СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

SIBERIAN FEDERAL UNIVERSITY

Электронный учебно-методический комплекс

Микропроцессорные системы

Учебная программа дисциплины

● Учебное пособие

Лабораторный практикум

Методические указания по самостоятельной работе

Банк тестовых заданий в системе UniTest

Красноярск
ИПК СФУ
2009

УДК 621.396.6(075)
ББК 32.844.1я73
М59

Электронный учебно-методический комплекс по дисциплине «Микропроцессорные системы» подготовлен в рамках реализации Программы развития федерального государственного образовательного учреждения высшего профессионального образования «Сибирский федеральный университет» (СФУ) на 2007–2010 гг.

Рецензенты:
Красноярский краевой фонд науки;
Экспертная комиссия СФУ по подготовке учебно-методических комплексов дисциплин

М59 Микропроцессорные системы [Электронный ресурс] : электрон. учеб. пособие / О. В. Непомнящий, Е. А. Вейсов, Г. А. Скотников, М. В. Савицкая. – Электрон. дан. (4 Мб). – Красноярск : ИПК СФУ, 2009. – (Микропроцессорные исследования : УМКД № 1626/338–2008 / рук. творч. коллектива О. В. Непомнящий). – 1 электрон. опт. диск (*DVD*). – Систем. требования : *Intel Pentium* (или аналогичный процессор других производителей) 1 ГГц ; 512 Мб оперативной памяти ; 50 Мб свободного дискового пространства ; привод *DVD* ; операционная система *Microsoft Windows XP SP* 2 / *Vista* (32 бит) ; *Adobe Reader* 7.0 (или аналогичный продукт для чтения файлов формата *pdf*).
ISBN 978-5-7638-1657-0 (комплекса)
ISBN 978-5-7638-1660-0 (учебного пособия)
Номер гос. регистрации в ФГУП НТЦ «Информрегистр» 0320902478 (комплекса)

Настоящее издание является частью электронного учебно-методического комплекса по дисциплине «Микропроцессорные системы», включающего учебная программа дисциплины, наглядное пособие «Микропроцессорные системы. Презентационные материалы», лабораторный практикум, методические указания по самостоятельной работе, контрольно-измерительные материалы «Микропроцессорные системы. Банк тестовых заданий».

Рассмотрены современные микропроцессоры и микропроцессорные системы, а также микропроцессорные системы с датчиками. Выделены типовые микропроцессорные системы на основе микроконтроллеров Atmel. Приведены методы и способы проектирования. Описаны принципы функционирования микропроцессорных средств управления.

Предназначено для студентов направлений подготовки бакалавров 230100.62, магистров 230100.68 «Информатика и вычислительная техника»; специалистов 230201.65 «Информационные системы в технологии» укрупненной группы 230000 «Информатика и вычислительная техника». Может быть использовано студентами направления подготовки специалистов 090102.65 «Компьютерная безопасность» укрупненной группы 090000 «Информационная безопасность», а также аспирантами информационных специальностей.

Рекомендовано к изданию Инновационно-методическим управлением СФУ

Редактор Л. И. Злобина

Разработка и оформление электронного образовательного ресурса: Центр технологий электронного обучения Информационно-телекоммуникационного комплекса СФУ; лаборатория по разработке мультимедийных электронных образовательных ресурсов при КрЦНИТ

Подп. к использованию 30.11.2009
Объем 4 Мб
Красноярск: СФУ, 660041, Красноярск, пр. Свободный, 79

Оглавление

ПРЕДИСЛОВИЕ

Данное учебное пособие предназначено для студентов, изучающих дисциплины, связанные с проектированием микропроцессорных систем в различных областях науки и техники.

В пособии изложены основные сведения о современных микропроцессорах и микроконтроллерах, приведены архитектуры и классификация современных микропроцессоров и микроконтроллеров, системы команд и их сравнительные характеристики. Достаточно большое внимание уделено большим интегральным схемам, дополняющим микропроцессоры (таймеры, контроллеры прямого доступа к памяти, последовательные приемопередатчики и др.) Представлен лабораторный практикум.

В книге собраны последние материалы по современным микропроцессорам, приведены рекомендации по их применению. В основу положены лекции, которые читают преподаватели кафедры «Вычислительная техника» Политехнического института СФУ студентам специальности «Вычислительные машины, комплексы, системы и сети».

Учебное пособие предназначено для студентов направлений подготовки бакалавров 230100.62, магистров 230100.68 «Информатика и вычислительная техника» по курсам «Организация ЭВМ и систем», «Микропроцессорные системы», «Интерфейсы и периферийные устройства»; специалистов 230201.65 «Информационные системы в технологии» укрупненной группы 230000 «Информатика и вычислительная техника». Может быть использовано студентами направления подготовки специалистов 090102.65 «Компьютерная безопасность» по курсу «Аппаратные средства вычислительной техники», а также аспирантами информационных специальностей.

ВВЕДЕНИЕ

Современная микропроцессорная техника является важнейшим средством при решении самых разнообразных задач в области сбора и обработки данных, систем автоматического управления и др. Знания в этой области становятся необходимыми все более широкому кругу специалистов.

Микропроцессорная техника в науке и технике, промышленности, сельском хозяйстве, военной отрасли, быту и других областях жизнедеятельности человека становится все более востребованной. Практически любая электронная система, обладающая достаточной функциональной сложностью, реализуется с помощью микропроцессорных устройств.

История развития микропроцессорной техники насчитывает более 30 лет. Родоначальником микропроцессорной техники считается микропроцессор I14004 фирмы Intel, роль которого для развития современной техники трудно оценить.

На протяжении многих лет не прекращается острое соперничество ведущих электронных фирм за лидерство в этой высокоперспективной области. Результатом соперничества является разработка новых семейств и типов микропроцессоров, расширение их функциональных возможностей, быстрый рост производительности и снижение стоимости.

В процессе многолетнего развития произошла дифференциация микропроцессоров по функционально-структурным особенностям и областям применения. В настоящее время имеются следующие основные классы микропроцессоров:

универсальные микропроцессоры с CISC-архитектурой;
универсальные микропроцессоры с RISC-архитектурой;
специализированные микропроцессоры (сигнальные и др.);
микроконтроллеры.

Универсальные микропроцессоры с CISC-архитектурой (Complicated Instruction Set Computer – компьютер со сложным набором команд) применяются главным образом в персональных компьютерах и серверах.

Лидером в этой области является фирма Intel, которой комплектуется более 80 % выпускаемых персональных компьютеров.

Универсальные микропроцессоры с RISC-архитектурой (Reduced Instruction Set Computer – компьютер с сокращенным набором команд) применяются в основном в рабочих станциях и мощных серверах. Ведущими производителями считаются фирмы Sun Microsystems и MIPS Computer Systems.

