3.4) составить программу для обработки элементов одномерного массива. Длина элементов исходного массива равна DW. Значения элементов исходного массива задать в сегменте данных (см. пример выше). Длину элементов результирующего массива, если он необходим, выбрать самостоятельно.

2. Получить загрузочный модуль и протестировать выполнение программы.

3. Выполнить расчет времени выполнения программы.

Таблица 3.4

Варианты индивидуальных заданий

№ вар.	Задание на обработку
1	Дан массив A[20]. Подсчитать количество элементов, делителем которых является число 4. Программу составить без использования команды деления
2	Сформировать массив из младших байтов массива A[30], в которых установлен старший разряд. Найти сумму элементов нового массива
3	Дан массив A[30]. Найти минимальный элемент нового массива B, элементы которого получены по алгоритму: B[1]=A[1]+A[2], B[2]=A[1]+A[3], … , B[29]=A[1]+A[30]
4	Поменять местами пять младших разрядов младшего

№ вар.	Задание на обработку
	байта каждого элемента массива A[25] с пятью младшими разрядами старшего байта этого же элемента. Найти максимальный элемент нового массива
5	Найти минимальный элемент массива A[20] и максимальный элемент массива B[20]. Поменять их местами. Выполнить операцию 20 раз
6	Подсчитать число значащих единиц каждого элемента массива A[20]. Из полученных чисел сформировать массив и найти количество четных элементов в этом массиве
7	Из элементов массива A[20] сформировать массив B[19], элементы которого равны –1, если каждый предыдущий элемент массива A меньше последующего; 0, если элементы равны; 1, если A[i]>A[i+1]. Подсчитать количество нулевых элементов массива B
8	Для массива A[40] найти минимальный и максимальный элементы. Вычислить Acp=(Amin+Amax)/2. Все элементы массива A, которые меньше Acp заменить на –1, равным Acp на 0, остальные на 1. Определить Acp без использования команды деления
9	В каждом элементе с четным индексом массива A[20] переставить байты. Элементы полученного массива расположить в порядке убывания
10	Дан массив A[10]. Найти минимальный элемент Amin и рассчитать его факториал (Amin!)
11	Для массива A[40] найти количество отрицательных элементов, а положительные увеличить в 4 раза без использования команд умножения
12	Сформировать новый массив из элементов массива A[40], в которых младший байт меньше нуля а старший больше нуля
13	Сформировать массив из старших байтов массива B[30] в порядке возрастания
14	Все отрицательные элементы массива A[20] увеличить на

№ вар.	Задание на обработку
	17, а затем умножить на 4. Все положительные разделить на 2 и уменьшить на 15. Найти сумму всех положительных элементов нового массива (без использования команд умножения и деления)
15	Расположить элементы массива A[20] в порядке убывания. Запомнить номера минимального и максимального элементов. Найти разницу между этими номерами
16	Определить разницу между минимальным и максимальным элементом массива A[25], четные элементы массива увеличить в 8 раз, а нечетные уменьшить в 2 раза (без использования команд умножения и деления)
17	В массиве A[40] найти минимальный элемент. Заменить 3 младших разряда старшего байта этого элемента на 3 младших разряда младшего байта. Вновь найти минимальный элемент. Операцию повторить 40 раз. Подсчитать количество элементов нового массива, у которых установлен 8-й разряд
18	В заданном массиве C[30] элементы с нулевыми тремя младшими битами переместить в начало массива, расположив их по возрастанию
19	Дан массив C[16] состоящий из 0 и 1. Составить число следующим образом – i-й бит равен i-му элементу массива
20	На основе заданного массива C[20] получить массив A[8], элементы которого соответствуют количеству отрицательных элементов в массиве C на каждом из 8-ми шагов. На каждом шаге элементы массива C циклически сдвигать на 2 разряда вправо
21	Дан массив A[56]. Найти сумму отрицательных и положительных элементов массива, инвертировать суммы
22	Дан массив C[33]. Нечетные элементы массива отсортировать по возрастанию, четные – по убыванию.

№ вар.	Задание на обработку
	Четным считается элемент, содержащий четное количество единиц
23	Дан массив A[15]. Сдвинуть элементы массива циклическим сдвигом так, чтобы минимальный элемент оказался на первом месте
24	Найти среднее арифметическое элементов массива A[32] и составить новый массив B из элементов, превышающих среднее. Полученный массив B упорядочить по возрастанию, в неотрицательных элементах поменять местами байты
25	На основе данного массива A[50] получить результирующий массив B[10], элементы которого равны сумме пяти подряд идущих элементов массива A, уменьшенных в 4 раза без использования операции деления
26	В заданном массиве B[42] элементы, содержащие единицы в 3-м и 4-м битах, переместить в конец массива, расположив их по убыванию
27	Элементы массива C[17] расположить в порядке убывания. В первых семи элементах нового массива поменять местами байты. Найти минимальный элемент среди нечетных элементов массива
28	Для заданного массива A[37] получить массив C, состоящий из удвоенных номеров неположительных элементов массива A. Элементы в массив C заносить таким образом, чтобы соблюдалось условие возрастания его элементов
29	Дан массив B[16]. Определить среднее арифметическое массива, не используя команду деления
30	Для заданного массива A[33] получить слово W, которое сформировать побитово, записывая на каждом шаге в i-й разряд W значение i-го разряда дизъюнкции элементов A[i] и A[33–i]. Полученное слово W подвергнуть операции конъюнкции с оставшимся элементом A[17]

№ вар.	Задание на обработку
31	Дан массив A[16]. Определить количество четных положительных элементов и сделать указанные элементы отрицательными
32	Дан массив B[20]. Найти максимальный положительный элемент и отрицательные элементы увеличить на найденный максимальный положительный элемент
33	Дан массив C[32]. Удвоить элементы, меньшие среднего арифметическое значение элементов массива. Удвоение произвести без использования операции умножения.
34	Дан массив Z[18]. Найти максимальный по модулю элемент массива. Положительные элементы сделать отрицательными и увеличить на найденный максимальный элемент
35	Дан массив G[30]. Найти минимальный по модулю не нулевой элемент и заменить на него нулевые элементы массива.

Лабораторная работа 4
Программирование ввода-вывода в режиме реального времени

Цель: изучение принципов организации ввода информации извне в УВМ и вывода информации из УВМ вовне, организации временной задержки при обработке данных на языке ассемблера, а также приобретение практических навыков программирования указанных операций.

Теоретические положения

Порты ввода-вывода

Порт представляет собой устройство, которое соединяет процессор с внешним миром. Через порт процессор получает сигналы с устройств ввода и посылает сигналы на устройство вывода. Теоретически процессор может управлять до 65 536 портами, начиная с нулевого порта. Для управления вводом-выводом непосредственно на уровне порта используются команды IN и OUT.

Команда IN передает данные из входного порта в регистр AL (байт) или в регистр AX (слово). Формат команды:

```
IN регистр, номер порта
```

Команда OUT передает данные в порт из регистра AL (байт) или из регистра AX (слово). Формат команды:

```
OUT номер порта, регистр
```

Номер порта можно указывать статически или динамически.

1. Статическое указание порта возможно непосредственной адресацией при непосредственном использовании значения от 0 до 255:

```
IN AL, порт# ; Ввод одного байта
OUT порт#, AX ; Вывод одного слова
```

2. Динамическое указание порта реализуется регистровой адресацией и устанавливается в регистре DX от 0 до 65535. Этот метод удобен для последовательной обработки нескольких портов.

Значение в регистре DX в этом случае увеличивается в цикле на 1. Пример ввода байта из порта 60H:

```
    MOV DX, 60H     ; Порт 60H (клавиатура)
    IN AL, DX   ; Ввод байта
```

Программная организация временной задержки

Программирование временных задержек при обработке информации на языке ассемблера осуществляется за счет того, что команды в процессоре выполняются в течение определенного числа тактов. Число тактов, за которые выполняются команды ассемблера, представлено в прил. А. Длительность же выполнения команд в единицах времени зависит от тактовой частоты процессора, который выполняет команды.

Наиболее простой алгоритм работы подпрограммы приведен на рис. 4.1. В этом алгоритме общее время задержки вычисляется по формуле

$$ТЦД=ТТ1+(ТТ2+ТТ3+ТТ4)\cdot NI+ТТ5,$$

где NI – число, первоначально записанное в счетчике. В качестве счетчика используется один из РОН BX, в который записывается число NI из регистра CX. Команда NOP традиционно используется здесь для увеличения времени выполнения цикла, а следовательно, и общей задержки. Вместо команды NOP можно использовать любую последовательность команд, выполнение которых не изменяет содержимого регистров. Время записи числа NI в регистр BX и возврата из подпрограммы фиксировано и в цикл не входит. Минимальная задержка определяется при NI=01 и равна

$$ТЦД\ мин=ТТ1+ТТ2+ТТ3+ТТ4+ТТ5;$$

максимальная задержка вычисляется по формуле:

$$ТЦД\ макс=ТТ1+(ТТ2+ТТ3+ТТ4)\cdot 65\,536+ТТ5,$$

где 65 536 – максимальное 8-ми разрядное двоичное число. Подпрограмма 4.1 реализует временную задержку, записанную в соответствии с данным алгоритмом.

Листинг 1

```
    MOV BX, CX
DLY:    NOP
```

```
LOOP DLY
RET
```

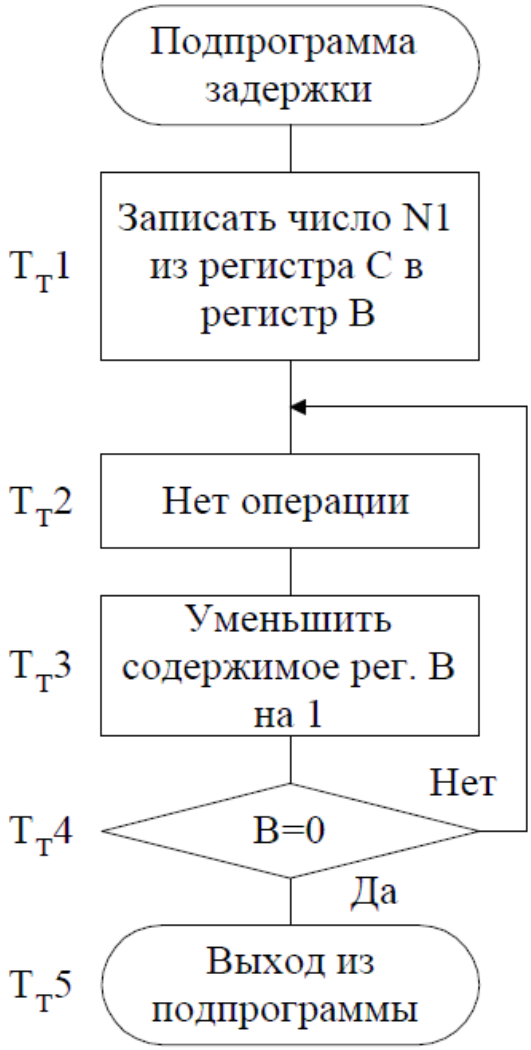


Рис. 4.1. Блок-схема алгоритма подпрограммы временной задержки

Учитывая быстродействие современных компьютеров для организации задержки в 1 мкс необходимо организовать двойной цикл. Попробуем сформировать цикл с 1000 итераций (см. листинг 2). Число тактов, за которое выполняется эта подпрограмма:

$$T= T_{CALL}+T_{MOV}+4*(T_{MOV}+250\cdot6*T_{NOP}+249\cdot T_{LOOP\_with\_jump}+ +T_{LOOP\_without\_jump}+T_{DEC}+T_{JNZ\_with\_jump}) - T_{JNZ\_with\_jump}+T_{JNZ\_without\_jump}+T_{RET}$$

$$T=29+4+4\ (4+249\cdot6\cdot3+241\cdot17+5+3+16)–16+4+20=32\ 213 \text{ тактов}$$

Листинг 2

```
DELAY:  MOV BX,     ; задаётся 1000 итераций двойным
        #4          циклом
    L1: MOV CX,     ; 4 такта
        #F9
    L2: NOP         ; 3 такта
        NOP
        NOP
        NOP
        NOP
        NOP
        LOOP L2     ; 5 тактов без перехода, 17 тактов с
                    переходом
        DEC BX      ; 3 такта
        JNZ L1      ; 4 такта без перехода, 16 тактов с
                    переходом
        RET         ; 20 тактов
```

При тактовой частоте процессора 3,4 ГГц длительность одного такта процессора приблизительно составляет 0,294 нс. Тогда длительность выполнения подпрограммы составит T_{DELAY}=32213·0,294=9471 нс. Для того, чтобы довести ее точно до 10 мкс пришлось метом подбора понизить число итераций внутреннего цикла до 242 (грубая доводка) и добавить 4 операции NOP в тело внешнего цикла (тонкая доводка). При этом число тактов составит:
T=29+4+4· (4·3+4+242·6·3+241·17+5+3+16)–16+4+20=34 013 тактов
а длительность выполнения подпрограммы:
T_{DELAY}=34013·0,294=9999,82 нс≈10 мкс.

Листинг 3 демонстрирует подпрограмму после доводки.

Листинг 3

```
DELAY:  MOV BX,     ; задержка на 10 мкс при тактовой
        #4          частоте 3,4 ГГц
    L1: MOV CX,     ; число итераций сокращаем до 968
        #F2
        NOP         ; добавляем 4 пустых операции во
                    внешнем
        NOP         ; цикле
```

```
        NOP
        NOP
L2:     NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        NOP
        LOOP L2         ; 5 тактов без перехода, 17 тактов с
                          переходом
        DEC BX          ; 3 такта
        JNZ L1          ; 4 такта без перехода, 16 тактов с
                          переходом
        RET             ; 20 тактов
```

Если в задаче необходимо также реализовать задержку на 1 мс, то можно использовать подпрограмму задержки на 10 мкс, вызвав ее 100 раз (см. листинг 4). При этом задержка составит незначительно больше главным образом за счет стократно выполняемых команд DEC и JNZ. При необходимости можно известными способами осуществить доводку программы до точного значения.

Листинг 4

```
ONE_MSE MOV DX,
      C: 100
     L3: CALL           ; вызов задержки на 10 мкс
         DELAY
         DEC DX
         JNZ L3
         RET
```

Аналогичным образом можно программировать более значительные временные задержки. Так, максимальное число тактов выполнения подпрограммы на листинге 4.2 достигается за счет загрузки регистров-счетчиков максимально возможными операндами и составит:

T=29+4+65 536·(4+65536·6·3+65535·17+5+3+16)–
16+4+20=150 324 576 297 тактов.
При этом временная задержка для процессора с тактовой частотой 3,4 ГГц:
T_{DELAY}=150 324 576 297·0,294 ≈ 44,2 с.

Использование тройного цикла может многократно нарастить задержку до совершенно запредельных значений. Хотя при необходимости организации больших временных задержек целесообразно использовать показания системных часов компьютера, показания которых можно считать посредством вызова функции 2Ch прерывания INT 21h.

Задание

Имеется СРВ, включающая в себя некоторую аппаратную часть периферийных устройств (ПУ) и ядро в виде УВМ (рис. 4.2), которая осуществляет обмен с периферией через один 16-разрядный порт ввода с заданным адресом A_{IN} и один 16-разрядный порт вывода с заданным адресом A_{OUT}. Входные 8-разрядные данные поступают на младший байт порта ввода.

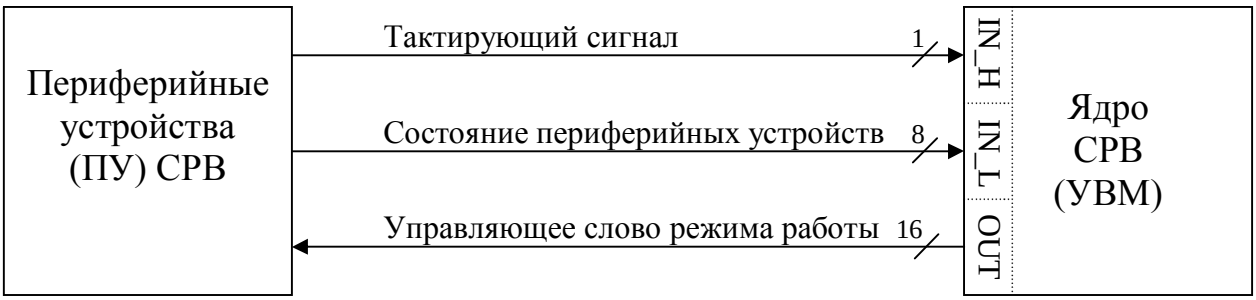


Рис. 4.2. Структурная схема СРВ

Для актуализации входных данных необходимо предварительно выводить через порт вывода заданное управляющее слово режима работы (УСРР). Отправление УСРР тактируется высоким уровнем сигнала через заданный разряд R_C порта ввода.