В последние годы очень активно внедряются в различную аппаратуру RISC-микропроцессоры семейства PowerPC – совместная разработка фирм IBM, Motorola, Apple Computers. RISC-микропроцессоры выпускают также и

другие фирмы. Фирмой Inmos для построения мультипроцессорных систем разработаны транспьютеры – оригинальные RISC-микропроцессоры.

В классе специализированных микропроцессоров в настоящее время наиболее широко представлены DSP (Digital Signal Processor – процессор цифровой обработки сигналов), основными производителями которых являются фирмы Texas Instruments, Analog Devices, Motorola, NEC. Кроме DSP выпускаются микропроцессоры, специализированные для передачи информации в системах коммуникации – коммуникационные контроллеры, имеющие возможность обработки графической информации.

Микроконтроллеры являются наиболее массовым представителем микропроцессорной техники. Интегрируя на одном кристалле высокопроизводительный процессор, память и набор периферийных устройств, микроконтроллеры позволяют с минимальными затратами реализовать широкую номенклатуру систем управления различными объектами и процессами.

Использование микроконтроллеров в системах управления и обработки информации обеспечивает исключительно высокие показатели эффективности при достаточно низкой стоимости. Микроконтроллерам практически нет альтернативы, когда нужно создать качественные и дешевые системы. Иногда система может состоять только из одного микроконтроллера. Исключение составляет применение программируемых логических интегральных схем (ПЛИС) в области обработки сигналов в том случае, когда требуется параллельная обработка большого потока входных данных.

Основным классификационным признаком микроконтроллеров является разрядность микропроцессора. Имеются 4-, 8-, 16-, 32-разрядные микроконтроллеры. Разрядность микроконтроллера определяется точностью данных, необходимых для управления объектом. Наиболее массовыми и постоянно расширяющими свои области применения являются 8-разрядные микроконтроллеры, которые дешевле 16- и 32-разрядных и имеют большую функциональность.

В учебном пособии рассматриваются вышеперечисленные семейства и группы микропроцессоров и микроконтроллеров, средства и технологии создания систем обработки сигналов и др.

1. МИКРОПРОЦЕССОРЫ И МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ

1.1. Основы микропроцессора

Чтобы понять, что такое микропроцессор (МП) и как он работает, необходимо усвоить несколько понятий.

Автомат – устройство, выполняющее некий ограниченный набор функций самостоятельно по заданной программе.

Программа – набор команд, выполняемых автоматом.

Команда – задание на выполнение автоматом определенного действия.

Память программ – устройство, которое хранит программу автомата.

Рассмотрим работу шарманки – музыкального автомата. Шарманка состоит из набора регистров, каждый из которых способен при его возбуждении издавать звук определенной тональности. Порядок и число возбуждений в каждый момент времени определяются меткой на специальном диске, вращающемся по кругу. Значит, меняя число и положение меток, мы можем получать различные мелодии.

Итак, шарманка – это автомат, диск – это память программ, метка – это команда для автомата. Теперь несложно понять, что же такое микропроцессор.

Как уже упоминалось, микропроцессор выполняет определенный набор команд, например чтение/запись памяти программ и данных, пересылку и загрузку данных, в памяти ЭВМ. Правда, на ранних этапах развития вычислительной техники программы существовали в виде отверстий в перфокартах или перфолентах и заносились на специальные наборные поля. Сейчас, как правило, они хранятся на магнитных носителях – гибких и жестких магнитных дисках.

1.2. Основные исторические сведения о развитии микропроцессоров

Первый универсальный микропроцессор 4004 фирмы Intel появился в 1971 г. Он мог выполнять любую программу из системы своих команд, которых, кстати, было всего 45. Мог ввести данные, обработать их и вывести результаты. Длина слова этого микропроцессора составляла всего 4 бита, а адресное пространство ограничивалось 4,5 Кбит. Он был ориентирован на применение в калькуляторах. Микропроцессор содержал около 1000 транзисторов и выполнял 8000 операций в секунду.

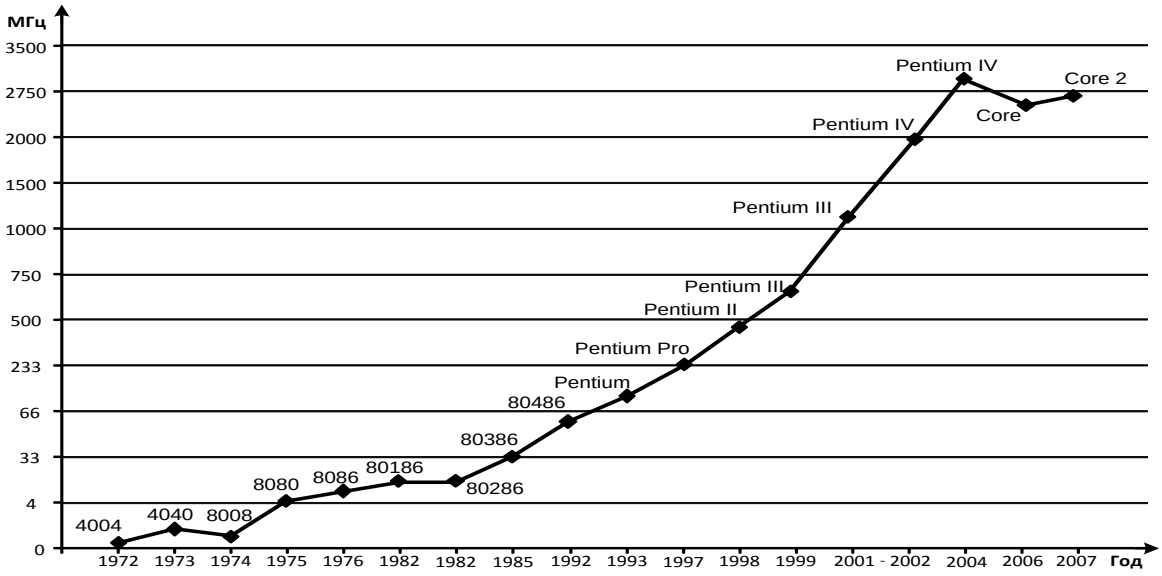


Рис. 1.1. Основные микропроцессорные семейства фирмы Intel

Через несколько лет фирма Intel выпустила микропроцессор 8008 (аналог 4004) с длиной слова 8 бит и 8080 – достаточно мощный для создания небольшого компьютера. Микропроцессор I8080 может выполнять десятичные и 16-битные арифметические операции, вызывать подпрограммы и адресовать память до 64 Кбайт. Шина данных имеет разрядность 8, а шина адреса – 16 бит. В России аналогом такого процессора стал микропроцессор КР580ИК80.

В последующие 10 лет число транзисторов в микропроцессоре увеличилось в 70 раз, размер слова составил 16 бит, а быстродействие возросло в 100 раз. Хотя уже были достигнуты некоторые физические ограничения для кристаллов, рынок стимулировал аналогичное развитие и в 1980-е гг. появляется микропроцессор I80386.

Последующей эволюцией в развитии микропроцессоров стало появление первого процессора со встроенным математическим сопроцессором I80486 в 1989 г. и Pentium в 1993-м.

В 1995 г. был разработан процессор Pentium Pro (150 МГц, 512 Кб кэш), позиционирующийся как серверный. Он отличался от аналогов большим кэшем и архитектурой, частично заимствованной у процессоров с архитектурой RISC. В Pentium Pro Intel впервые включил технологию динамического исполнения (Dynamic Execution), т. е. инструкции могут исполняться не только последовательно, но и параллельно с помощью предсказания ветвей кода и переупорядоченного исполнения инструкций. Тем самым значительно повысилась эффективность процессора – количество команд, выполняемых за такт.

В 1998 г. был выпущен процессор Pentium II Xeon. Системы, основанные на этом процессоре, могли быть сконфигурированы из 4, 8 и более процессоров.

В конце февраля 1999 г. были анонсированы Pentium III. Изготовлены по технологическому процессу 0,25 мкм, ядро Katmai, добавлен набор инструк-

ций SSE, размер L1 кэш – 32 Кб (16 + 16), L2 кэш – 512 Кб (работает на половине частоты ядра, расположен рядом с микросхемой процессора в картридже).

В конце ноября 2000 г. Intel представляет процессор Pentium 4 (кодовое название Willamette), архитектура NetBurst которого коренным образом отличается от своей предшественницы P6. Основным отличием было увеличение конвейера до 20 стадий, что позволило сильно увеличить частоту процессора. Тактовая частота первых экземпляров составила 1.4 и 1.5 ГГц. Интересный факт: арифметико-логическое устройство данного процессора работает на частоте, в два раза превышающей частоту ядра! В новом процессоре также обновился блок инструкций SSE, дополнился еще 144 инструкциями и стал именоваться SSE2. Претерпел изменения и кэш первого уровня, его объем сократился до 8 Кб для данных, для хранения инструкций появился новый переработанный кэш (Trace Cache).

Дальнейшее развитие процессоров Intel было связано с переходом на 64-битную архитектуру – IA-64 (Intel Architecture-64 bit). Основной идеологией новых процессоров стала технология EPIC (Explicitly Parallel Instruction Computing – обработка команд с явным параллелизмом). Само собой, сходств между RISC и EPIC осталось предостаточно, однако EPIC технологически выглядит намного совершеннее и по праву может считаться полноценной эволюцией идеологии RISC. Параллельность выполнения команд, которая является одним из ключевых преимуществ IA-64, достигается благодаря тому, что команды теперь поступают группами по три штуки (так называемый пучок команд), причем мощный процессор способен за один цикл обработать сразу несколько подобных пучков. Внутри процессора выполнение команд распределяется между соответствующими функциональными блоками, многие из которых для обеспечения идеальной параллельности дублируют возможности друг друга (то есть присутствует несколько блоков для целочисленных вычислений, несколько для вычислений с плавающей точкой и т. д.). Первое воплощение архитектуры IA-64 появилось на свет в мае 2001 г., им стал процессор Itanium.

В настоящее время увеличение производительности процессоров, в основном, ведется за счет применения многоядерной архитектуры Multi-core Intel Processors. Одним из представителей такой архитектуры является линейка процессоров Intel Xeon Processor 5400, произведенных по 45-нанометровой технологии. На базе данных процессов возможно построение современных высокопроизводительных масштабируемых многопроцессорных систем.

1.3. Микропроцессор – основа ЭВМ

Микропроцессор, как и любое вычислительное устройство, состоит из двух основных блоков: управляющего и операционного.

Микропроцессор является важнейшей составной частью ЭВМ (рис. 1.2). Рассмотрим состав и функции основных элементов ЭВМ.

В состав ЭВМ входят процессор, оперативная память и внешние устройства.

Процессор служит для управления всеми элементами системы и организует работу по выполнению заданной программы (преобразование данных).

Память подразделяется на оперативное запоминающее устройство (ОЗУ) и пассивное запоминающее устройство (ПЗУ) и служит для хранения программ и данных.

Внешние устройства обеспечивают ввод/вывод программ и данных в ЭВМ.

В составе ЭВМ имеется центральный процессорный элемент, соединенный со всеми элементами системы при помощи системной магистрали. Системная магистраль, в свою очередь, состоит из трех шин: адреса, данных и управления.

Шина адреса предназначена для передачи текущего адреса, к которому идет обращение, всем элементам ЭВМ.

Шина данных предназначена для передачи данных между центральным процессором, памятью и внешними устройствами, а также между оперативной памятью (ОП) и внешним устройством (ВУ) в режиме прямого доступа в память (ПДП).

Шина управления предназначена для передачи сигналов управления между МП и остальными элементами ЭВМ.

Вся работа ЭВМ – это выполнение какой-либо программы, будь то программа операционной системы или программа пользователя. Достаточно рассмотреть порядок выполнения программы процессором для того, чтобы понять, как работает ЭВМ.

Работа ЭВМ начинается с программы начальной установки (BIOS), которая подготавливает все элементы к работе. В IBM PC программируют таймер, контроллеры дисков, адаптеры параллельной и последовательной связи и т. д. Затем загружается ядро операционной системы (с диска или дискеты) и управление передается этой программе. Теперь ЭВМ готова к выполнению программ.

1.4. Микропроцессорные системы

Микропроцессор – это микросхема или совокупность микросхем (или кристаллов), выполняющая арифметические и логические операции над данными и осуществляющая программное управление вычислительным процессом.

Микропроцессорные средства – это наборы микросхем (БИС), комплекты, совместимые по уровням напряжений, сигналам и передаваемой ин-

формации, в состав которых входят: МП, ОЗУ, ПЗУ, управление вводом/выводом и т. д.

1.4.1. Классификация микропроцессоров. Понятие о разрядности и системе команд

Вообще говоря, понятие классификации микропроцессоров довольно субъективно. Каждый, кто считает себя посвященным в существо вопроса, может предложить собственную (отличную от других) классификацию. Классификация необходима для упорядочивания всего многообразия существующих МП, выработки критериев их оценки. В силу этих причин на первый план могут выступать различные критерии классификации. Рассмотрим наиболее распространенные.