Перед отправлением УСРР его необходимо модифицировать в 2 этапа:

1. Начальное значение старшего байта УСРР (УСРР[1]) – заданная комбинация U_0. При каждом следующем запросе данных

значение УСРР[1] меняется на заданное приращение ΔU. Вследствие аппаратных временных затрат периферийной части актуализация данных происходит через заданный интервал времени τ.

2. Младший байт УСРР (УСРР[0]) подвергается побитовой модификации следующим образом:

– на каждой нечётной итерации заданные биты SR устанавливаются, а биты RS сбрасываются;
– на каждой чётной итерации заданные биты SR сбрасываются, а биты RS устанавливаются.

Составить программу на языке ассемблера, которая осуществляет опрос внешних устройств через порт ввода и записывает отправленные УСРР и соответствующие им данные о состоянии периферийных устройств в одномерные массивы заданной длины *N*. Реализацию задержки времени τ осуществлять при помощи подпрограммы DELAY. Блок-схема алгоритма управляющей программы показана на рис. 4.3. Заданные параметры СРВ представлены в таблице.

В отчете представить:

– тему, цель, задание с рис. 4.2 и заданными параметрами работы СРВ;
– процедуру расчета времени задержки при составлении подпрограммы DELAY;
– текст составленной программы;
– дамп памяти, отображающий содержимое результатных массивов
– выводы.

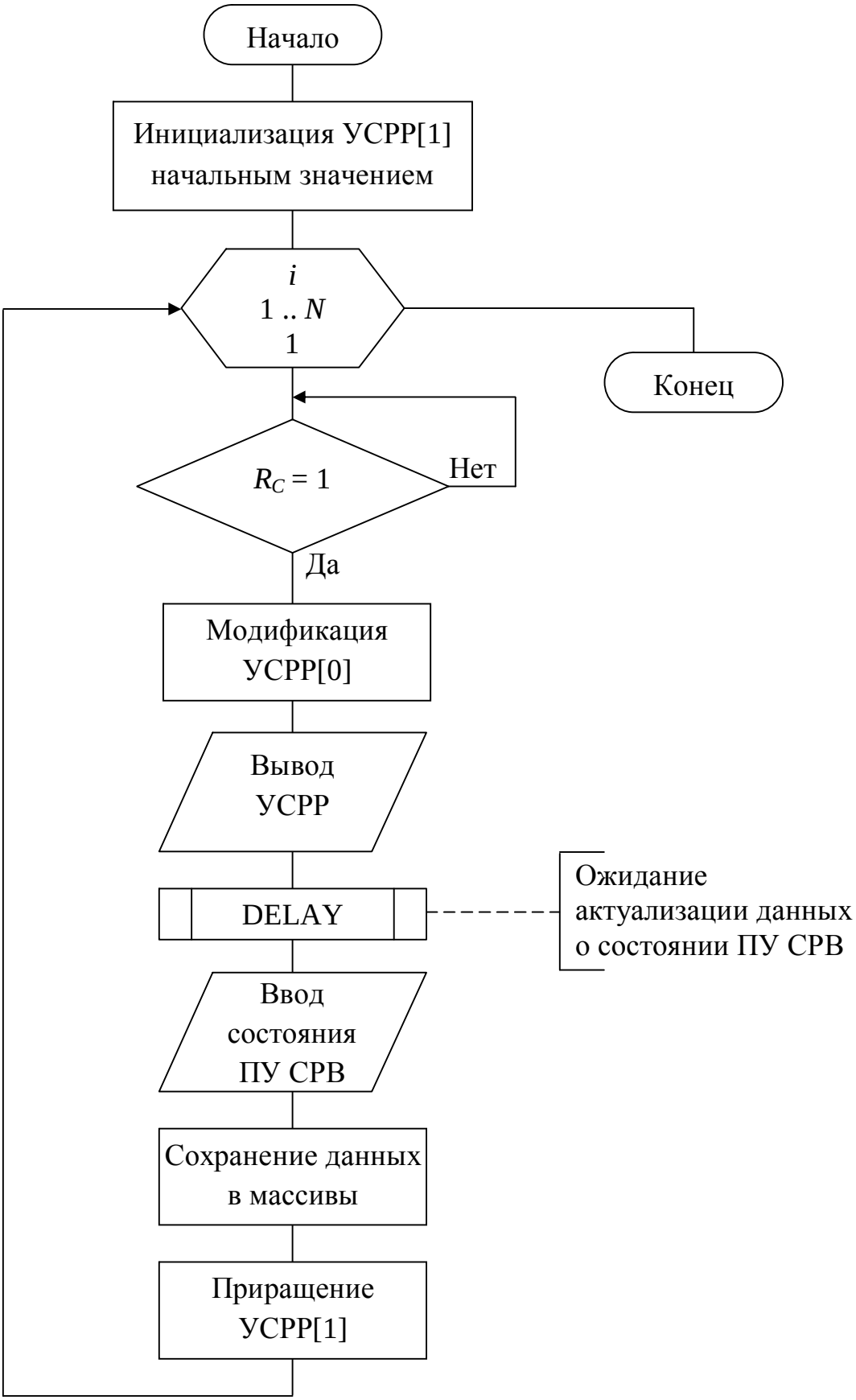


Рис. 4.3. Укрупненная блок-схема алгоритма функционирования СРВ

Заданные параметры работы СРВ

Номер варианта	Адреса портов		Параметры УСРР, байт				Параметры синхронизации		
			старший		младший				
	A_{IN}	A_{OUT}	U_0	ΔU	SR	RS	τ, мс	R_C	N
1	501h	500h	01h	10h	1, 2	6, 7	145	15	15
2	503h	502h	03h	08h	2, 4	5	8,2	14	30
3	505h	504h	05h	05h	0	4, 7	9	13	32
4	507h	506h	0Ah	04h	2, 5	0	5,6	12	48
5	509h	508h	80h	04h	1, 6	4	17	11	30
6	50Bh	50Ah	83h	02h	0-3	7	0,75	10	50
7	50Dh	50Ch	85h	07h	3, 4	5, 7	25	9	18
8	50Fh	50Eh	88h	0Ah	3-6	7	98	8	10
9	8A5h	8A6h	8Ah	03h	2, 7	1, 4	1,25	9	36
10	8A7h	8A8h	8Ch	06h	6	4, 7	51	10	20
11	8A9h	8AAh	66h	0Ch	7	5, 6	112	11	12
12	8ABh	8ACh	6Ah	0Bh	2, 3	4	203	12	14
13	8ADh	8AEh	6Dh	0Ah	0, 6	5, 7	114	13	15
14	8AFh	8B0h	0Fh	09h	7	4-6	33	14	25
15	8B1h	8B2h	0Eh	0Bh	0-2	3-5	48	15	21
16	8B3h	8B4h	0Ch	0Ch	3, 5	4, 6	54,5	14	20
17	8B5h	8B6h	0Ah	0Dh	0-5	7	103	13	18
18	8B7h	8B8h	08h	0Eh	0-3	5	186	12	16
19	8B9h	8Bah	04h	0Fh	2-5	6	196	11	15
20	8BBh	8BCh	02h	14h	0, 1	2	205	10	12
21	079h	07Ah	01h	18h	3, 4	1, 5	245	9	10
22	07Bh	07Ch	10h	14h	5	0, 7	144,5	8	12
23	07Dh	07Eh	20h	12h	5, 7	1	102	9	13
24	07Fh	080h	40h	11h	6, 7	0	44,8	10	11
25	081h	082h	80h	11h	5-7	1, 3	355	11	6
26	083h	084h	A0h	0Ah	3, 7	4	253	12	8
27	085h	086h	B0h	08h	5	6, 7	201	13	8
28	087h	088h	C0h	06h	3-7	1	180	14	10
29	089h	08Ah	D0h	04h	2, 6	0	222	15	10
30	08Bh	08Ch	E0h	02h	0, 3	5	168	14	15
31	9B1h	9B2h	1Eh	0Bh	5, 6	4, 7	55	15	21
32	9B3h	9B4h	1Ch	0Ch	5-7	2	4,5	14	20
33	9B5h	9B6h	1Ah	0Dh	1, 2	3	10,3	13	18
34	9B7h	9B8h	18h	0Eh	6	0, 5	200	12	16
35	9B9h	9Bah	14h	0Fh	6	0, 2	19,6	11	15

Лабораторная работа 5
Организация обмена между датчиками, УВМ и исполнительными устройствами

Цель: изучение принципов организации инфообмена между ядром СРВ и периферийными устройствами; приобретение практических навыков разработки функциональных схем, алгоритмов и управляющих программ для управления технологическими процессами с использованием измерительных преобразователей и исполнительных устройств.

Теоретические положения

Для ввода аналоговой информации в компьютерную систему используются аналого-цифровые преобразователи (АЦП). Это устройство характеризуется:

- временем преобразования входного аналогового сигнала в двоичный код;
- диапазоном входного аналогового сигнала;
- разрядностью выходного двоичного кода.

Первая характеристика влияет на то, что входной непрерывный сигнал преобразуется в последовательность дискретных отсчетов в определенные моменты времени. Вторая и третья характеристики влияют на точность преобразования, которая в числовом выражении равна весу младшего разряда АЦП. Схема работы АЦП показана на рис. 5.1.

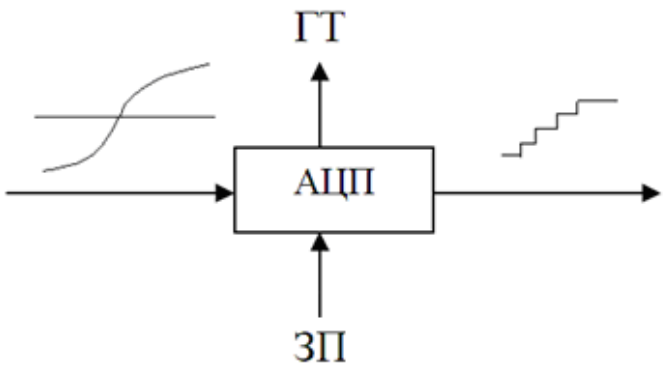


Рис. 5.1. Дискретизация сигнала по времени

ЗП – запуск АЦП – сигнал, по фронту (или спаду) которого значение на входе фиксируется в запоминающем устройстве (устройство выборки/хранения) и начинается его преобразование в двоичный код.

ГТ – готовность – сигнал, который информирует о том, что преобразования завершены и двоичный код на выходе АЦП соответствует аналоговому значению на входе в момент поступления сигнала ЗП. Таким образом, существует интервал от момента поступления сигнала запуска до момента появления сигнала готовности (время, требуемое для преобразования и зависящее от типа АЦП и количества значащих единиц в результирующем коде).

Квантование по уровню

Предположим, что аналоговый сигнал на входе АЦП меняется от 0 до 3В, а АЦП имеет следующие параметры:

– разрядность равна 10;

– допустимый диапазон входного сигнала 0–5 В.

Необходимо определить, какой двоичный код на выходе АЦП соответствует входному сигналу 2В. Возможное количество комбинаций на выходе $2^{10} = 1024$. Комбинации могут меняться от 0 до 1023. Точность преобразования АЦП, а значит, и вес одного двоичного разряда равен

$$\frac{5B - 0B}{1024} = 0{,}0048828125 \text{ В/бит.}$$

Точность преобразования означает, что изменение входного сигнала на величину, меньшую 0,0048828125 В, никак не повлияет на изменение двоичного кода на выходе АЦП. То есть каждой из $2^{10} = 1024$ выходных комбинаций соответствует определённый уровень входного напряжения.

Для определения двоичного кода на выходе АЦП, соответствующего входному аналоговому сигналу 2 В, необходимо разделить его на точность:

$$2\text{В}/0{,}0048828125 = 409{,}6_{10} = 110011001_2.$$

Таким образом, зная уровень входного сигнала *u*, разрядность АЦП *n*, а также предельные значения

преобразования u_1 и u_2, можно рассчитать соответствующий сигналу u двоичный эквивалент по следующей формуле:

$$D_{10} = ROUND\left[\frac{u-u_1}{u_2-u_1}\cdot 2^n\right] \rightarrow D_{16}.$$

По окончании преобразования АЦП устанавливает выход ГТ в единицу, что является признаком достоверности кода.

Принцип работы цифроаналогового преобразователя аналогичен АЦП. Отличие заключается только в том, что для ввода аналоговой информации от нескольких аналоговых датчиков необходимо к АЦП добавлять коммутатор, который переключает входной сигнал с одного датчика на другой, а при выдаче управляющего аналогового сигнала на несколько исполнительных механизмов необходимо использовать соответствующее количество ЦАП.

Градуировочная характеристика датчика

Под градуировочной характеристикой датчика понимается зависимость выходного сигнала датчика (напряжение или ток) от изменения измеряемого параметра. Как правило, это линейная характеристика (рис. 5.2). В случае нелинейности градуировочной характеристики можно с достаточной точностью свести ее к линейной на небольшом интервале изменения входного параметра.

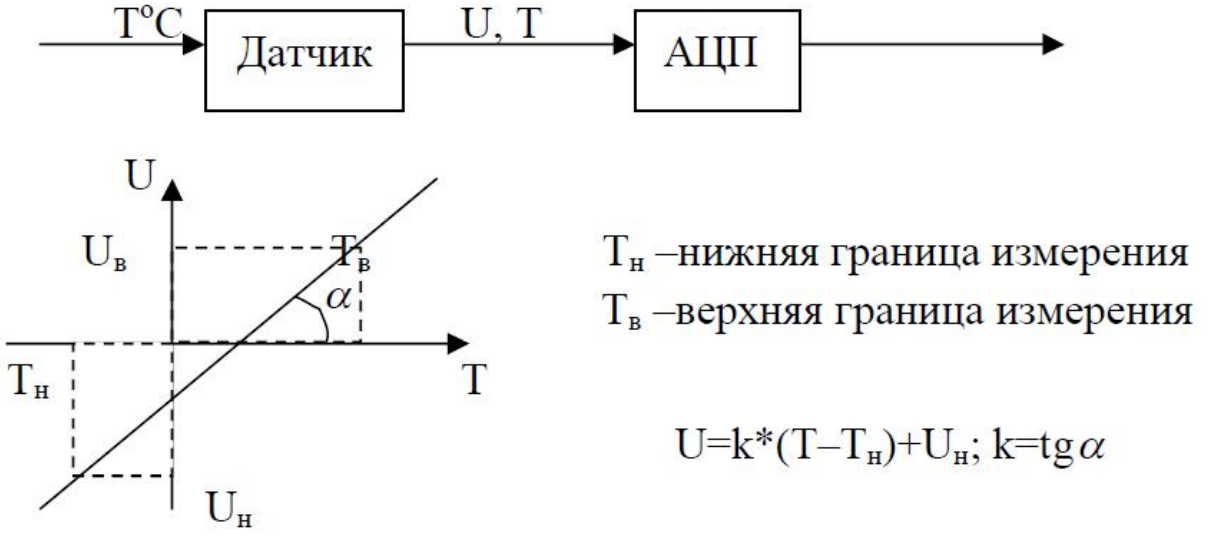


Рис. 5.2. Градуировочная характеристика датчика

Тн – Тв определяет интервал измерения параметра датчиком.

Uн – Uв определяет интервал изменения выходного сигнала датчика (возможно после усиления).

Рассмотрим следующий пример: дан объект, для измерения характеристик которого используются датчики температуры и давления. Датчики опрашиваются последовательно. Текущие показания датчиков сохраняются в памяти и в следующем цикле управления заменяются. Цикл опроса датчиков равен двум секундам. Если значение температуры объекта достигло 90 °С, необходимо выдать аналоговое управляющее воздействие уровня 14,8 В на исполнительный механизм. Структурная схема системы управления и назначение разрядов входных и выходных портов интерфейса связи приведена на рис. 5.3.