По назначению:

универсальные, предназначены для выполнения функций управления и вычисления;

специализированные, выполняют только узкоспециализированные функции вычисления или управления (вычисление сложных алгебраических функций или управление конкретным устройством – станком, автомобилем и т. д.).

По виду обрабатываемой информации:

цифровые, работают с бинарными (2-уровневыми) сигналами, обозначающими логический нуль и единицу. Обычно имеет смысл ссылка на тип цифрового сигнала: ТТЛ, ЭСЛ или КМОП-уровень;

аналоговые, работают с аналоговыми (непрерывными) уровнями сигналов. В случае аналогового сигнала имеют смысл предельные значения входного напряжения или тока.

По разрядности данных:

фиксированные, в случае фиксированной разрядности указывается конкретное значение длины информации (бит, байт, слово и т. д.);

переменные, в случае переменной разрядности указывается значение кванта, на который возможно наращивание разрядности (2, 4 или 8 бит).

По тактовой частоте:

статические, имеют нижний предел тактовой частоты равный нулю, т. е. при отсутствии тактовой частоты МП перейдет в состояние «Ожидание», а по ее появлении продолжит свою работу;

динамические, имеют нижний предел тактовой частоты не равный 0, т. е. при снижении частоты синхронизации ниже придельного уровня МП перестает нормально функционировать.

По виду синхронизации:

синхронные;

асинхронные.

Предлагаемая классификация во многом повторяет предыдущую.

По компоновке:

однокристальные;
многокристальные;
многокристальные секционные.
По числу управляющих магистралей:
совмещенные;
раздельные.
По системе команд:
фиксированная;
переменная.

Разрядность микропроцессора

Под разрядностью микропроцессора следует понимать величину его разрядной сетки, определяемой соотношением разрядности шины данных и адреса. Строго говоря, не существует точного определения разрядности, но, как правило, под разрядностью понимают ширину поля данных. В литературе можно зачастую встретить так называемые 8-, 16-, 32x64 или 8/16-разрядные процессоры. Так, МП Intel 8080 (I8080) имеет 8-разрядную шину данных и 16-разрядную шину адреса, но по типу обрабатываемых данных относится к 8-разрядному процессору. МП Intel 80386 (I80386) принято называть 32-разрядным процессором, так как он имеет 32-разрядные раздельные шины адреса и данных. Существуют также понятия разрядности машинного слова, разрядности внутренних устройств МП, но под любым определением разрядности следует понимать максимально возможную величину обрабатываемых данных, выраженную в битах (иногда в байтах, словах или двойных словах). Следует заметить, что существуют микропроцессорные системы (МПС) с изменяемой разрядностью данных.

Команда и система команд микропроцессора

Под командой следует понимать задание на выполнение микропроцессором определенного действия. Система команд МП – это набор функций, определенных для микропроцессора. Наиболее существенными для ознакомления с особенностями системы команд являются три признака: длина команды, функциональный признак и способ адресации.

По длине (или по величине занимаемых байтов) команды подразделяют на однобайтовые, двухбайтовые, трехбайтовые и т. д. При этом первый байт (или слово – в более мощных процессорах) всегда отводится под код команды, а последующие содержат либо данные, либо адрес, по которому они хранятся в памяти.

По функциональным признакам, т. е. по виду выполняемых действий, команды подразделяют на следующие группы: группа команд пересылки, группа арифметических команд, группа логических команд, группа команд переходов, группа команд управления и работы со стеком, группа команд

ввода/вывода, группа команд управления процессором, группа специализированных команд.

По способу адресации (т. е. по виду обращения к памяти или внутренним устройствам МП) различают следующие виды команд: регистровая адресация (команды обращения к внутренним регистрам МП), команды непосредственного обращения к памяти, команды косвенного обращения (команды, в которых адрес ячейки памяти указан не явно, а через указатель, хранящийся во внутреннем регистре процессора или в ячейке памяти). Существуют также всевозможные комбинации адресации в различных МП.

Рассмотрим одну из простейших функций МП – сложение двух чисел.

Допустим, что структура командного слова МП имеет такой вид:

Код операции	Адрес операнда	Адрес следующей команды

Таблица 1.1

Дамп памяти

Ячейка ОЗУ	Команда	Действие
2051	01 0641 2052	Запись содержимого ячейки 0641 в аккумулятор
2052	15 0642 2053	Сложение содержимого ячейки 0642 в аккумуляторе
2053	02 0643 2054	Запись содержимого аккумулятора в ячейку 0643
2054	00 0000 0000	Стоп

0641	Слагаемое А	
0642	Слагаемое В	
0643	Результат	

Тогда структура программы будет иметь следующий вид:

1-й шаг – чтение слагаемых из ОЗУ;

2-й шаг – запись слагаемых в арифметико-логическое устройство (АЛУ);

3-й шаг – сложение;

4-й шаг – запись результата в ОЗУ;

5-й шаг – останов.

Допустим, коды операций следующие:

01 – вызов операнда из ОЗУ в аккумулятор;

02 – запись содержимого аккумулятора в ОЗУ;

15 – сложение содержимого аккумулятора с содержимым ячейки ОЗУ;

00 – останов.

Из табл. 1.1 видим, что каждой команде МП соответствует код операции (КОП). Количество операций микропроцессора определяется величиной его внутреннего регистра команд. Так, если регистр команд имеет разрядность 8 бит, то очевидно, что в системе команд такого микропроцессора не может присутствовать более чем 256 возможных операций.

1.4.2. Основные характеристики и критерии производительности микропроцессора

К характеристикам МП следует отнести в первую очередь следующие: тип микроэлектронной технологии, количество кристаллов и их типоразмеры, количество транзисторов на кристалле, количество выводов корпуса кристалла. Эти технологические характеристики определяют экономическую целесообразность выпуска самого МП. Далее идут разрядность обрабатываемого слова, быстродействие микропроцессора (тактовая частота на внутренних и внешних шинах, количество и время выполнения основных операций в секунду), емкость адресуемой памяти, тип управляющих устройств, эффективность системы команд, число уровней прерывания и прямого доступа к памяти, пропускная способность интерфейсов ввода/вывода, количество и уровни питающих напряжений, номинальные параметры используемых сигналов, потребляемая мощность; число, состав и назначение дополнительных устройств, входящих в микропроцессорный комплект; поддержка проектирования программного обеспечения и некоторые специализированные характеристики.

Основной характеристикой любой МПС является ее производительность, под которой в общем случае понимают количество выполняемых в единицу времени элементарных операций и время доступа к памяти и внешним устройствам. Критериями максимальной производительности МПС следует считать в первую очередь минимальное время доступа к памяти и максимально возможную тактовую частоту процессора.

1.4.3. Архитектура простейших микропроцессорных систем

Магистрально-модульный принцип построения МПС показан на рис. 1.2.