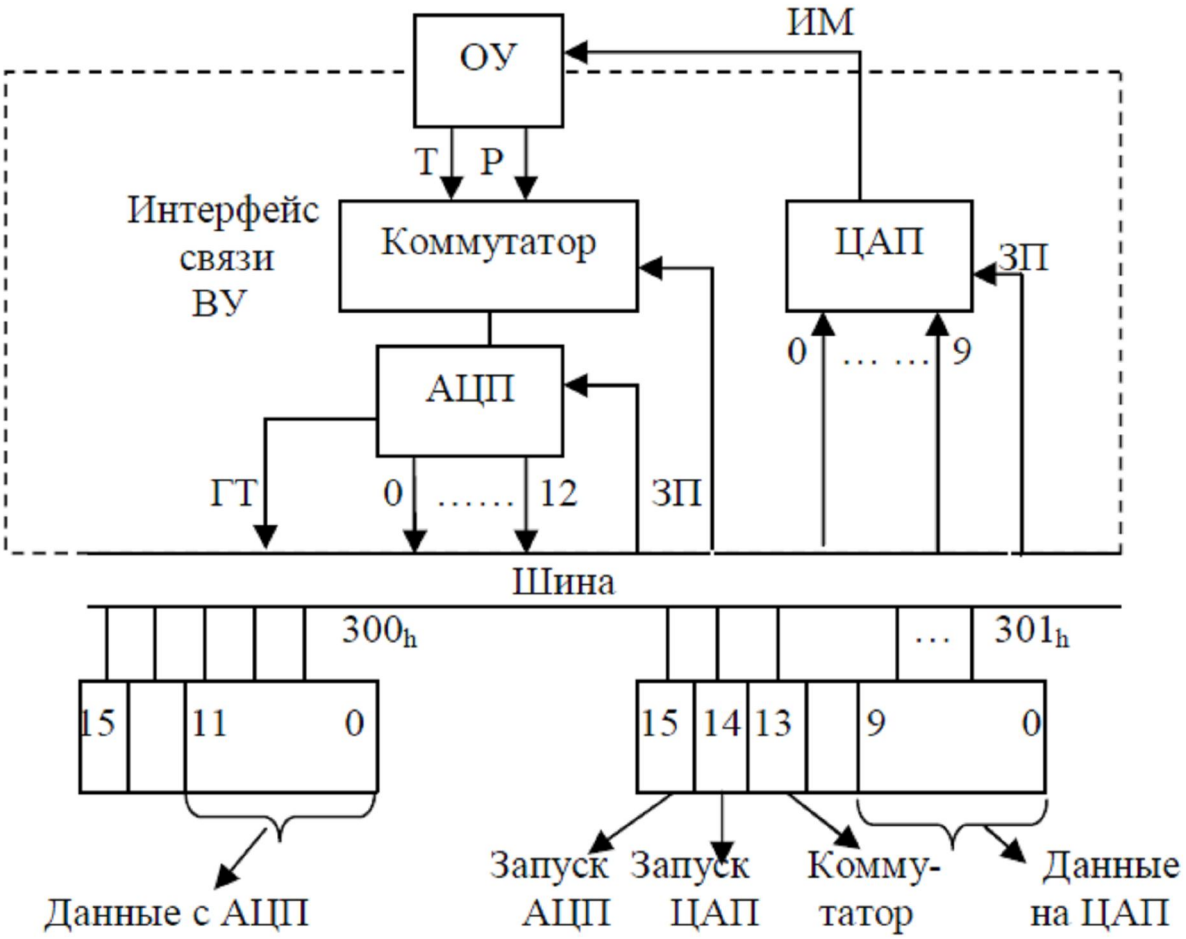


Рис. 5.3. Структурная схема СРВ в примере

300 h – адрес входного порта.

301 h – адрес выходного порта.

Параметры датчика температуры:

– диапазон входного сигнала –40…240 °С;

– диапазон выходного сигнала 0…10 В.

Параметры АЦП:

– разрядность – 12;

– диапазон входного сигнала 0…12 В.

Параметры ЦАП:

– разрядность – 10;

– диапазон выходного сигнала –24…24 В.

Управляющий сигнал на коммутатор = 0 – измерение температуры.

Управляющий сигнал на коммутатор = 1 – измерение давления.

Время коммутации сигнала – 20 мс.

Градировочная характеристика датчика температуры показана на рис. 5.4.

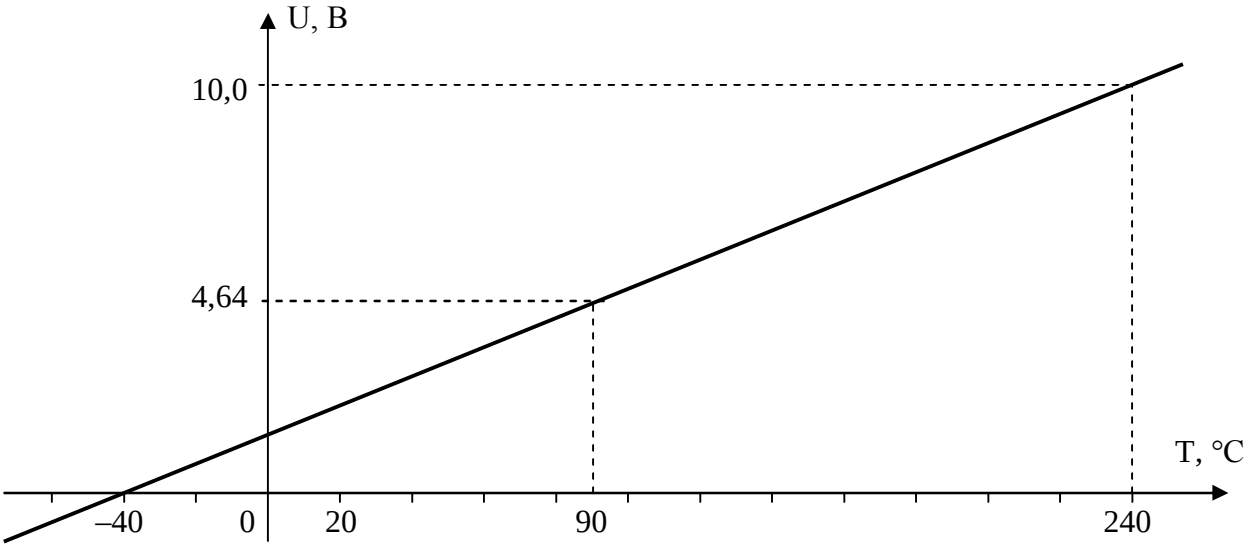


Рис. 5.4. Градировочная характеристика датчика в примере

Рассчитаем контрольный уровень напряжения на выходе термодатчика, соответствующего температуре 90°С:

$U=k\cdot(T-T_н)+U_н$; $k=tg\alpha$.

$k=tg\alpha=(10-0)/(240-40)=1/28$.

$U=1/28\ (90-40)+0=130/28\approx4{,}64$ В.

Рассчитаем цифровой эквивалент на выходе АЦП уровня напряжения на выходе термодатчика, соответствующего температуре 90°С:

$$D_{10} = ROUND\left[\frac{4{,}64 - -0}{12 - 0} \cdot 2^{12}\right] = 1584, \qquad D_{16} = \mathbf{630h}.$$

Рассчитаем цифровой эквивалент на входе ЦАП, соответствующий уровню управляющего воздействия 14,8 В:

$$D_{10} = ROUND\left[\frac{14{,}8 - -24}{24 - -24} \cdot 2^{10}\right] = 828, \qquad D_{16} = \mathbf{33Ch}.$$

Программа, обеспечивающая выполнение поставленной задачи, приведена далее. Опрос датчиков выполняется в режиме опроса по флагу. Задержка между циклами опроса датчиков осуществляется программно.

```
Met0:     mov ax, 0
          mov dx, 301h   ; Вывод на коммутатор 0
          out dx, ax
          or ax, 8000h   ; запуск АЦП, чтобы не изменялось
                         ; значение коммутатора
          out dx, ax
          mov dx, 300h
Met1:     in ax, dx      ; опрос порта с адресом 300h по флагу
          test ax, 8000h
          jz Met1
          and ax, 0FFFh  ; выделение информационных разрядов
          mov t, ax
          cmp ax, 0630h  ; сравнение с контрольным значением
          mov ax, 0
          jb Met_2
          mov ax, 033Ch
          ; опрос датчика давления
Met_2:    or ax, 6000h   ; программирование запуска АЦП
                         ; и коммутации датчика давления
          mov dx, 301h
          out dx, ax     ; вывод управляющих воздействий
          call Time20ms  ; ожидание коммутации
          or ax, 8000h   ; программирование запуска АЦП
          out dx, ax
          mov dx, 300h
Met3:     in ax, dx
          test ax, 8000h
          jz Met3
          and ax, 0FFFh
          mov p, ax
          call Time2Sec ; Временная задержка на 2 с.
          jmp Met0
```

Задание

Задан технологический процесс или функционирование объекта управления, контроль за состоянием которого осуществляется с помощью аналоговых и дискретных датчиков. Для аналоговых датчиков определены допустимый диапазон измерений, диапазон выходного сигнала, инерционность и интервал или режим опроса. Известен АЦП, преобразующий входную аналоговую информацию в двоичный код. В соответствии с измеряемыми значениями необходимо выдать аналоговое управляющее воздействие на исполнительные устройства, регулирующие протекание технологического процесса или состояние объекта управления. Для вывода аналоговой информации используется ЦАП. Разрядности АЦП и ЦАП, диапазоны аналоговых сигналов и максимальное время преобразования ЦАП заданы в табл. 5.1. Варианты технологических процессов приведены в прил. Б. По согласованию с преподавателем технологический процесс может быть скорректирован.

Таблица 5.1

Варианты выбора характеристик преобразователей

№ вар.	Разрядность АЦП	Разрядность ЦАП	Диапазон напряжения, В		Макс. время преобразования ЦАП, мс
			Входной сигнал	Выходное воздействие	
1	10	–	0 … 1	–	–
2	12	14	–10 … 10	–5 … 5	10
3	10	8	0 … 30	0 … 90	20
4	16	10	0 … 24	0 … 43,5	160
5	12	10	–10 … 10	0 … 43,5	160
6	10	–	0…9,6	–	–
7	16	–	0…24	–	–
8	12	10	–10 … 10	–6,4…12,8	80
9	14	8	0…10	0…80	45
10	11	–	–0,5…0,5	–	–

№ вар.	Разрядность АЦП	Разрядность ЦАП	Диапазон напряжения, В		Макс. время преобразования ЦАП, мс
			Входной сигнал	Выходное воздействие	
11	10	8	–2 … 2	0…10	–
12	6	9	–2…12	0…12	–
13	10	16	0…64	0…24	–
14	12	–	–10…10	–	–
15	12	10	–5…5	–5…5	12
16	10	8	0…15	0…10	–
17	8	10	–10…10	0…24	150
18	8	8	0…9,6	0…10	–
19	14	–	0…36	–	240
20	10	12	0,5… 4,5	–5…5	60
21	16	8	0…48	0…12	250
22	10	10	0…2	8…24	–
23	–	–	–	–	–
24	8	10	0…20	0…0,1	–
25	10	–	0…32	–	–
26	16	10	0…6,1	–24…24	450
27	8	–	0…24	–	–
28	10	12	0…24	0…100 мА	–
29	–	8	–	0…36	25
30	8	8	0…19,5	0…120	–
31		10		0…300 мА	
32	–	10	–	0…40 мА	–
33	–	6	–	0…16	–
34	14	–	0…10	–	–
35	12	8	0…80	0…5	200
36	10	10	0…5	0…110	–

Порядок выполнения

1. Сформировать список устройств для устройства связи с объектом (УСО).
2. Построить градуировочную характеристику для каждого аналогового датчика (при наличии).
3. Рассчитать двоичные эквиваленты контрольных и управляющих непрерывных величин.
4. Определить количество, разрядность и назначение портов ввода-вывода.
5. Определить назначение отдельных разрядов портов.
6. Разработать структурную схему интерфейса связи.
7. Разработать алгоритм функционирования объекта.
8. Рассчитать параметры процедуры временной задержки (при необходимости).
9. Составить программу управления (драйвер) на языке ассемблера.
10. На языке высокого уровня разработать приложение, в графическом режиме имитирующее функционирование СРВ. Обеспечить индикацию контрольных и управляющих величин. Предусмотреть возможность оперативного изменения значений контрольных параметров объекта. Масштаб времени может быть любым.

Содержание отчета:

- исходные данные для выполнения работы;
- расчеты двоичных эквивалентов контрольных и управляющих величин;
- структурная схема УСО;
- укрупненный алгоритм функционирования системы;
- текст управляющей программы;
- изображения, порождаемые имитационной программой;
- выводы.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

Программирование на языке ассемблера

1. Гаркуша, О.В., Добровольская, Н.Ю. Ассемблер в примерах и задачах: учебное пособие / О.В. Гаркуша, Н.Ю. Добровольская; Министерство науки и высшего образования Российской Федерации, Кубанский государственный университет. − Краснодар: Кубанский гос. ун-т, 2022. − 134 с.
2. Программирование на языке Ассемблера IA-32 в среде RADAsm [Текст] : учебное пособие / Ю. В. Кольцов, О. В. Гаркуша, Н. Ю. Добровольская, А. В. Харченко ; М-во образования и науки Рос. Федерации, Кубанский гос. ун-т. - Краснодар : [Кубанский государственный университет], 2014. - 38 с.
3. Абель П. Язык ассемблера для IBM PC и программирования / пер. с англ. Ю.В. Сальникова. М.: Высшая школа, 1992.
4. Джордейн Р. Справочник программиста персональных компьютеров типа IBM PC, XT и AT: пер. с англ. / предисл. Н.В. Гайского. М.: Финансы и статистика, 1992.
5. Гук М. Аппаратные средства IBM PC: энциклопедия. СПб.: Питер, 2000.
6. Зубков С.В. Assembler для DOS, Windows и Unix. М.: ДМК, 1999.
7. Каган Е.М. Электронные вычислительные машины и системы. М.: Машиностроение, 1991.
8. Скэнлон Л. Персональные ЭВМ IBM PC и XT. Программирование на языке ассемблера: пер. с англ. М.: Радио и связь, 1989.
9. Трой Д. Програмирование на языке Си для персонального компьютера IBM PC. М.: Радио и связь, 1991.
10. Фролов А.В., Фролов А.Г. Аппаратное обеспечение IBM PC. Т. 2, кн. 1–2. М.: Диалог-МИФИ, 1992.
11. Ю-Чжен Ло, Гибсон Г. Микропроцессоры семейства 8086/8088. М.: Радио и связь, 1987.
12. Юров В., Хорошенко С. Assembler: учебный курс. СПб.: Питер, 2000.

Системы реального времени

13. Прокопенко, А.В. Синтез систем реального времени с гарантированной доступностью программно-информационных ресурсов : монография / А.В. Прокопенко, М.А. Русаков, Р.Ю. Царев ; Министерство образования и науки Российской Федерации, Сибирский Федеральный университет. - Красноярск [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=364075.

14. Мясников В.И. Операционные системы реального времени: лаб. практикум / Поволжский государственный технологический университет. - Йошкар-Ола : ПГТУ, 2016. - 140 с. : табл., ил. - ISBN 978-5-8158-1773-9 ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=459493.

15. Гриценко, Ю.Б. Системы реального времени : учебное пособие / Ю.Б. Гриценко ; Федеральное агентство по образованию, ТУСУР. Кафедра автоматизации обработки информации. - Томск : ТУСУР, 2009. - 263 с. : табл., схем. ; [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=208657.

16. Эрджиес Кайхан. Распределенные системы реального времени. Теория и практика. ДМК-Пресс, 2020 г. – 382 с.

17. Аш Ж. Датчики измерительных систем: в 2 кн.; пер. с франц. М.: Мир, 1992. Кн. 1.

18. Аш Ж. Датчики измерительных систем: в 2 кн.; пер. с франц. М.: Мир, 1992. Кн. 2.

19. Датчики: справочник / З.Ю. Готра, Л.Я. Ильницкий, Е.С. Полищук [и др.]. Львов: Каменяр, 1995.

20. Датчики Холла и магниторезисторы / пер. с польск. В.И. Тихонова и К.Б. Македонской. М.: Энергия, 1971.

21. Како Н., Яманэ Я. Датчики и микро-ЭВМ: пер. с япон. Л.: Энергоатомиздат, Ленингр. отд-е, 1986.

22. Малов В.В. Пьезорезонансные датчики. 2-е изд., перераб. и доп. М.: Энергоатомиздат, 1989.

23. Микеров А.Г. Электромеханические датчики и электронные компоненты управляемых вентильных двигателей: учеб. пособие. СПб., 1999.

ПРИЛОЖЕНИЕ А

ВРЕМЯ ИСПОЛНЕНИЯ КОМАНД МП 8086

Приведенными в этом приложении двумя таблицами целесообразно пользоваться для вычисления времени, которое потребуется МП 8086 для исполнения команд программы. Время исполнения команд дается в циклах тактового генератора микропроцессора 8086. Чтобы преобразовать это время в наносекунды, необходимо умножить его на 210 (табл. А.1).

Пользуясь табл. А.2, следует иметь в виду следующее:

1. Время исполнения некоторых команд удлиняется, если их операндами являются слова, а не байты. Для этих команд время исполнения указано в виде b(w), где b и w означают числа циклов для операндов-байтов и операндов-слов соответственно.