В МПС все связи между отдельными функциональными блоками осуществляются, как правило, шинами.

Под шиной подразумевается физическая группа передачи сигналов, обладающих функциональной общностью (по каждой линии передается один двоичный разряд информации).

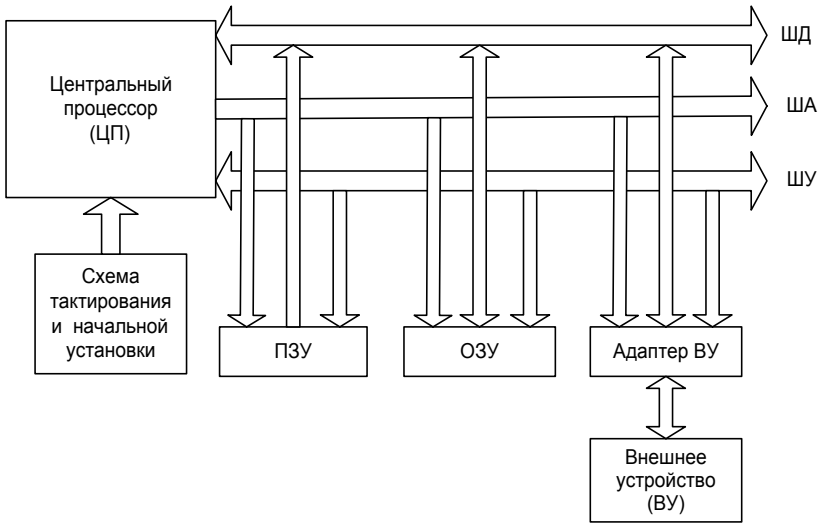


Рис. 1.2. Простейшая микропроцессорная система

Физически шины реализуются в виде параллельных проводящих участков печатной платы или жгутов. Кроме шины данных (ШД), как правило, различают шину адреса (ША) и шину управления (ШУ). Передаваемые по ША адреса формируются в МП. Они необходимы для определения пути передачи данных внутри МПС, в том числе для выбора ячейки памяти, куда необходимо занести или откуда необходимо считать информацию. В определении такта передачи могут принимать участие и управляющие сигналы, подсоединяющие или, наоборот, блокирующие те или иные устройства МПС. В отличие от ША и ШУ шина данных является шиной двунаправленной. Данные по этой шине могут передаваться от микропроцессора к какому-нибудь устройству МПС либо пересылаться в МП от какого-то устройства, доступ к которому обеспечивают сигналы адресной шины. Естественно, что в каждый момент времени данные могут передаваться лишь в одном направлении, определяемом режимом работы микропроцессора.

К основным режимам работы следует отнести: 1) запись данных в память машины; 2) чтение данных из памяти машины; 3) пересылку данных в устройство ввода/вывода; 4) чтение данных с устройства ввода/вывода; 5) выполнение операций с содержимым внутренних регистров микропроцессора. При реализации последнего режима внешние по отношению к МП шины МПС не используются, т. е. все действия происходят внутри МП. Реализация первых четырех режимов оказывает определяющее влияние на работу шины данных. Работа по реализации любой программы МПС, построенной по типу архитектуры с тремя шинами, состоит в выполнении следующих действий для каждой команды программы:

1. Микропроцессор формирует адрес, по которому хранится код операции команды, переводя в соответствующее состояние шину адреса.
2. Код операции считывается из памяти по сформированному адресу и пересылается в микропроцессор.

3. Микропроцессор дешифрирует (идентифицирует) команду.
4. Микропроцессор настраивается на выполнение одного из перечис-ленных выше пяти основных режимов в соответствии с результатами дешифрирования считанного из памяти кода команды.

Перечисленные выше пять режимов являются основными, но не единственно возможными. Существуют и другие, но они будут рассмотрены при изучении конкретного микропроцессора.

1.4.4. Архитектуры многопроцессорных вычислительных систем. Принципы построения MPP- и SMP-систем

Симметричная многопроцессорная архитектура SMP

Главной особенностью систем с архитектурой SMP (symmetric multiprocessing) является наличие общей физической памяти, разделяемой всеми процессорами (рис. 1.3).

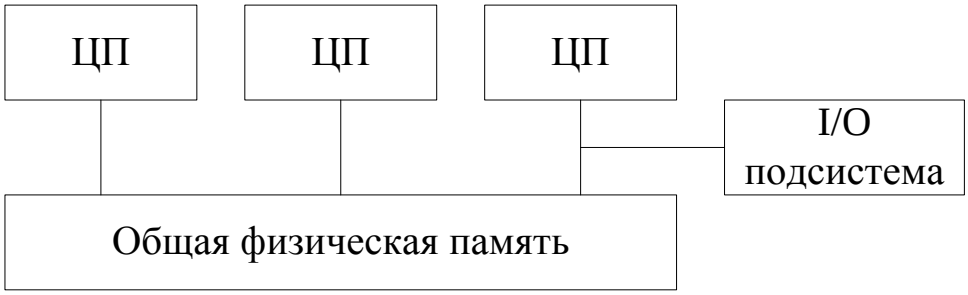


Рис. 1.3. Архитектуры SMP

Память служит, в частности, для передачи сообщений между процессорами, при этом все вычислительные устройства при обращении к ней имеют равные права и одну и ту же адресацию для всех ячеек памяти. Поэтому SMP-архитектура называется симметричной. Последнее обстоятельство позволяет очень эффективно обмениваться данными с другими вычислительными устройствами. SMP-система строится на основе высокоскоростной системной шины (SGI PowerPath, Sun Gigaplane, DEC TurboLaser), к слотам которой подключаются функциональные блоки типов: процессоры (ЦП), подсистема ввода/вывода (I/O) и т. п. Для подсоединения к модулям I/O используются уже более медленные шины (PCI, VME64). Наиболее известными SMP-системами являются SMP-серверы и рабочие станции на базе процессоров Intel (IBM, HP, Compaq, Dell, ALR, Unisys, DG, Fujitsu и др.). Вся система работает под управлением единой ОС (обычно UNIX-подобной, но для Intel-платформ поддерживается Windows NT). ОС автоматически (в процессе работы) распределяет процессы по процессорам, но иногда возможна и явная привязка.

Основные преимущества SMP-систем:

- простота и универсальность для программирования. Архитектура SMP не накладывает ограничений на модель программирования, используемую при создании приложения: обычно используется модель параллельных ветвей, когда все процессоры работают независимо друг от друга. Однако можно реализовать и модели, использующие межпроцессорный обмен. Использование общей памяти увеличивает скорость такого обмена, пользователь также имеет доступ сразу ко всему объему памяти. Для SMP-систем существуют довольно эффективные средства автоматического распараллеливания;
- простота эксплуатации. Как правило, SMP-системы используют систему кондиционирования, основанную на воздушном охлаждении, что облегчает их техническое обслуживание;
- относительно невысокая цена.

Недостаток SMP-систем – системы с общей памятью плохо масштабируются.

Этот существенный недостаток SMP-систем не позволяет считать их по-настоящему перспективными. Причиной плохой масштабируемости является то, что в данный момент шина способна обрабатывать только одну транзакцию, вследствие чего возникают проблемы разрешения конфликтов при одновременном обращении нескольких процессоров к одним и тем же областям общей физической памяти. Вычислительные элементы начинают друг другу мешать. Когда произойдет такой конфликт, зависит от скорости связи и от количества вычислительных элементов. В настоящее время конфликты могут происходить при наличии 8–24 процессоров. Кроме того, системная шина имеет ограниченную (хоть и высокую) пропускную способность (ПС) и ограниченное число слотов. Все это очевидно препятствует увеличению производительности при увеличении числа процессоров и числа подключаемых пользователей. В реальных системах можно задействовать не более 32 процессоров. Для построения масштабируемых систем на базе SMP используются кластерные или NUMA-архитектуры. При работе с SMP-системами используют так называемую парадигму программирования с разделяемой памятью (shared memory paradigm).

Массивно-параллельная архитектура MPP

Главная особенность массивно-параллельной архитектуры (MPP – massive parallel processing) состоит в том, что память физически разделена. В этом случае система строится из отдельных модулей, содержащих процессор, локальный банк операционной памяти (ОП), коммуникационные процессоры (рутеры) или сетевые адаптеры, иногда – жесткие диски и/или другие устройства ввода/вывода. По сути такие модули представляют собой полнофункциональные компьютеры (рис. 1.4). Доступ к банку ОП из данного модуля имеют только процессоры (ЦП) из этого же модуля. Модули соединяются специальными коммуникационными каналами. Пользователь может определить логический номер процессора, к которому он подключен, и организовать обмен

сообщениями с другими процессорами. Используются два варианта работы операционной системы (ОС) на машинах MPP-архитектуры. В одном полноценная операционная система (ОС) работает только на управляющей машине (front-end), на каждом отдельном модуле функционирует сильно урезанный вариант ОС, обеспечивающий работу только расположенной в нем ветви параллельного приложения. Во втором варианте на каждом модуле работает полноценная UNIX-подобная ОС, устанавливаемая отдельно.

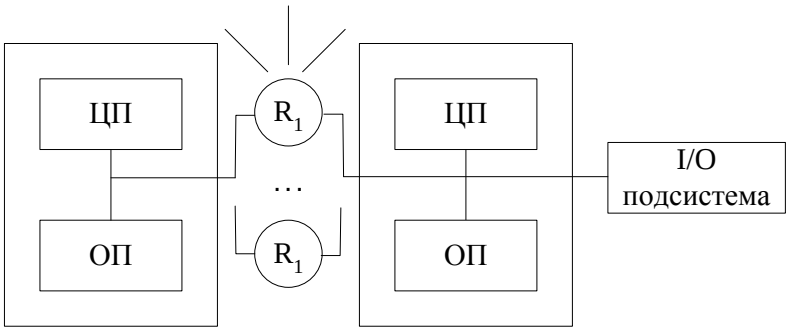


Рис. 1.4. Архитектуры с раздельной памятью MPP

Главным преимуществом систем с раздельной памятью является хорошая масштабируемость: в отличие от SMP-систем в машинах с раздельной памятью каждый процессор имеет доступ только к своей локальной памяти, в связи с чем не возникает необходимости в потактовой синхронизации процессоров. Практически все рекорды по производительности на сегодня устанавливаются на машинах именно такой архитектуры, состоящих из нескольких тысяч процессоров (ASCI Red, ASCI Blue Pacific).

Недостатки:

- отсутствие общей памяти заметно снижает скорость межпроцессорного обмена, поскольку нет общей среды для хранения данных, предназначенных для обмена между процессорами. Требуется специальная техника программирования для реализации обмена сообщениями между процессорами;
- каждый процессор может использовать только ограниченный объем локального банка памяти;
- вследствие указанных архитектурных недостатков требуются значительные усилия для того, чтобы максимально использовать системные ресурсы. Именно этим определяется высокая цена программного обеспечения для массивно-параллельных систем с раздельной памятью.

1.5. Структура однокристального МП, состав и назначение элементов

Структура однокристального МП приведена на рис. 1.5.

Доступные блоки МП (выделены на рисунке толстой рамкой):

регистр-аккумулятор,

счетчик команд,
блок регистров B, C, D, E,
регистр признаков.
Недоступные блоки МП:
регистр адреса,
схема управления,
арифметико-логическое устройство,
блок регистров временного хранения данных,
регистр команд.

При работе с микропроцессором программисту необходимо иметь информацию о числе и назначении всех регистров, специальных указателей, регистра флагов, системы команд. Число и назначение регистров, флагов и команд программист, как правило, изменить не может. Он может изменить только содержимое регистров и использовать команды в любой нужной ему комбинации. Как известно, под регистром подразумевается специальное запоминающее устройство (ЗУ), состоящее из элементов (триггеров) с двумя устойчивыми состояниями. Число элементов 8 соответствует одному байту. Существуют 8-, 16-, 32-, 64- и т. д. разрядные регистры. Все регистры разбиты на группы и различаются функциональным назначением.