2. Для большинства из тех команд, которые обращаются к памяти, в столбце циклов добавлена аббревиатура EA (Effective Address – исполнительный адрес). Она сообщает, что для вычисления исполнительного адреса требуются дополнительные циклы; требуемое на это время указано в табл. А.1.

3. Времена исполнения команд условной передачи управления и команд управления циклами зависят от того, должен ли быть выполнен переход. Если переход выполняется, то следует брать из столбца циклов большее число. В противном случае, если управление проскакивает к следующей команде, выбирать меньшее число.

Предположим, например, что требуется найти время исполнения команды: ADD ES:[BX], DX

Общая форма этой команды – *ADD память, регистр*, поэтому ее время исполнения дается соотношением: 9(24) + EA.

Так как в данном случае команда ADD оперирует словами, то надо взять не 9, а 24. Поскольку в соотношение входит EA, то надо добавить время вычисления адреса (табл. А.2). Операнд-приёмник имеет форму [BX], по которой находим, что надо добавить пять дополнительных циклов. Но поскольку в операнд входит указание замены сегмента (ES:), то надо добавить 2 цикла. Тогда полное исполнение равно 31 циклу тактового генератора.

Таблица А.1

Времена вычисления исполнительного адреса

Компоненты исполнительного адреса	Формат операнда	Число тактов[1]
Только сдвиг	сдвиг	6
	метка	
Только база или индекс	[BX]	5
	[BP]	
	[DI]	
	[SI]	
Сдвиг + база или индекс	[BX] + сдвиг	9
	[BP] + сдвиг	
	[DI] + сдвиг	
	[SI] + сдвиг	
База + индекс	[BX] [SI]	7
	[BX] [DI]	
	[BP] [SI]	8
	[BP] [DI]	
Сдвиг + база + индекс	[BX] [SI] + сдвиг	11
	[BX] [DI] + сдвиг	
	[BP] [SI] + сдвиг	12
	[BP] [DI] + сдвиг	

[1] При замене сегмента добавьте два такта.

Таблица А.2

Времена исполнения команд

Команда	Число тактов	Число байтов
AAA	4	1
AAD	60	2
AAM	83	1
AAS	4	1
ADC регистр,регистр	3	2
ADC регистр, память	9 (13)+EA	2–4
ADC память, регистр	16 (24)+EA	2–4
ADC регистр, непосредственный операнд	4	3–4
ADC память, непосредственный операнд	17 (25)+EA	3–6
ADC аккумулятор, непосредственный операнд	4	2–3
ADD регистр, регистр	3	2
ADD регистр, память	9 (13)+EA	2–4
ADD память, регистр	16 (24)+EA	2–4

Продолжение табл. А.2

Команда	Число тактов	Число байтов
ADD регистр, непосредственный операнд	4	3–4
ADD память, непосредственный операнд	17 (25)+EA	3–6
ADD аккумулятор, непосредственный операнд	4	2–3
AND регистр, регистр	3	2
AND регистр, память	9 (13)+EA	2–4
	16 (24)+EA	2–4
AND регистр, непосредственный операнд	4	3–4
AND память, непосредственный операнд	17 (15)+EA	3–6
AND аккумулятор, непосредственный операнд	4	2–3
CALL процедура-атрибут FAR	23	3
CALL процедура-атрибут NEAR	36	5
CALL указатель-память 16	29	2–4
CALL указатель-регистр 16	24	2
CALL указатель-память 32	57	2–4
CBW	2	1
CLC	2	1
CLD	2	1
CLI	2	1
CMC	2	1
CMP регистр, регистр	3	2
CMP регистр, память	9 (13)+EA	2–4
CMP память, регистр	9 (13)+EA	2–4
CMP регистр, непосредственный операнд	4	3–4
CMP память, непосредственный операнд	10(14)+EA	3–6
CMP аккумулятор, непосредственный операнд	4	2–3
CMPS строка_приемник, строка_источник	22 (30)	1
CMPS (повтор) строка_приемник, строка_источник	9+22 (30)/повтор	1
CWD	5	1
DAA	4	1
DAS	4	1
DEC регистр 16	2	1
DEC регистр 8	3	2
DEC память	15 (23)+EA	2–4
DIV регистр 8	80-90	2
DIV регистр 16	144-162	2
DIV память 8	(86-96)+EA	2–4
DIV память 16	(154-172)+EA	2–4
ESC непосредственный операнд, память	(8-35)+EA	2–4
ESC непосредственный операнд, регистр	2	2
HLT	2	1
IDIV регистр 8	101-112	2

Продолжение табл. А.2

Команда	Число тактов	Число байтов
IDIV регистр 16	165-184	2
IDIV память 8	(107-118)+EA	2—4
IDIV память 16	(175-194)+EA	2—4
IMUL регистр 8	80—98	2
IMUL регистр 16	128—154	2
IMUL память 8	(86—104)+EA	2—4
IMUL память 16	(138—164)+EA	2—4
IN аккумулятор, непосредственный-операнд 8	10 (14)	2
IN аккумулятор, DX	8 (12)	1
INC регистр 16	2	1
INC регистр 8	3	2
INC память	15 (23)+EA	2—4
INT 3	52	1
INT непосредственный-операнд 8 (не тип 3)	51	2
INTO	53 или 4	1
IRET	32	1
Все команды условного перехода, кроме JCXZ:		
Jccc близкая-метка	16 или 4	2
JCXZ близкая-метка	18 или 6	2
JMP близкая-метка	15	2
JMP метка-атрибут NEAR	15	3
JMP метка-атрибут FAR	15	5
JMP указатель-память 16	18+EA	2—4
JMP указатель-регистр 16	11	2
JMP указатель-память 32	24+EA	2—4
LAHF	4	1
LDS регистр 16, память 32	24+EA	2—4
LEA регистр 16, память 16	2+EA	2—4
LES регистр 16, память 32	24+EA	2—4
LOCK	2	1
LODS строка_источник	12 (16)	1
LODS (повтор) строка_источник	9+13 (17)/повтор	1
LOOP близкая_метка	17 или 5	2
LOOPE/LOOPZ близкая_метка	18 или 6	2
LOOPNE/LOOPNZ близкая_метка	19 или 5	2
MOV память, аккумулятор	10 (14)	3
MOV аккумулятор, память	10 (14)	3
MOV регистр, регистр	2	2
MOV регистр, память	8 (12)+EA	2—4
MOV память, регистр	9 (13)+EA	2—4
MOV регистр, непосредственный-операнд	4	2—3
MOV память, непосредственный-операнд	10 (14)+EA	3—6
MOV регистр-сегмента, регистр 16	2	2

Продолжение табл. А.2

Команда	Число тактов	Число байтов
MOV регистр-сегмента, память 16	8 (12)+EA	2—4
MOV регистр 16, регистр-сегмента	2	2
MOV память 16, регистр-сегмента	9 (13)+EA	2—4
MOVS строка_приемник, строка_источник	18 (26)	1
MOVS (повтор) строка_приемник, строка_источник	9+17 (25)/повтор	1
MUL регистр 8	70—77	2
MUL регистр 16	118—133	2
MUL память 8	(76—83)+EA	2—4
MUL память 16	(128—143)+EA	2—4
NEG регистр	3	2
NEG память	16 (24)+EA	2—4
NOP	3	1
NOT регистр	3	2
NOT память	16 (24)+EA	2—4
OR регистр, регистр	3	2
OR регистр, память	9 (13)+EA	2—4
OR память, регистр	16 (24)+EA	2—4
OR регистр, непосредственный-операнд	4	3—4
OR память, непосредственный-операнд	17 (15)+EA	3—6
OR аккумулятор, непосредственный-операнд	4	2—3
OUT непосредственный-операнд 8, аккумулятор	10 (14)	2
OUT DX, аккумулятор	8 (12)	1
POP регистр	12	1
POP регистр-сегмента (CS недопустим)	12	1
POP память	25+EA	2—4
POPF	12	1
PUSH регистр	15	1
PUSH регистр-сегмента (CS недопустим)	14	1
PUSH память	24+EA	2—4
PUSHF	14	1
RCL регистр, 1	2	2
RCL регистр, CL	8+4/бит	2
RCL память, 1	15 (23)+EA	2—4
RCL память, CL	20 (28)+EA+4/бит	2—4
RCR регистр, 1	2	2
RCR регистр, CL	8+4/бит	2
RCR память, 1	15 (23)+EA	2—4
RCR память, CL	20 (28)+EA+4/бит	2—4
REP	2	1
REPE/REPZ	2	1

Продолжение табл. А.2

Команда	Число тактов	Число байтов
REPNE/REPNZ	2	1
RET (внутри сегмента, без удаления значений из стека)	20	1
RET (внутри сегмента, с удалением значений из стека)	24	3
RET (между сегментами, без удаления значений из стека)	32	1
RET (между сегментами, с удалением значений из стека)	31	3
ROL регистр, 1	2	2
ROL регистр, CL	8+4/бит	2
ROL память, 1	15 (23)+EA	2—4
ROL память, CL	20 (28)+EA+4/бит	2—4
ROR регистр, 1	2	2
ROR регистр, CL	8+4/бит	2
ROR память, 1	15 (23)+EA	2—4
ROR память, CL	20 (28)+EA+4/бит	2—4
SAHF	4	1
SAL/SHL регистр, 1	2	2
SAL/SHL регистр, CL	8+4/бит	2
SAL/SHL память, 1	15 (23)+EA	2—4
SAL/SHL память, CL	20 (28)+EA+4/бит	2—4
SAR регистр, 1	2	2
SAR регистр, CL	8+4/бит	2
SAR память, 1	15 (23)+EA	2—4
SAR память, CL	20 (28)+EA+4/бит	2—4
SBB регистр, регистр	3	2
SBB регистр, память	9 (13)+EA	2—4
SBB память, регистр	16 (24)+EA	2—4
SBB регистр, непосредственный-операнд	4	3—4
SBB память, непосредственный-операнд	17 (25)+EA	3—6
SBB аккумулятор, непосредственный-операнд	4	2—3
SCAS строка_приемник	15 (19)	1
SCAS (повтор) строка_приемник	9+15 (19)/повтор	1
SHR регистр, 1	2	2
SHR регистр, CL	8+4/бит	2
SHR память, 1	15 (23)+EA	2—4
SHR память, CL	20 (28)+EA+4/бит	2—4
STC	2	1
STD	2	1
STI	2	1
STOS строка_приемник	11 (15)	1
STOS (повтор) строка_приемник	9+10 (14)/повтор	1
SUB регистр, регистр	3	2
SUB регистр, память	9 (13)+EA	2—4
SUB память, регистр	16 (24)+EA	2—4

Продолжение табл. А.2

Команда	Число тактов	Число байтов
SUB регистр, непосредственный-операнд	4	3–4
SUB память, непосредственный-операнд	17 (25)+EA	3–6
SUB аккумулятор, непосредственный-операнд	4	2–3
TEST регистр, регистр	3	2
TEST регистр, память	9 (13)+EA	2–4
TEST регистр, непосредственный-операнд	4	3–4
TEST память, непосредственный-операнд	10 (14)+EA	3–6
TEST аккумулятор, непосредственный-операнд	4	2–3
WAIT	3+5n	1
XCHG аккумулятор, регистр 16	3	1
XCHG память, регистр	17 (25)+EA	2–4
XCHG регистр, регистр	4	2
XLAT таблица_источник	11	1
XOR регистр, регистр	3	2
XOR регистр, память	9 (13)+EA	2–4
XOR память, регистр	16 (24)+EA	2–4
XOR регистр, непосредственный-операнд	4	3–4
XOR память, непосредственный-операнд	17 (15)+EA	3–6
XOR аккумулятор, непосредственный-операнд	4	2–3

ПРИЛОЖЕНИЕ Б

ЗАДАННЫЕ ТЕХНОЛОГИЧЕСКИЕ ПРОЦЕССЫ

1. Управление атомным реактором. Имеется система измерения мощности атомного реактора номинальной мощностью 85 МВт, на выходе которой формируется напряжение в интервале от 0 до 800 мВ, что соответствует измерительному диапазону от 0 до 1000 МВт. Необходимо обеспечить управление контрольными стержнями (КС) в количестве 7 шт. КС приводится в движение четырехфазным шаговым двигателем, на который подается управляющая последовательность импульсов частотой 20 Гц, показанная на рис. Б.1. Погружение КС достигается поворотом шагового двигателя влево, извлечение КС – вправо. Положение КС в активной зоне фиксируется счетчиком импульсов, при этом 3 импульса соответствуют 1% погружения КС в активную зону.

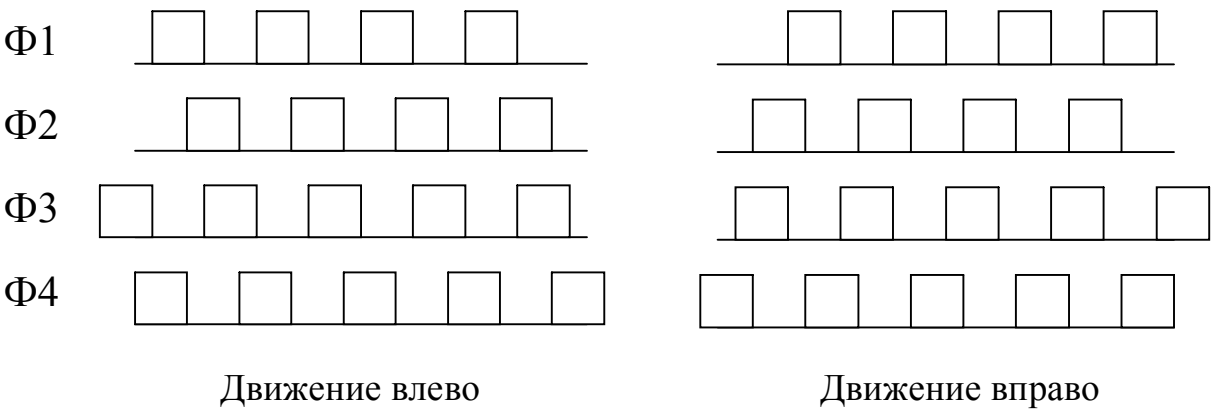


Рис. Б.1. Управляющая последовательность импульсов для движения шагового двигателя

Режим работы реактора задается при помощи 5-позиционного переключателя и обеспечивается положением КС согласно табл. Б.1.

В случае выхода мощности за пределы соответствующего выбранному режиму интервала необходимо применить

параметры режима, противоположного тому, в интервал которого выходит мощность.

Таблица Б.1

Заданные режимы работы реактора

Режим работы реактора	Интервал мощности, МВт	Положение КС, %						
		1	2	3	4	5	6	7
4. Максимальный	80…85	25	15	5	15	5	15	5
3. Нормальный	50…80	50	20	0	20	0	20	0
2. Ослабленный	20…50	75	25	25	25	25	25	25
1. Минимальный	5…20	100	50	25	50	25	50	25
0. Заглушен	0…5	100	100	100	100	100	100	100

Примеры:

1. В режиме 3 мощность возрастает до 82 МВт (соответствует режиму 4) – активируется режим 2.
2. В режиме 3 мощность падает до 30 МВт (соответствует режиму 2) – активируется режим 4.
3. В режиме 3 мощность падает до 15 МВт (соответствует режиму 1) – активируется режим 4, так как режима номером выше нет.
4. В режиме 2 мощность падает до 15 МВт (соответствует режиму 1) – активируется режим 3.
5. В режиме 2 мощность падает до 4 МВт (соответствует режиму 0) – активируется режим 4.

При работе в режиме 4 при превышении мощности на каждые 5 МВт должен активироваться следующий режим вплоть до глушения реактора при превышении порога 100 МВт.

Также необходимо предусмотреть аварийный останов реактора нажатием аварийной кнопки. При этом реактор переходит в режим 0.

2. Управление вентилятором. Имеется вентилятор, обеспечивающий проветривание комнаты. Мощность двигателя, а значит, и скорость вращения, определяется управляющим напряжением. Режим включения вентилятора должен соответствовать уровню температуры, которая измеряется

четырьмя датчиками, и значению влажности, которая определяется датчиком, дополненным компаратором, выдающим 0, если уровень влажности соответствует норме или 1 в противном случае. Диапазон температуры, измеряемый датчиками, составляет –20°С ÷ 60°С, диапазон выходного сигнала –1,5 ÷ 4,5В. Частота опроса датчиков соответственно равна 20 мс, 80 мс, 60 мс и 120 мс. Пока среднее значение температуры в помещении ниже 20°С – скорость вращения лопастей минимальна. При достижении среднего значения температуры в помещении 20°С вентилятор переключается во второй режим, когда срабатывает датчик влажности – в третий режим. Напряжения, соответствующее режимам, равны –2 В, 2.8 В и 4 В.