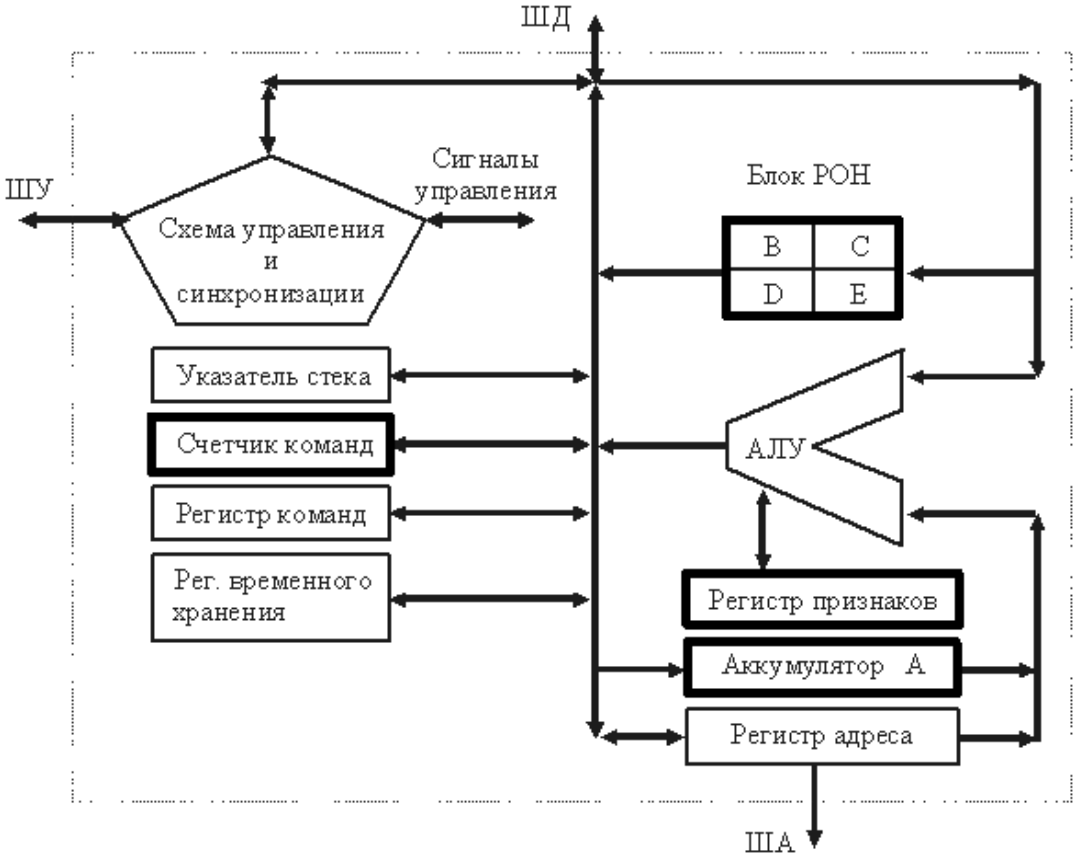


Рис. 1.5. Структура однокристального МП

Основными блоками МП являются: блок регистров общего назначения (РОН) со схемой выборки регистров; регистр комад с дешифратором команд

и формирователем машинных циклов; арифметико-логическое устройство с регистром-аккумулятором, выполняющим арифметические и логические операции; регистры временного хранения данных W и Z; флаговый регистр; устройство управления и синхронизации.

Регистры общего назначения используются для хранения данных и промежуточных результатов вычислений, выполняемых с помощью арифметико-логического устройства. Они позволяют адресоваться как ко всему регистру, так и к отдельным байтам или словам.

Аккумулятор – специальный регистр, как правило, одно-или двухбайтовый. При выполнении арифметических и логических операций служит источником одного из операндов и местом хранения результатов выполнения операций.

Регистр команд – регистр, в котором хранится код выполняемой команды. Этот регистр, как указывалось выше, является недоступным регистром. Это означает, что не существует команды, которая могла бы изменить его содержимое. После выполнения очередной команды в регистр автоматически заносится код следующей команды из ячейки оперативной памяти, адрес которой находится в счетчике команд.

Счетчик команд – регистр (PC – Program Counter), хранящий адрес следующей команды, которая должна быть выполнена вслед за предыдущей. Счетчик команд автоматически получает приращение хранимого в нем адреса в зависимости от того, какую по длительности команду микропроцессор считывает из памяти, указывая всегда на первый байт следующей команды. На содержимое регистра можно повлиять только с помощью команд, изменяющих последовательное выполнение программы (например, команд перехода или вызова подпрограмм).

Указатель стека – регистр (SP – Stack Pointer), хранящий адрес очередной ячейки стека. Стеком называется особым образом организованный участок памяти, выделяемый программистом для временного хранения содержимого внутренних регистров МП, со специальным режимом доступа. Эта область оперативной памяти необходима в том случае, когда нужно прекратить выполнение реализуемой последовательности команд и вернуться к ней позже. Например, для немедленного выполнения специальной подпрограммы или при прерывании программы данные от МП поступают в верхнюю часть стековой памяти. Содержимое указателя стека уменьшается на единицу (2, 4, в зависимости от формата заносимых данных), чтобы всегда указывать на адрес последней заполненной ячейки стека (дно свободного пространства стека). Когда же данные выбираются (считываются) из стека, содержимое указателя стека увеличивается с каждым выбранным байтом (словом, двойным словом). Операции со стеком называются стековыми. С их помощью легко организуются многоуровневые (вложенные) прерывания: аппаратные и программные.

Флаговый регистр – регистр, в простейшем случае содержащий 5 двоичных разрядов, называемых флагами по числу хранимых в нем специальных признаков результатов некоторых операций. Иногда его называют регистром

признаков, или регистром флагов. Значение флага указывает на результат выполнения операций. Например, в микропроцессоре I8080 флаговый регистр содержит 5 флагов (рис. 1.6):

S	Z		AC		P	C

S – знак (Sign)
Z – нуль (Zero)
AC – вспомогательный перенос (Auxiliary Carry)
P – четность (Parity)
C – перенос (Carry)

Рис. 1.6. Регистр флагов МП I8080

Флаги всегда устанавливаются или сбрасываются автоматически после выполнения очередной команды, влияющей на флаги, в зависимости от результата операции. Флаг считается установленным, если флаговый разряд принимает значение 1, и сброшенным, если значение разряда 0. Состояние флагов используется в командах условного перехода. Результаты выполнения арифметических и логических операций над содержимым аккумулятора и регистров общего назначения или содержимым ячеек памяти оказывают влияние на флаги следующим образом.

Флаг нуля устанавливается в состояние 1, если после выполнения какой-либо команды получен нулевой результат, и сбрасывается в 0 в случае ненулевого результата.

Флаг переноса устанавливается в 1, если в результате операций сложения и сдвига появляется единица переноса из старшего разряда после выполнения операций вычитания или сравнения, в противном случае флаг сбрасывается в 0.

Флаг знака устанавливается в 1, если в результате выполнения операций появляется логическая единица в старшем разряде байта данных (указание на отрицательный результат), и сбрасывается в 0 в случае нулевого значения старшего разряда (указание на положительный результат).

Флаг четности устанавливается в 1, если после выполнения операций сумма единиц в байте данных, подсчитываемых с помощью операции сложения по модулю 2 (значение суммы по модулю 2 равно 0), и сбрасывается в 0 в противном случае (число единиц нечетное).

Флаг дополнительного переноса устанавливается в 1, если в результате выполнения команды появляется сигнал переноса из третьего разряда в четвертый в байте данных результата. Если такого переноса нет, то флаг дополнительного переноса сбрасывается в 0. Сигнал этого флага используется во многих схемах вычислений, однако он особенно необходим для сложения чисел в двоично-десятичной форме.