3. Производство лекарственных препаратов. Необходимо обеспечить непрерывную работу участка подготовки жидкого компонента для производства лекарственных препаратов. На участке имеется три емкости одинакового объема. Жидкость подается в первую емкость, в которой нагревается до температуры 490 °С. Затем жидкость поступает во вторую емкость, где она охлаждается естественным образом до 360 °С и поддерживается на этом уровне. По сигналу оператора жидкость поступает в третью емкость, где нагревается до 564 °С и выдерживается в течение 30 мин., после чего сливается.

В каждой емкости установлены 2 датчика температуры с диапазоном измерения 0…600 °С и диапазоном выходного сигнала 0…24 В. Емкости 1 и 3 предполагают 2 режима нагрева: интенсивный – подача напряжения 86 В; нормальный – подача напряжения 36 В. При разнице между заданной и фактической температурой более 20 °С задействуется интенсивный режим нагрева. Емкость 2 предполагает только нормальный режим нагрева.

Заполнение емкости 1 и перекачка между емкостями осуществляется насосами, которые управляются непрерывным напряжением в диапазоне 0…56 В. При включении напряжение нарастает равномерно от 0 В до 56 В в течение 3,5 с. При

выключении напряжение понижается равномерно от 56 В до 0 В в течение 2,8 с.

Каждая емкость оборудована выпускным клапаном, открытие которого управляется дискретным сигналом посредством ЭМР.

4. Расфасовка муки в мешки. Необходимо обеспечить непрерывную работу одного канала в системе расфасовки муки. В каждом канале имеется бункер для хранения муки и расфасовке ее в бумажные мешки по 15 и 30 кг. Наличие мешка в позиции определяется двумя ФЭ. ФЭ1 определяет факт наличия мешка, ФЭ2 – тип (высоту) мешка. Факт смены мешка фиксируется изменениями сигнала ФЭ1.

Бункер контролируется тензосистемой с диапазоном измерения от 100 до 9500 кгс и диапазоном выходного напряжения от 0 до 15 В. Бункер оснащен устройством блокирования загрузки, которое управляется дискретно. При достижении массы муки в бункере 8 т бункер блокируется через 2 с после фиксации этой массы.

Бункер оснащен дозатором с ЭМП, управляемым непрерывно подачей напряжения в диапазоне от 6 до 32 В. При напряжении ниже 6 В дозатор не работает. При напряжении 32 В дозатор работает с максимальной продуктивностью. Количество дозированной муки определяется на основе показаний тензосистемы. Величина управляющего воздействия на дозатор задана в табл. Б.2.

Таблица Б.2

Заданные управляющие воздействия на дозатор

Остаток до номинальной массы, кг	Более 5	1…5	0,4…1	0…0,4
Воздействие на дозатор, В	31,5	18,5	9,5	6,5

5. Нагрев жидкости. Имеется емкость, предназначенная для подогрева жидкости. При нагреве жидкость превращается в пар. Производится измерение давление с помощью датчика, дополненного компаратором. Когда давление превысит заданное компаратором значение, выдается 1. При этом необходимо

выключить подогрев и открыть клапан для того, чтобы выпустить пар. Когда давление уменьшится (на выходе компаратора 0), клапан закрыть и включить подогрев. Клапан приводится в движение двигателем в соответствии с графиком на рис. Б.2. Для измерения температуры в системе имеется 5 датчиков, измеряющих температуру в диапазоне –30°С ÷ 145°С, диапазон выходного сигнала –3 ÷ 9В. Необходимо опрашивать датчики в следующем режиме:

– сначала опрашивается только первый датчик до тех пор, пока измеряемая температура не достигнет 5°С;

– далее опрашивается второй до тех пор, пока измеряемая температура не станет выше 60°С;

– далее опрашиваются все остальные датчики в течение 5 мин.

Из полученных значений сформировать массивы.

Нагревательные элементы управляются одним дискретным сигналом.

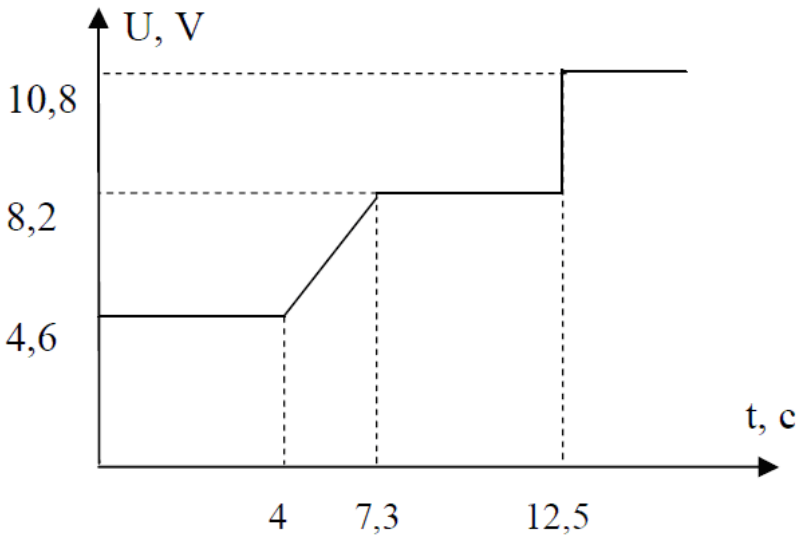


Рис. Б.2. Управляющее воздействие на клапан

6. Управление конвейерной линией. Линия приводится в движение двумя асинхронными ЭД, которые управляются дискретно посредством ЭМР. Необходимо организовать контроль провисания ленты. Для контроля используется аналоговый датчик с диапазоном входного воздействия 0…25 см провисания и соответствующим диапазоном выходного напряжения 0…6.5 В. Параметры управления представлены в табл. Б.3.

При провисании ленты более чем на 5 см необходимо выводить сообщение оператору и сигнализацию в соответствии с таблицей. При провисании ленты более чем на 15 см необходимо остановить конвейер. Звуковая и световая сигнализация управляется дискретно посредством оптического реле. Мигание осуществляется с частотой 2 Гц.

Таблица Б.3

Параметры управления конвейерной линией

Величина провисания, см	Тип сообщения оператору	Сигнал оповещения
5…10	Информационное	Нет
10…15	Предупреждающее	Непрерывный световой сигнал
>15	Аварийное	Мигающий световой сигнал + звуковой сигнал

7. Дозирование сыпучих продуктов. Имеется бункер с сыпучим продуктом, отгрузка которого осуществляется в ж/д вагоны. Позиция вагона для загрузки фиксируется двумя концевыми выключателями (КВ). При срабатывании обоих КВ считывать вместимость вагона и при наличии ненулевой вместимости начинать загрузку. При срабатывании одного из двух КВ формировать предупреждающее сообщение оператору.

Чтение и запись актуальной вместимости осуществляется в цифровом виде под управлением разрешающих сигналов чтения и записи. Если вагон заполнен до предела, то при записи выводится 0.

В систему может быть введена масса отгружаемого продукта – число в интервале о 0 до 60000. Если данная величина не введена, то отгрузка осуществляется до предела.

Дозирование из бункера осуществляется посредством задвижки, управляемой ЭМП. Контроль массы отгружаемой муки осуществляется четырьмя датчиками массы, включенными в мостовую схему, напряжение с выхода которой находится в диапазоне от 0 до 20 В и соответствует диапазону измерения 0…100 т.

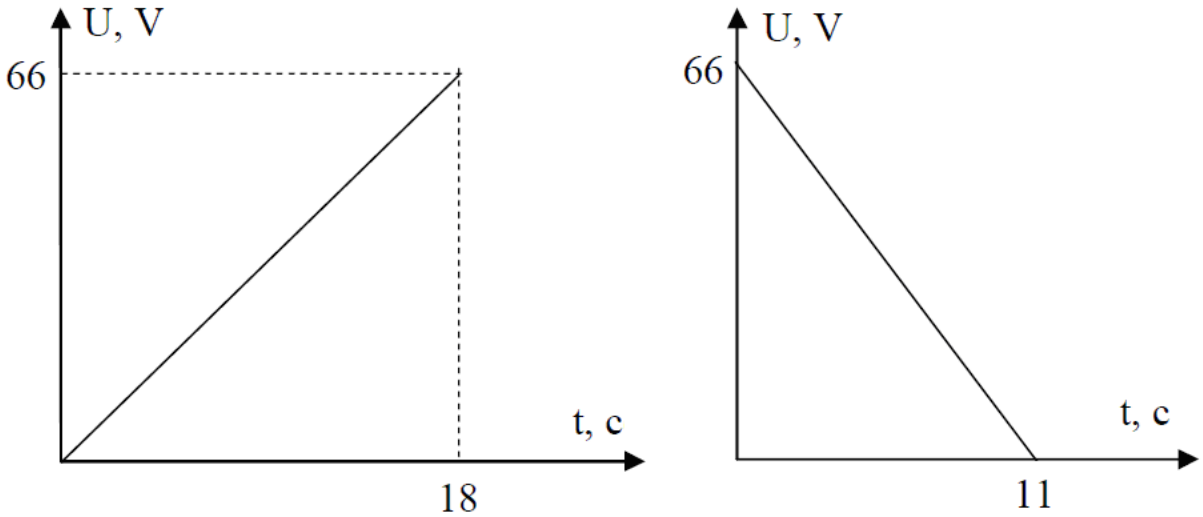
При отгрузке необходимой массы продукта необходимо выждать 50 мс и закрыть задвижку. В случае наполнения вагона до предела необходимо выждать 150 мс и закрыть задвижку.

8. Управление конвейером. Имеется конвейер, по которому движутся детали. Необходимо производить контроль размеров деталей. Размер детали определяются двумя дискретными датчиками. Первый выдает 1, если размер меньше нормального. Второй выдает 0, если размер больше нормального. В первом случае открывается заслонка корзины бракованных деталей посредством электромагнитного привода. Во втором случае включается поворотное устройство, изменяющее направление движения детали в цех обработки. Поворотное устройство управляется двигателем, на который необходимо подать напряжение 11,3 В в течение 36 с. Для установки в исходное положение на двигатель поворотного устройства необходимо подать напряжение –2,2 В. Следует предусмотреть время для отработки команд. У деталей, направляемых в цех обработки, производится измерение температуры с помощью 2 бесконтактных датчиков, измеряющих температуру в диапазоне –30°С ÷ 95°С, диапазон выходного сигнала –3 ÷ 9,5В. Интервалы опроса соответственно равны 0,1 с и 0,5 с. Датчики опрашиваются до тех пор, пока среднее значение измеряемой температуры не достигнет 60 °С.

9. Нагрев помещения. Имеется система, позволяющая включать обогрев помещения и регулирование температуры в помещении. Включение обогрева производится, если температура в помещении ниже заданного уровня. Контроль за температурой осуществляется датчиком, выдающим 0 или 1. Включение обогрева – подача напряжения на тэн согласно рис. Б.3, а, выключение согласно рис. Б.3, б.

В системе также производится измерение влажности с помощью 12 датчиков. Датчики опрашиваются в следующем режиме: датчики Д1, Д3, Д5 опрашиваются с интервалом 2 с, датчики Д7, Д9, Д11 с интервалом 4 с, датчики Д2, Д4, Д6 с интервалом 6 с и датчики Д8, Д10, Д12 с интервалом 8 с. Опрос

прекратить, когда в любой группе датчиков измеряемая суммарная влажность не превысит 80%. Датчики измеряют влажность в диапазоне от 10 до 100%, диапазон выходного сигнала 0,45 ÷ 9,2В.



а б

Рис. Б.3. Управляющее воздействие на обогреватель

10. Управление магистральным агрегатом нефтепровода. Транспортировка нефти по трубе обеспечивается магистральным агрегатом (МА), включающим в себя 2 бустерных насоса (основной и дополнительный), исполнительным механизмом которых выступает электродвигатель, управляемый дискретно.

Давление нефти в трубе на участке МА контролируется четырьмя аналоговыми датчиками (по 2 датчика до и после МА) с диапазоном измерения 0…20 атм и диапазоном выходного сигнала 0…320 мВ. Датчики опрашиваются поочередно с интервалом в 20 мс: сначала датчики до МА, затем через 0,4 с – после МА.

Если среднее давление в трубе до МА ниже 10,8 атм, то работает основной насос. При этом если разность среднего давления в трубе до МА и среднего давления в трубе после МА более 0,8 атм, то работает резервный насос.

11. Система предотвращения столкновения автомобиля. Имеется аналоговый датчик, фиксирующий

расстояние до препятствия с диапазоном измерения 0…30 м и соответствующим выходным диапазоном напряжения 0…600 мВ. В случае, если в течение 100 мс фиксируется сближение с препятствием более чем на 1,5 м, формируется сигнал «Опасное сближение» и выполняется торможение (табл. Б.4).

Таблица Б.4

Параметры формирования сигнала

Сближение за 100 мс, м	Тип сообщения водителю	Режим торможения
<1,5	Нет	Нет
1,5…2,5	Предупреждающее	Нет
2,5…3,5	Предупреждающее	Служебное торможение
>3,5	Аварийное	Экстренное торможение

Служебное торможение выполняется посредством выдачи управляющего напряжения, нарастающего от 0 до 9,8 В в течение 2,4 с. При экстренном торможении выдается аналогичное напряжение в течение 0,2 с.

12. Система пожаротушения. В помещении имеется дискретный датчик задымленности и 4 аналоговых датчика температуры. Датчик задымленности выдает 1 в случае повышения уровня дыма. Диапазон измерения параметра датчика температуры от 2°С до 145°С, диапазон выходного сигнала 0,05…3,5 В. Инерционность датчиков температуры равна 0,6 с. Также имеется дискретный звуковой сигнализатор пожара. Для гашения пожара в помещении есть 2 водоогнетушителя (распрыскиватели воды). Если среднее значение температуры превысило 83°С, необходимо сформировать дискретный сигнал пожара и включить распрыскиватели. Вода подается с помощью водонасоса, мощность которого меняется в зависимости от поданного на него напряжения. Для запуска водонасоса необходимо в течение 140 мс нарастить напряжение от 0 до 12 В. Если в течение 10 с после включения соответствующего режима температура не опускается ниже 70°С, то необходимо в течение 120 мс повысить напряжение до 24 В. Распрыскиватели

отключаются, когда температура опускается ниже 45°С. Для отключения водонасоса необходимо понизить напряжение от текущего значения до нуля в течение 360 мс. Сигнал пожара отключать при падении температуры ниже 45°С и сброса датчика задымленности.

13. Управление уровнем жидкого металла в кристаллизаторе машины непрерывного литься заготовок (МНЛЗ). Литье стальных заготовок осуществляется непрерывным способом посредством разливки стали из ковша параллельно по нескольким ручьям. На входе каждого ручья расположен кристаллизатор, который интенсивно охлаждается водой и обеспечивает первичное затвердевание слитка. Далее слиток охлаждается открытой струей, после чего режется на заготовки одинаковой длины.

Для нормальной работы системы уровень металла в кристаллизаторе должен поддерживаться на уровне 70% его наполнения. Для контроля уровня используется датчик уровня металла в кристаллизаторе (ДУМК), который установлен на отметке 100%. ДУМК фиксирует уровень радиоактивного излучения от изотопа, расположенного на нулевой отметке, и преобразует его в напряжение в диапазоне от 0 до 50 В. При этом отметке 0% соответствует максимальное напряжение, а отметке 100% соответствует напряжение 10,5 В.

При разливке металла открытой струей уровень металла в кристаллизаторе регулируется за счет изменения скорости извлечения слитка, которая задается вращением тянущего ролика. Ролик приводится в движение электродвигателем с редуктором. Для изменения частоты вращения ролика необходимо изменять напряжение на входе редуктора.

Алгоритм управления:

– начальное значение напряжения на управляющем входе редуктора 9,8 В;

– при повышении уровня металла на величину от 2 до 5% необходимо повышать напряжение на редукторе на 25 мВ каждые 0,2 с до начала возрастания уровня;

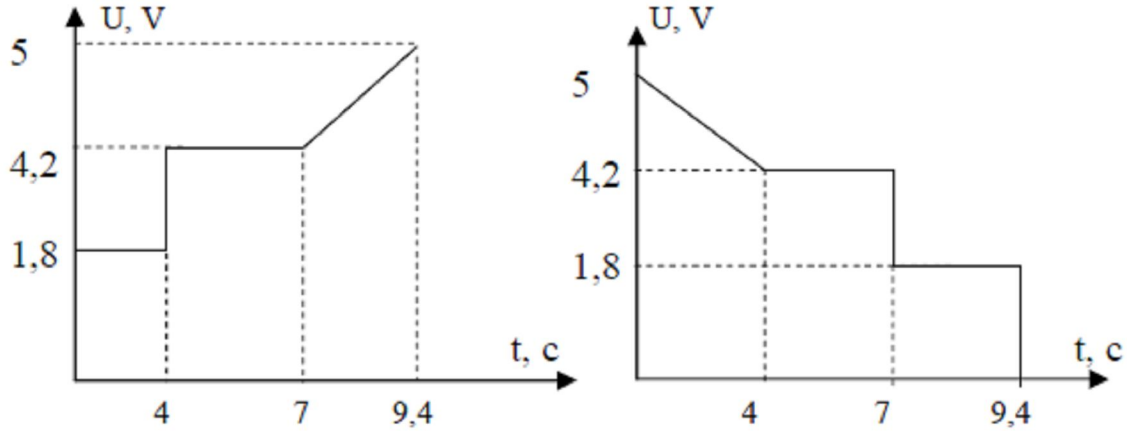
– при понижении уровня металла на величину от 2 до 5% необходимо понижать напряжение на редукторе на 25 мВ каждые 0,2 с до начала убывания уровня.

14. Система контроля физиологического состояния человека осуществляет контроль частоты сердечных сокращений и дыхания человека. Для контроля частоты сердечных сокращений используется дискретный датчик удара, который при каждом сердечном сокращении формирует треугольный импульс длительностью 50 нс. Для контроля дыхания используется 2 аналоговых датчика дыхательных движений (по одному на грудную клетку и переднюю брюшную стенку), которые формируют на выходе уровень напряжения в интервале от –700 до +700 мВ, пропорциональный глубине дыхания. Система должна предусматривать калибровку датчиков дыхательных движений по нижнему уровняю (полный выдох) и верхнему уровню (полный вдох).

Необходимо обеспечить подсчет частоты сердечных сокращений, частоты дыхательных движений и глубины дыхания человека. Подсчет ведется в течение 3 мин. При этом измерительные данные о глубине дыхания (максимум и минимум непрерывного сигнала) фиксируются в двух одномерных массивах. Частота сердечных сокращений вычисляется каждые 10 с. Если она превышает 100 ударов в минуту, то формируется звуковой сигнал длительностью 100 мс с частотой 1 Гц и амплитудой 1 ед. При превышении 120 ударов в минуту – с частотой 3 Гц и амплитудой 2 ед. При превышении 150 ударов в минуту – с частотой 5 Гц и амплитудой 3 ед. Амплитуда звукового сигнала определяется уровнем напряжения, которое подается на динамик при помощи 8-разрядных ЦАП с выходным диапазоном напряжения 0…50 мВ.

15. Управление стиральным агрегатом. Имеется стиральный агрегат. Датчик уровня фиксирует уровень воды, подаваемой с помощью насоса, который включается подачей напряжения 8,3 В. После заполнения водой включается двигатель. Вращение двигателя по часовой стрелке запускается

согласно графику на рис. Б.4, а. Выключение двигателя производится по графику на рис. Б.4, б. При движении против часовой стрелки напряжение меняет знак. Режим вращения: по часовой стрелке – 5 мин; против часовой стрелки – 3 мин.



а б

Рис. Б.4. Управляющее воздействие на двигатель

Процесс повторяется трижды. Температура воды измеряется восемью датчиками, измеряющими температуру в диапазоне –30°С ÷ 115°С, диапазон выходного сигнала –1,2 ÷ 4,35 В. Написать программу опроса датчиков Д1, Д3 с интервалом 30 мс, датчиков Д4 и Д5 с интервалом 50 мс, датчиков Д2, Д8 с интервалом 90 мс и датчиков Д6, Д7 с интервалом 100 мс. Опрос прекратить, когда максимальное значение, полученное от датчиков, превысит значение 95°С.

16. Предотвращение высокой концентрации метана в шахте посредством активизации режима управления вентилированием, обеспечивающего уровень концентрации пыли и метана.

На участке выработки угля имеется 12 аналоговых датчиков концентрации метана и 5 дискретных датчиков пыли. Каждый датчик измеряет концентрацию метана в интервале 0…42 % и выдает на выход эквивалентный аналоговый сигнал в диапазоне 0…8,5 В. Датчики метана опрашиваются поочередно с интервалом 0,25 с. Предусмотрены следующие режимы реагирования на показатели концентрации метана (табл. Б.5).

Интенсивность вентилирования регулируется управляющим напряжением следующим образом: 2,8 В – низкая; 5,4 В – средняя; 7,6 В – высокая; 9,1 В – очень высокая. Звуковое оповещение управляется дискретным сигналом.

Таблица Б.5

Режимы реагирования на показатели концентрации метана

№ режима	Концентрация метана, %		Концентрация пыли	Интенсив-ность вентилир-я	Звуковое оповеще-ние
	максимальное	среднее			
0	–	–	Нормальная	Низкая	Нет
1	–	–	Повышенная	Средняя	Нет
2	–	2,4	–	Средняя	Нет
3	–	3,6	Повышенная	Высокая	Да
4	4,4	–	–	Высокая	Да
5	4,4	–	Повышенная	Оч. высокая	Да

17. Управление микроклиматом в теплице. Производится контроль влажности земли в теплице. Если влаги достаточно, то поливочное устройство закрыто. В случае снижения влажности включаются насос для подачи воды и двигатель, вращающий поливочное устройство. Насос включается подачей на него напряжения 8 В. Скорость вращения поливочного устройства определяется напряжением, подаваемым на двигатель (рис. Б.5, а).

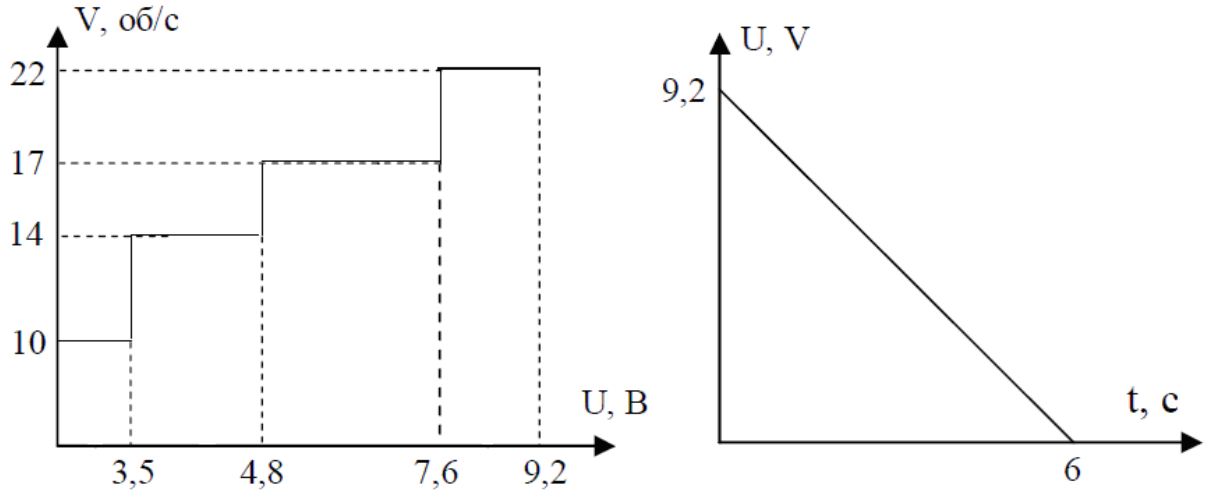


Рис. Б.5. Управляющее воздействие на привод поливочного устройства

Выключение двигателя поливочного устройства производится согласно рис. Б.5, б. Уровень влажности определяется датчиком, выдающим 0 или 1. В системе также производится измерение температуры воздуха 5 датчиками, измеряющими температуру в диапазоне –30°С ÷ 130°С, диапазон выходного сигнала –1 ÷ 4,7 В. Интервалы опроса соответственно равны 0,24 с; 0,12 с; 0,06 с; 0,12 с; 0,24 с.

Датчики опрашиваются в следующем режиме:

– сначала опрашивается только первый датчик до тех пор, пока измеряемая температура не достигнет 17°С;

– далее опрашиваются все остальные датчики в течение 3 мин. Из полученных значений сформировать массивы.

18. Контроль трансформатора. Имеется масляный трансформатор. Уровень масла фиксируется датчиком уровня. Для измерения температуры масла используются 5 датчиков, измеряющих температуру в диапазоне 10°С ÷ 120°С, диапазон выходного сигнала 0,4 ÷ 4,8В. Датчики опрашиваются в следующем режиме:

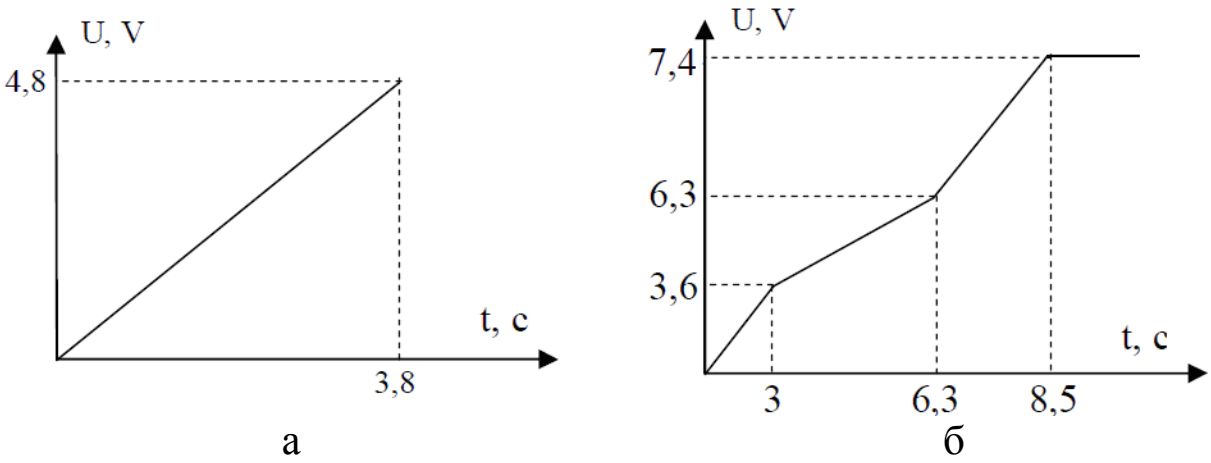
– первый опрашивается непрерывно до тех пор, пока температура не достигнет значения 78°С;

– затем опрашиваются попеременно второй и третий датчики в течение 15 с;

– затем попеременно первый и четвертый в течение 10 с.

Следует учитывать, что инерционность датчиков равна 0,2 с.

Если максимальная температура за период опроса превысит 93°С, необходимо включить двигатель охладителя, подав на него напряжение по схеме на рис. Б.6, а. При снижении уровня масла необходимо включить насос для добавления масла, выдав на него управляющее воздействие по схеме на рис. Б.6, б. Выключение двигателей осуществляется сбросом напряжения в ноль.



а б

Рис. Б.6. Управляющее воздействие на двигатель охладителя

19. Управление работой лифта 6-этажного здания. В системе имеется 6 внутренних кнопок для вызова нужного этажа в лифте. На каждом этаже имеются по 2 независимых концевых выключателя (КВ), определяющих позицию лифта на этаже, а также кнопка вызова лифта. Сигналы с обеих групп кнопок и обеих групп КВ кодируются четырьмя шифраторами CD 8×3.

Управление двигателем и дверями лифта осуществляется дискретно:

– работа на подъём/спуск – 0/1;
– включение/выключение двигателя – 0/1;
– открытие/закрытие двери лифта – 0/1.

Двери могут быть открыты, только если лифт не движется и на этаже (на входе присутствует 2 одинаковых сигнала с КВ). Если лифт не движется и сигналы с КВ разные, генерируется сигнал тревоги.

Следует реализовать программную временную задержку на 10 с после открытия двери. В программной задержке нужно предусмотреть возможность досрочного выхода из нее в случае нажатия внутренней кнопки лифта, отличной от номера этажа, и продлить задержку на 10 с в случае нажатия внутренней кнопки лифта, совпадающей с номером этажа.

В системе имеется аналоговый датчик натяжения троса лифта с диапазоном измерения 0…20 кН и диапазоном выходного напряжения 0…30 В. Фиксация силы натяжения менее 1 кН трактуется как обрыв троса. При фиксации обрыва троса

необходимо выдать дискретный сигнал на систему аварийного торможения лифта и подать сигнал тревоги.

20. Система очищения парафина. Процесс получения очищенного парафина заключается в том, что при нагреве технического парафина примеси плавятся быстрее. При нагреве парафина до температуры 68°С необходимо открыть клапан для слива примесей. Через 2 мин. клапан закрывается, нагрев продолжается до температуры 89°С. Открывается второй клапан для слива готового парафина. Измерение температуры производится 6 датчиками, измеряющими температуру в диапазоне от 25°С до 145°С, диапазон выходного сигнала 0,5… 4,5 В. Интервалы опроса датчиков соответственно равны 0,2 с; 0,15 с; 0,05 с; 0,1 с; 0,2 с; 0,15 с. Датчики опрашиваются в следующем режиме:

– сначала опрашивается только второй датчик до тех пор, пока измеряемая температура не достигнет 61°С;

– далее опрашиваются все остальные датчики, пока средняя температура не достигнет 68°С.

Кроме того, шестой датчик снабжен компаратором, который выдает импульс высокого уровня, когда температура превысит 89°С.

Открытие клапана производится выдачей на его двигатель напряжения 3,7 В. Закрытие клапана производится выдачей на его двигатель напряжения –4,6 В.

21. Система контроля корректности электропитания. Имеется электрическая цепь, питающая электросталеплавильную печь переменным током номиналом 2000 А с допуском ±5 А. Сила электрического тока в проводнике измеряется бесконтактным способом при помощи индуктивного датчика, в обмотках которого при помещении его в заданную позицию относительно проводника возникает ЭДС, пропорциональная силе тока. Диапазон измерения датчика тока: 0…4000 А. Диапазон выходного сигнала датчика тока: 0…30 В.

Задача системы – отслеживать величину тока при работающей установке. При выходе силы тока за пределы зоны допуска не более чем на 2 А необходимо зажигать одну из сигнальных ламп «Недопустимое повышенные/понижение тока». При выходе силы тока за пределы зоны допуска более чем на 2 А необходимо зажигать одну из сигнальных ламп «Критическое повышенные/понижение тока» и включать сирену. Включение сирены осуществляется посредством подачи на динамик непрерывного сигнала согласно графику на рис. Б.7. Если по истечении 50 сек. сила тока по-прежнему находится в критическом значении, то запускать сирену снова.

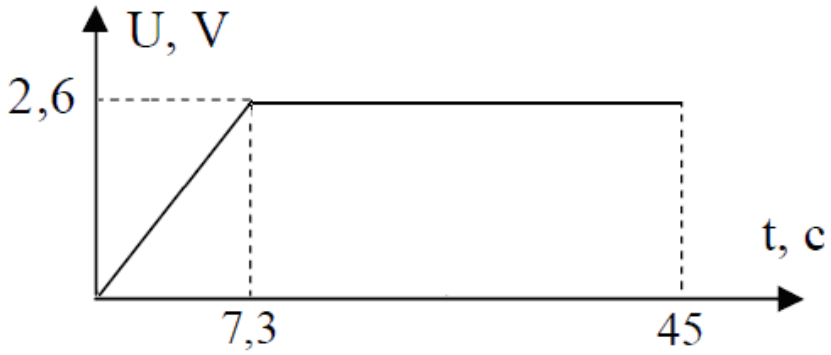


Рис. Б.7. Управляющее воздействие на динамик

22. Система контроля критических параметров авиационного двигателя (АД). Осуществляется контроль критической температуры на внутренних стенках корпуса АД. 11 дискретных датчиков температуры выдают сигнал «1» при превышении температуры в некоторой точке корпуса. О достижении критической температуры сигнализируют 5 сигнальных ламп по схеме индикации в табл. Б.6.

Таблица Б.6

Схема индикации состояния датчиков температуры АД

№ датчика	1	2	3	4	5	6	7	8	9	10	11
Лампа 1	1	1								1	1
Лампа 2			1	1	1	1					
Лампа 3					1	1			1	1	1
Лампа 4				1	1	1	1	1			1
Лампа 5	1						1	1	1		1

Также осуществляется контроль вибрации корпуса АД четырьмя аналоговыми датчиками. Каждый датчик измеряет вибрацию в диапазоне 0…1100 мм/с, а диапазон выходного напряжения составляет 0…250 мВ.

Устройство оповещения о недопустимом уровне вибрации реализовано посредством 4-х сигнальных ламп и непрерывного звукового устройства, и управляется согласно табл. Б.7. Если уровень вибрации по обеим группам датчиков выше допустимого, непрерывное управление осуществляется максимальным уровнем напряжения.

Таблица Б.7

Параметры управления устройством оповещения

№ датчика вибрации	Уровень (скорость) вибрации, мм/с	Номер лампы	Напряжение, В
1 или/и 2	200…450	6	15,6
1 или/и 2	>450	7	19,2
3 или/и 4	400…810	8	19,2
3 или/и 4	>810	9	22,8

23. Контроль движения в лаборатории. В помещении лаборатории позиционно отслеживается движение животных. Для этого площадь лаборатории разделена на 8 секторов: 2 ряда по 4 сектора. За состоянием секторов ведется контроль при помощи 8 дискретных датчиков движения и двух видеокамер. Каждая видеокамера включается/выключается дискретным сигналом и позиционируется при помощи четырехфазного шагового двигателя. Камера перемещается вокруг вертикальной оси межу углами 30° и 150° относительно плоскости стены с шагом 2° влево и вправо. Камера 1 расположена на границе секторов 2 и 3. Камера 2 расположена на границе секторов 6 и 7. Исходная позиция камеры в режиме ожидания – 90°, камера выключена. Задача состоит в позиционировании видеокамер для наиболее полного обзора движения. Если движение зафиксировано в одном секторе, обе камеры незамедлительно включаются и позиционируются на этот сектор. Если движение

зафиксировано в нескольких секторах, камеры позиционируются на промежуточной позиции между активными секторами.

Схема движения шагового двигателя показана на рис. Б.1 (слева движение к отметке 30°, справа – к отметке 150°). Двигатель управляется наборами дискретных сигналов, следующих с частотой 10 Гц. При этом один период фазы №1 соответствует шагу 2°. При отсутствии сигналов от всех датчиков движения система устанавливает обе видеокамеры в исходную позицию и переходит в режим ожидания.

24. Электрокардиостимулятор. Имеется устройство для стимулирования работы сердечной мышцы посредством вывода стимулирующей последовательности импульсов (СПИ) правильной прямоугольной формы. Устройство оснащено аналоговым датчиком пульса, который формирует на выходе уровень напряжения, пропорциональный частоте сокращений сердечной мышцы. Диапазон выходного сигнала датчика: от 0 до 15 В, что соответствует интервалу от 0 до 180 мин$^{-1}$.

Также устройство оснащено тремя дискретными переключателями:

1) 3-позиционный переключатель режима работы;
2) 14-позиционный переключатель частоты СПИ;
3) 11-позиционный переключатель амплитуды СПИ.

Переключателем режима работы предусмотрено 3 режима:

1. Устройство выключено
2. Стимуляция без учета показаний пульса
3. Стимуляция с учетом показаний пульса

В режиме 1 устройство не выводит стимулирующий сигнал и ожидает переключения в другой режим.

В режиме 2 устройство формирует СПИ на основании установленных значений на переключателях 2 и 3.

В режиме 3 устройство формирует СПИ на основании установленных значений на переключателях 2 и 3 с коррекцией по фактическим показаниям датчика пульса по формуле:

$$\nu_{СПИ} = \frac{\nu_{установленная} + 2 \cdot \nu_{пульса\ факт.}}{3}, \text{ мин}^{-1}.$$

Заданный массив V(14) частот СПИ, устанавливаемых посредством переключателя 2, мин$^{-1}$: {40, 50, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 120}.

Заданный массив A(11) амплитуд СПИ, устанавливаемых посредством переключателя 3, мВ: {15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 45}.

Вывод СПИ осуществляется в режиме работы 2 или 3 через электрод, на который выдается аналоговый сигнал с амплитудой, установленной на переключателе 3.

25. Контроль скорости транспортного потока. Объект контроля – участок проезжей части с тремя полосами для движения в одном направлении. Скоростной режим движения автомобильного транспорта на участке контролируется тремя ультразвуковыми датчиками скорости транспорта (ДСТ) (по 1 датчику на каждую полосу). Диапазон измерения ДСТ: 0…100 м/с. Выходное напряжение ДСТ прямо пропорционально измеряемой скорости и находится в диапазоне 0…18 В.

Необходимо вести опрос датчиков в течение 10 сек. с интервалом в 1 сек. Если все 10 значений по полосе нулевые, то принять скорость по полосе 0 м/с. Если из 10 значений по полосе есть хотя бы одно ненулевое, то рассчитывать среднюю скорость по полосе только по ненулевым значениям.

По полученным ненулевым значениям средней скорости по полосам рассчитать среднюю скорость потока в км/ч и вывести на трёхразрядную панель индикаторов в двоично-десятичном коде. Если по всем полосам средняя скорость равна нулю, то панель индикаторов погасить.

При средней скорости потока более 60 км/ч зажигать информационное табло «Скорость потока превышает максимальную!».

При средней скорости потока более 80 км/ч циклически зажигать и гасить информационное табло «Скорость потока недопустимо высокая!» с частотой 1 Гц.

Управление информационными табло дискретное (1 бит на каждое табло).

26. Управление процедурой донации плазмы крови. Объект управления – сепаратор плазмы крови, предназначенный для забора крови из вены донора, сепарирования плазмы и возврата остаточного материала обратно в вену донора. Процедура выполняется тремя циклами, в каждый из которых сепарируется по 200 мл плазмы.

При подаче питания сепаратор входит в режим ожидания запуска процесса и в это время может принимать пользовательские настройки. Запуск процедуры осуществляется нажатием кнопки «Пуск». При этом начинается первый цикл процедуры. Также имеется кнопка «Стоп» аварийной остановки процесса.

Сепаратор снабжен счетчиком измерения объема сепарированной плазмы (СОП), который при достижении заданного объема выдает дискретный сигнал высокого уровня и сбрасывается в ноль.

Сепаратор снабжен насосом, который управляется непрерывным электрическим сигналом уровня ±18 В и может работать в двух направлениях: на забор и на возврат материала. Предусмотрено 3 режима работы насоса (табл. Б.8). Каждому режиму на панели сепаратора соответствует светодиодный индикатор. Датчик давления крови перед входом в насос измеряет давление в диапазоне от 0 до 33 мм рт. ст. с выдачей пропорционального выходного напряжения в диапазоне 0…4,8 В. В табл. Б.8 указаны диапазоны давления крови, при попадании в который должен активироваться указанный режим. В интервалах [2,0, 3,5] и [5,1, 8,4] сохраняется текущий режим работы насоса.

Таблица Б.8

Режимы работы насоса сепаратора плазмы

№	Направление работы	Давление крови в вене, мм рт. ст.	Номинальное напряжение, В	Звуковое оповещение при переходе в режим
0	останов	<2,0	0	4 Гц в течение 1 с
1	забор	3,5…5,1	9,8	2 Гц в течение 0,5 с
2	забор	>8,4	17,2	40 Гц в течение 0,25 с
3	возврат	–	–17,2	12 Гц в течение 0,25 с

В начале каждого цикла устанавливается режим 1. Спустя 3 секунды начинается непрерывный контроль давления крови в вене и управление режимом работы насоса, включая звуковую сигнализацию при смене режимов (см. табл. Б.8). Забор продолжается до поступления сигнала от СОП. После этого насос переводится в режим 0 и спустя 12 секунд – в режим 3. Возврат продолжается до тех пор, пока датчик давления фиксирует ненулевое давление на выходе насоса. Давление близкое к нулю означает отсутствие материала на выходе насоса. По истечении 6 секунд насос переводится в режим 0 и через 3 секунды начинается следующий цикл. По окончанию третьего цикла выдается звуковой сигнал 4 Гц в течение 3 с.

27. Управление системой шахтного водоотлива. Объект управления – участок водоотливного оборудования угольной шахты закрытого типа. На участке расположено 9 одинаковых дискретных датчиков уровня воды: 3 датчика на уровне ординара, 3 датчика на уровне 1 м выше уровня ординара, 3 датчика на уровне 1,5 м выше уровня ординара. Уровень считается затопленным, если любой их датчиков этого уровня выдает сигнал уровня логического ноля.

Также на участке имеется 4 одинаковых насосных установки (НУ), каждая из которых приводится в действие асинхронным электродвигателем переменного тока при помощи электромагнитного реле (ЭМР). ЭМР коммутирует на двигатель трехфазное напряжение 380 В и при срабатывании выдает дискретный сигнал уровня логической единицы. Кроме того, каждая установка оснащена двумя датчиками скорости потока воды с рабочим диапазоном измерения 0…10 м/с и диапазоном выходного напряжения 0…16 В.

При фиксации воды на уровне ординара работает 1 НУ, на уровне 1 м выше уровня ординара работает 2 НУ, на уровне 1,5 м выше уровня ординара работает 3 НУ.

Если ЭМР работающей НУ выдает сигнал уровня логического ноля, на пульт оператора передается сигнал Fault Relay # и запускается любая свободная НУ.

Для каждой работающей НУ вычисляется минимальное значение скорости потока воды. Если это значение ниже 2,2 м/с, на пульт оператора передается сигнал Fault Pump # и запускается любая свободная НУ.

Во всех случаях: если для включения нет свободной исправной НУ, на пульт оператора передается сигнал об аварии на объекте. Все сигналы оператору сопровождаются звуковым сигналом частотой 8 Гц и завершаются после нажатия кнопки подтверждения получения сигнала.

28. Нормирование давления воды в резервуаре. Имеется резервуар объемом 100 м^3 для буферного хранения воды, используемой для охлаждения изготавливаемой металлопродукции. Для нормального функционирования системы охлаждения давление в баке должно поддерживаться в пределах 330–360 кПа. Вследствие неравномерности расходования воды и нестабильного давления в системе подпитки могут возникать непредсказуемые выходы фактического значения давления в баке за пределы указанного интервала. В этом случае необходимо незамедлительно нормализовать давление в баке, которое контролируется двумя одинаковыми датчиками давления. Эти датчики с диапазоном измерения 0…1 МПа выдают на выходе эквивалентное напряжение в диапазоне от 0 до 24 В. При измерении давления в баке используется среднее значение показаний обоих датчиков.

Для повышения давления в баке используется бустерный насос, который может работать в режимах 1) низкой, 2) средней и 3) высокой интенсивности. Соответствующий режим активируется подачей на насос двоичного кода режима (выключенное состояние – режим 0). Схема управления насосом:

- режим 0 при значении давления не ниже 333 кПа;
- режим 1 при значении давления ниже 333 кПа;
- режим 2 при значении давления ниже 326 кПа;
- режим 3 при значении давления ниже 318 кПа.

При падении давления в баке ниже 310 кПа в течение более чем 1 секунды, выдается аварийный сигнал на пульт оператора об недопустимо низком уровне давления в баке.

Для понижения давления в баке служит сбросной клапан, который управляется постоянным током в диапазоне 4–20 мА. Току 4 мА и ниже соответствует полностью закрытое состояние клапана. Току 20 мА и выше соответствует степень открытия 100%. В интервале 4–20 мА степень открытия клапана линейна. Степень открытия сбросного клапана для управления давлением в баке:

– 5% при давлении в баке выше 360 кПа;
– 10% при давлении в баке выше 365 кПа;
– 25% при давлении в баке выше 370 кПа;
– 50% при давлении в баке выше 375 кПа;
– 100% при давлении в баке выше 380 кПа;

При раскрытии сбросного клапана на 100% в течение более чем 1 секунды, выдается аварийный сигнал на пульт оператора об недопустимо высоком уровне давления в баке системы охлаждения.

29. Система обмена мгновенными сообщениями предприятия. Система посредством центральной УВМ обеспечивает рассылку сообщений по каналам связи между 12-ю узлам в режиме реального времени. Все узлы сети постоянно находятся в режиме ожидания сообщений. Узел, инициирующий передачу (источник), при отправке сообщения в УВМ переходит в режим ожидания подтверждений от остальных узлов принимающей стороны (приёмников) и находится в этом режиме 300 мс, после чего переходит в режим ожидания сообщений.

УВМ обеспечивает ретрансляцию сообщения всем узлам кроме источника. Каждый узел-приемник при получении сообщения мгновенно передает в УВМ сигнал о получении. Максимальное время ретрансляции сообщения составляет 25 мс.

УВМ сразу после рассылки сообщения формирует массив состояний из 12 чисел формата байт. В элемент, соответствующий источнику, записывается 0, в остальные FF. Далее УВМ в течение 100 мс сканирует порт ввода сигналов о получении (интервал дискретизации при сканировании – 1 мс). В процессе сканирования УВМ фиксирует сигналы от приёмников,

формирует 16-разрядное слово флагов доставки и записывает в массив состояний время ожидания подтверждения от каждого узла в миллисекундах. По завершению сканирования УВМ отправляет слово флагов доставки через порт вывода №1.

Также УВМ осуществляет последовательную передачу элементов массива состояний через порт вывода №2 на ЦАП, который своим выходом соединен с блоком индикации. Интервал между выдачей слов состояния равен максимальному времени преобразования ЦАП (см. табл. 5.1). Синхронно с передачей элементов массива состояний УВМ выдает на блок индикации двоичный эквивалент индексов элементов. Блок индикации включает 12 ламп накаливания номинальным напряжением 36 В. Лампы с максимальной яркостью горения соответствуют приёмникам, не подтвердившим получение сообщения.

По завершению передачи массива состояний УВМ переходит в режим ожидания нового сообщения.

30. Управление автоматическими воротами. Объект управляется электроприводом, на который подается постоянное напряжение уровня 116 В с переходным процессом по включению (а) и выключению (б) согласно рис. Б.8. Сигналы открытия и закрытия ворот поступают от управляющих кнопок в ядро СРВ и непосредственно на электропривод ворот, тем самым определяя его режим работы: открытия и закрытия.

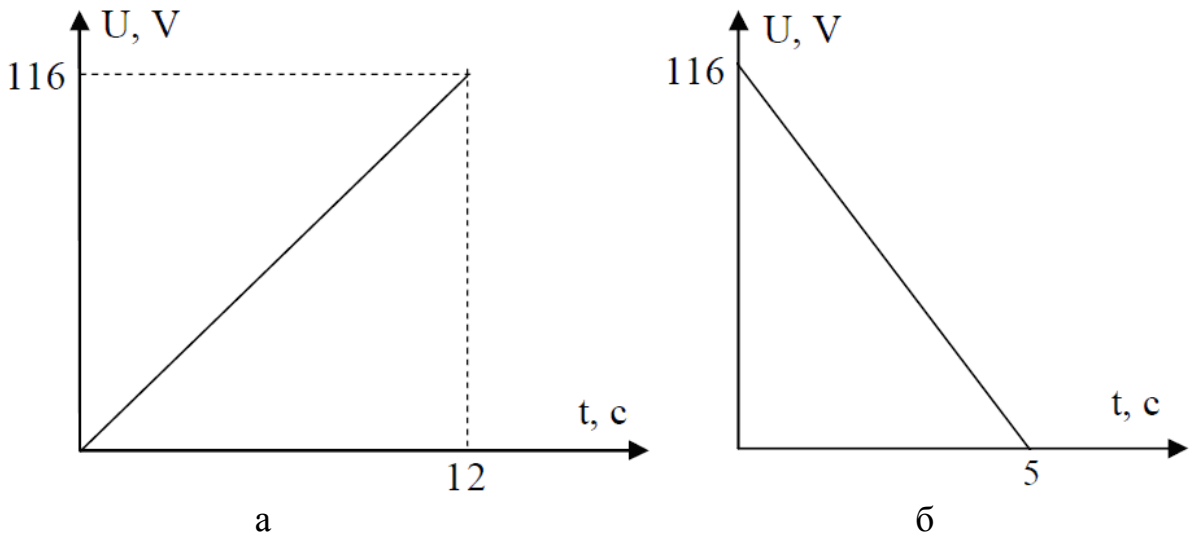


Рис. Б.8. Управляющее воздействие на электропривод ворот

Рабочее пространство движения створок ворот ограничено двумя парами концевых выключателей (КВ), которые фиксируют крайние положения створок в открытом и закрытом состояниях. Также в рабочем пространстве осуществляется контроль присутствия помех. Для этого используется два ультразвуковых датчика дискретного типа. Кроме того объект оснащен тензометрическим устройством, фиксирующим механическое напряжение исполнительного механизма, возникающее в результате его деформации при соприкосновении с препятствием. Датчик фиксирует механическое напряжение в диапазоне от 0 до 120 кгс/мм^2 и выдает на аналоговый выход электрическое напряжение в диапазоне от 0 до 15 В.

Активная фаза работы СРВ протекает в период движения створок ворот в рабочем пространстве. При нажатии на кнопку открытия ворот на электропривод в переходной период подается напряжение согласно графику на рис. Б.8-а и далее поддерживается на установившемся уровне. При срабатывании первого из двух КВ положения «Открыто» движение створок замедляется до 50% согласно графику на рис. Б.8-б, а при срабатывании второго КВ – далее снижается до нуля.

В процессе движения могут произойти следующие аварийные ситуации (табл. Б.9). Причем в ситуации №2 режим замедления обратим, и в случае падения напряжения ниже уровня 55 кгс/мм^2 рабочая скорость движения возобновляется согласно графику на рис. Б.8-а.

Таблица Б.9

Аварийные ситуации в активной фазе управления воротами

№	Аварийная ситуация	Реакция СРВ
1	Любой из ультразвуковых датчиков фиксирует посторонний объект	Ворота останавливаются*
2	Датчик напряжения фиксирует достижение предела пропорциональности (70 кгс/мм^2)	Ворота замедляются* до 30%
3	Датчик напряжения фиксирует достижение 95% предела упругости (95 кгс/мм^2)	Ворота останавливаются*

* Замедление движения выполняется по графику на рис. Б.8-б

В ситуациях 1 и 3, приводящих к полному останову, ворота могут быть только перезапущены оператором нажатием на одну из двух управляющих кнопок.

При нажатии на кнопку закрытия ворот управление происходит аналогично по описанному выше алгоритму с тем лишь отличием, что для контроля завершения закрытия используется другая пара КВ положения «Закрыто».

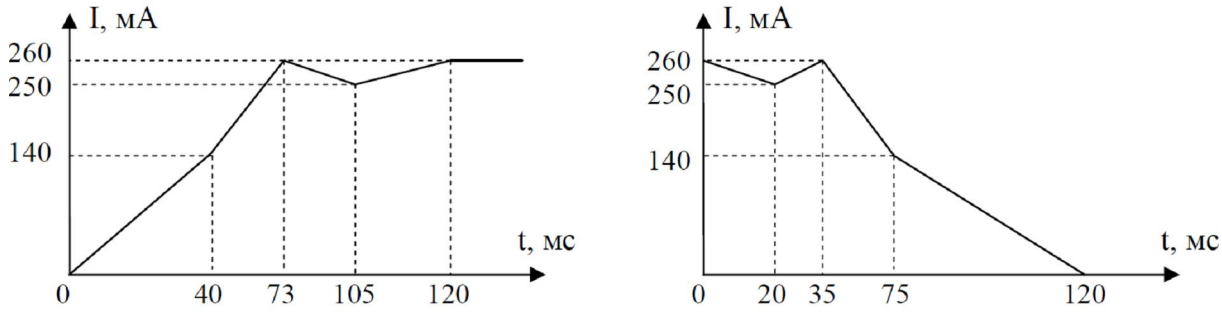
31. Управление газорезкой МНЛЗ. Объект управления – газорежущее устройство, предназначенное для отрезания заготовки определённой длины при ручьевой разливке стали на МНЛЗ (см. вариант 13, 1-й абзац). При выходе из системы охлаждения непрерывный слиток необходимо разрезать на заготовки заданной длины. Учет длин отрезаемых заготовок осуществляет внешняя подсистема расчета длины заготовки (ПРДЗ).

На резак посредством дискретно управляемых клапанов подается газ и кислород. Резак перемещается поперек ручья под управлением дискретного электроприводного устройства (предусмотрено два дискретных управляющих сигнала для движения резака вперед и назад).

Резак монтируется на тележке, которая может перемещаться вдоль ручья в направлении исходной позиции за счет работы дискретно управляемого электропривода. В исходной позиции тележки и в позиции максимального удаления расположены концевые выключатели КВ0 и КВ1. Перемещение тележки в исходную позицию завершается при срабатывании КВ0.

Также на тележке смонтировано механическое устройство захвата, которое обеспечивает фиксацию резака на слитке. Таким образом реализуется перемещение тележки вперёд для резки слитка. Захват приводится в действие соленоидным клапаном, который управляется постоянным током согласно графикам, показанным на рис. Б.9.

На пульте оператора имеется главная аварийная кнопка, по нажатию на которую работа объекта полностью блокируется: резак выключается, захват отпускается.



а б

Рис. Б.9. Управляющее воздействие соленоидный клапан захвата заготовки (а – захват, б – отпускание)

Исходное состояние объекта: захват отпущен, клапаны закрыты, резак и тележка в исходных позициях.

Цикл резки начинается, когда от ПРДЗ поступает разрешающий сигнал. Разрешением для резки также является выполнение следующих условий:

- главная аварийная кнопка не нажата;
- давление газа и кислорода выше минимального.

После включения захвата включается резак и начинает движение поперек ручья. Время резки T определяется по формуле:

$$T = \frac{L + 30}{V_C}$$

где L – заданная длина стороны квадратного профиля или диаметр круглого профиля заготовки, $V_C = 25\ мм/с$ – скорость движения резака.

За 1,5 с до конца резки выдается разрешение на запуск рольганга для транспортировки заготовки на перекладыватель. По окончанию резки объект возвращается в исходное состояние.

Если в процессе резки тележка доезжает до позиции максимального удаления, что фиксируется срабатыванием КВ1, то процесс прерывается и объект возвращается в исходное состояние. Загорается лампочка «Заготовка не отрезана».

Если в процессе резки дискретные датчики давления фиксируют падение давления газа или кислорода в процессе резки, то процесс прерывается и объект возвращается в исходное состояние. Загорается лампочка «Заготовка не отрезана».

32. Система аварийного оповещения о наводнении.

Объект управления – участок контроля аварийного подъема воды, который находится в пойме реки выше по течению относительно крупного города.

На участке расположено 6 одинаковых дискретных датчиков уровня воды: 3 датчика на 1 м выше уровня ординара и 3 датчика на уровне 3 м выше уровня ординара. Уровень считается затопленным, если любой их датчиков этого уровня выдает сигнал уровня логической единицы.

Если система фиксирует подъем воды до 1 метра выше уровня ординара, выдается дискретный сигнал на оранжевую сигнальную лампу. Если вода опускается ниже отметки «1 метр выше уровня ординара», оранжевая сигнальная лампа гаснет.

Если система фиксирует подъем воды до 3 метра выше уровня ординара, выдается дискретный сигнал на красную сигнальную лампу и аналоговый сигнал в виде постоянного тока на гудок согласно графику на рис. Б.10. Сигналы на гудок и красную сигнальную лампу перестают подаваться, когда вода опускается ниже отметки «3 метра выше уровня ординара».

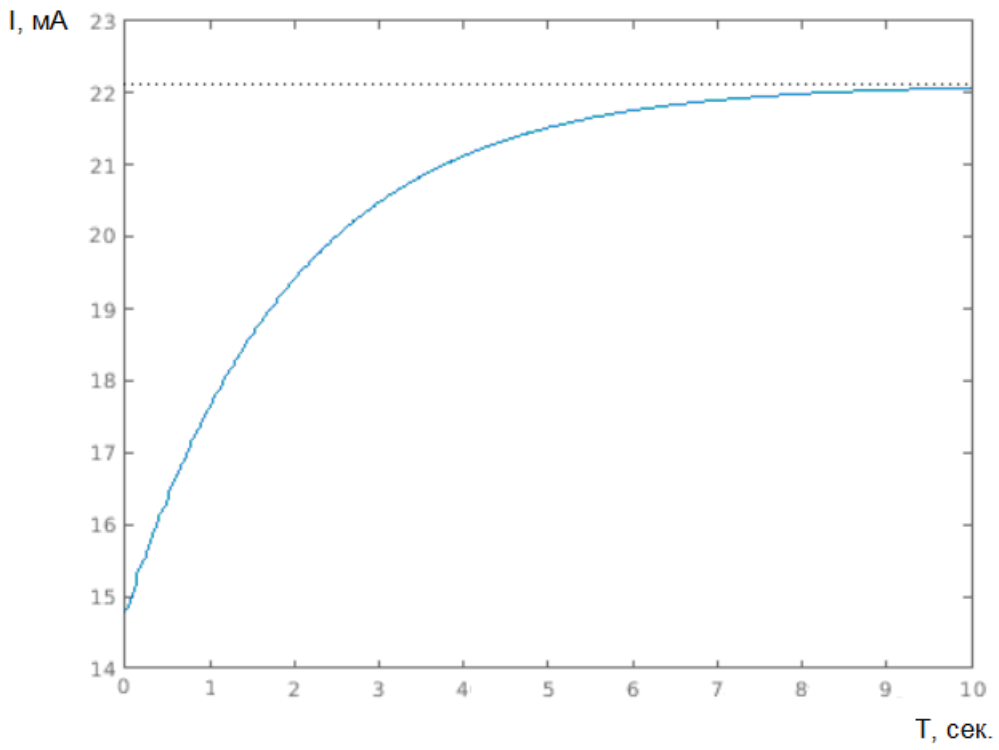


Рис. Б.10. График выдачи сигнала на устройство звукового оповещения

33. Контроль концентрации аммиака на территории производственного объекта. Ведётся наблюдение за концентрацией аммиака на территории площадью 1,2 кв. км. Территория разделена на 8 секторов, каждый из которых поделен на 8 подсекторов. В каждом подсекторе расположено пол 1 дискретному датчику минимально допустимой концентрации аммиака в воздухе. Сигналы с датчиков сегмента мультиплексируются (время задержки данных на мультиплексоре 8x1 составляет 2 мс). Опрос датчиков каждого сегмента происходит по одному с интервалом в 50 мс. Результаты опроса датчиков отображаются на светодиодной матрице размером 8x8. Если датчик сработал – горит соответствующий светодиод. Если датчики срабатывают не более чем в трех секторах, на динамик выдается последовательность импульсов уровня 9,5 В с частотой 1 Гц и скважностью 0,3. Если аммиак фиксируется в четырех и более секторах, на динамик выдается последовательность импульсов уровня 16 В с частотой 8 Гц и скважностью 0,5.

34. Система контроля концентрации вируса. Объект управления – закрытое общественное помещение, в котором ведется контроль вирусной нагрузки, для чего используется 4 специализированных анализатора. Каждый анализатор измеряет концентрацию вируса в диапазоне 0,45…32,1 × 10^5 МЕ/$м^3$. на выходе анализатора формируется пропорциональное напряжение в диапазоне 0…9,8 В.

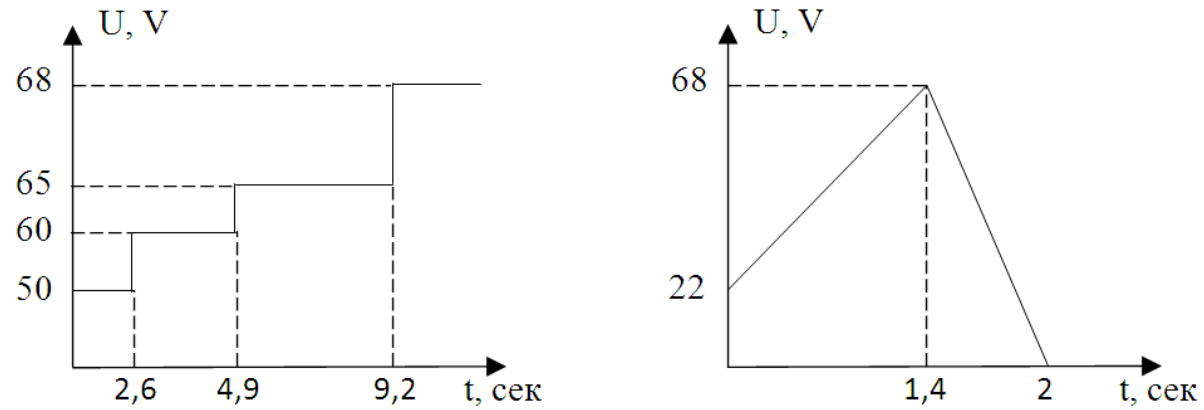
Помещение оснащено дискретно управляемой системой вентиляции, которая может работать в двух режимах: нормальном и интенсивном. Также предусмотрена сигнализация аварийно высокой концентрации вируса. Выбор режима работы объекта осуществляется в соответствии с табл. Б.10.

Таблица Б.10

Аварийные ситуации в активной фазе управления воротами

Вирусная нагрузка, × 10^5 МЕ/$м^3$	Режим работы системы вентиляции	Аварийная сигнализация
ниже 1,21	не работает	не работает
от 1,21 до 8,7	нормальный	не работает
выше 8,7	интенсивный	работает

35. Процесс разогрева специализированного сверхточного электропаяльника представлен временно́й диаграммой на рис. Б.11,а. Обеспечить нагрев паяльника до заданной температуры 90°С и поддержание ее в течении 10 с, для чего необходимо каждые 2 с подавать напряжение по схеме на рис. Б.11,б.



а) диаграмма разогрева б) диаграмма подогрева

Рис. Б.11. График выдачи управления на электропаяльник

Для измерения температуры паяльника используются 5 датчиков, измеряющие температуру в диапазоне 5 – 195 °С, диапазон выходного сигнала 0,01 – 4,85 В. Временна́я диаграмма опроса датчиков приведена на рис. Б.12. Процесс нагрева начинается после нажатия кнопки «Запуск».

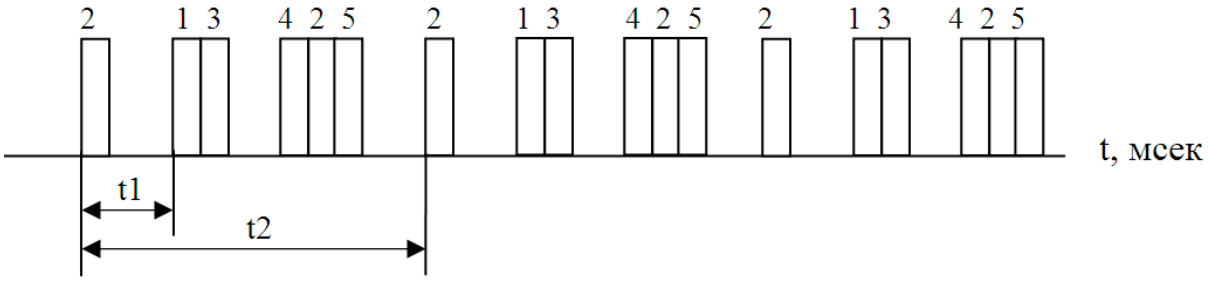


Рис. Б.11. Диаграмма опроса датчиков температуры электропаяльника

36. Электропила для распила дерева контролируется датчиком усилия, который выдает напряжение пропорциональное давлению, оказываемому на пилу. К датчику подсоединен компаратор напряжения, выдающий 5 В после превышения

давления некоторого порога. Изначально пила вращается со скоростью 5 об./с., что соответствует 10 В, которые подаются на двигатель. По сигналу, поступающему от компаратора необходимо:

1. Измерить ширину листа с помощью специальной измерительной системы.

2. В соответствии с шириной листа (1 м, 1.45 м, 2.2 м) на электромотор пилы подается напряжение 60, 80, 100 В, что соответствует 30, 40, 50 об./с соответственно.

Измерительная система представляет собой набор лазерных датчиков, вмонтированных в специальный прочный корпус, защищающий их от пыли. Датчики устанавливаются на штангу, которая прикреплена к полу. При подаче сигнала на измерительную систему, формируется байтовый цифровой код, соответствующий ширине листа в миллиметрах, и подаёт сигнал готовности. Разрядность кода 12 бит, диапазон измерения – от 0 до 4096 мм.

После распила дерева необходимо уменьшить обороты электропилы до первоначальных оборотов (5 об./с). Для определения напряжения, подаваемого на пилу, за основу берется среднее значение ширины листа. Для измерения давления используются 7 датчиков, измеряющие параметр в диапазоне 5 – 135 ед. изм., диапазон выходного сигнала 0,15 – 4,05В. Интервалы опроса соответственно равны 0,2 с; 0,15 с; 0,05 с; 0,1 с; 0,2 с; 0,15; 0,05 с. Датчики опрашиваются в следующем режиме: сначала опрашиваются только первый и четвертый датчик до тех пор, пока максимальное из этих двух значений не достигнет 75 ед. изм.; далее опрашиваются все остальные датчики в течении 5 мин.

СОДЕРЖАНИЕ

Учебное издание

П О Л Е Т А Й К И Н Алексей Николаевич

СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Учебное пособие

Подписано 02.03.2020. Формат 60×84 1/16.
Печать цифровая. Уч. изд. л. 5,0.
Тираж 100 экз. Заказ №

Кубанский государственный университет.
350040, г. Краснодар, ул. Ставропольская, 149.

Издательско-полиграфический центр
Кубанского государственного университета.
350040, г. Краснодар, ул. Ставропольская, 149.