1.6. Многоядерные микропроцессорные системы

Для решения задач, требующих разработки систем с повышенной производительносью, широкое распространение получили многоядерные микропроцессорные системы. Причиной появления таких процессоров стал рост требований к производительности микропроцессороных систем. До последнего времени одним из основных методов повышения производительности было повышение тактовой частоты процессоров при одновременном совершенствовании систем буферизации обмена данными с основной памятью. Этот рост частот становился возможным по мере уменьшения размеров отдельных элементов микросхем при переходе к новым техпроцессам.

Однако в настоящее время такой путь развития стал достаточно трудной задачей в силу действия ограничений, определяемых законами физики. Например, скорость доступа к памяти растет не так быстро, как скорость работы вычислительных устройств, что может свести на нет преимущества от повышения частоты процессора.

Практически единственным решением этой задачи в рамках существующих технологий является распараллеливание вычислений, суть которого состоит в том, что программа на уровне исходного программного кода может быть разделена на несколько независимых потоков команд, выполняемых на самостоятельных вычислительных блоках. Варианты построения таких систем были известны и проработаны достаточно давно. Сейчас же эти варианты переносятся с уровня законченных процессорных блоков на уровень микросхем процессоров; при этом многопроцессорность дополняется суперскалярностью.

Первой в технологической гонке на пути создания двухъядерных микропроцессоров оказалась фирма IBM, начав в 2001 г. продажу двухъядерного процессора IBM Power4 для серверов. В 2002 г. почти одновременно AMD и Intel объявляют о перспективах создания своих двухъядерных процессоров, и в этом же году появляются процессоры Intel Xeon и Intel Pentium 4 с технологией Hyper-Threading. В 2004 г. свой двухъядерный процессор выпустила Sun (UltraSPARC IV), а таже ARM (MPCore). В том же году IBM выпустила второе поколение своих двухъядерных процессоров IBM Power5.

В 2005 г. Intel выпустила первый в мире двухъядерный процессор архитектуры x86. Почти одновременно AMD анонсировала полную линейку двухъядерных процессоров Opteron и Athlon 64 X2.

1.7. Управляющий автомат простейшей микропроцессорной системы

1.7.1. Алгоритм управляющего автомата

Управляющее устройство МП состоит из двух независимых частей:

1. Первичного автомата, управляющего процессами внутри МП (ПУА).
2. Схемы, обрабатывающей осведомительные сигналы и генерирующей управляющие сигналы МПС.

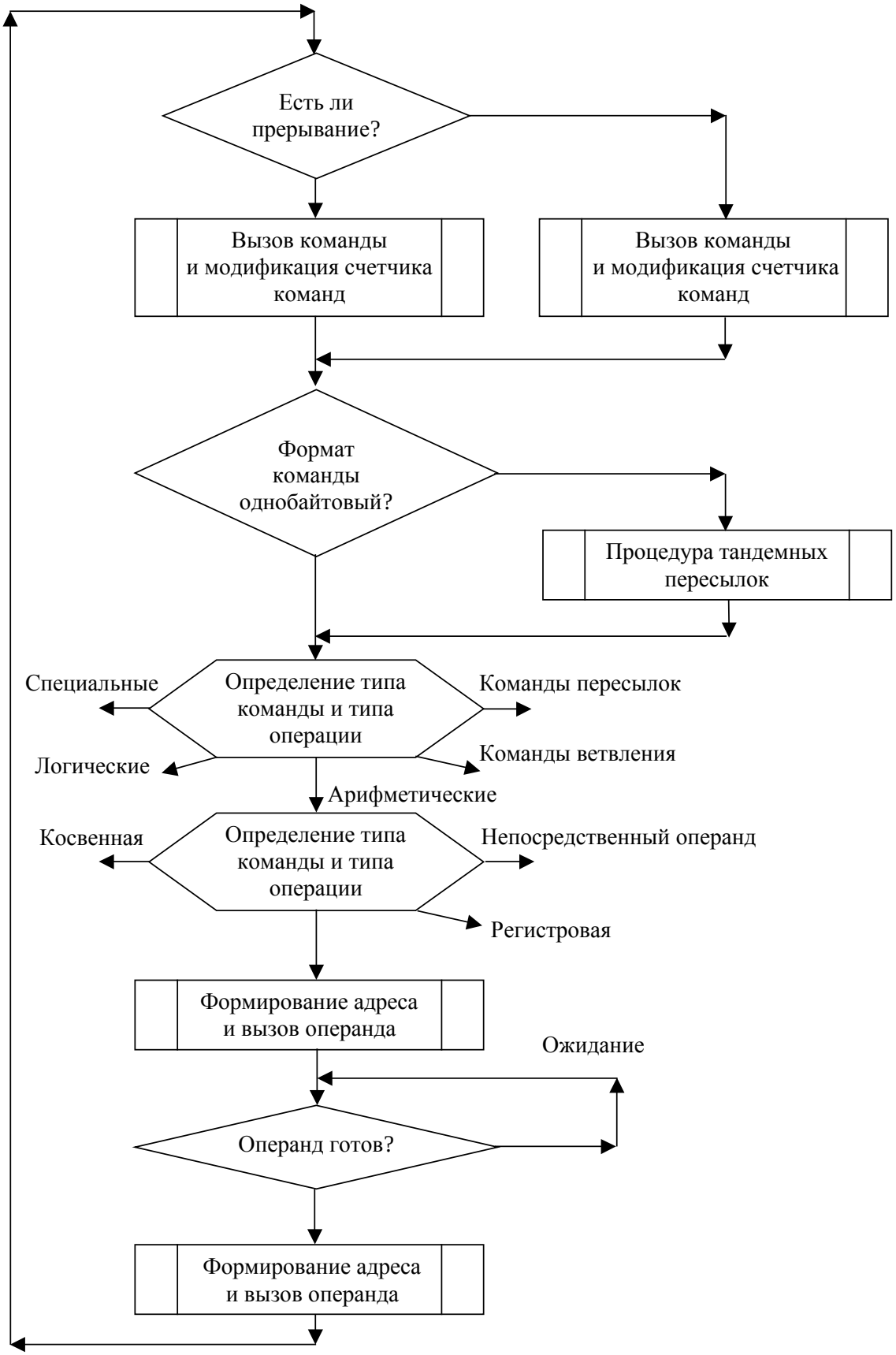


Рис. 1.7. Примерная схема алгоритма
функционирования управляющего автомата в течение цикла

Первичный управляющий автомат (рис. 1.7) работает следующим образом. Он инициирует начало выполнения очередной команды, вырабатывая микроприказ выдачи содержимого счетчика команд PC на шину адреса. Выбранный из памяти байт информации помещается в регистр команд. Первичный управляющий автомат (ПУА), в состав которого входит дешифратор команды, приступает к преобразованию кода команды в серию микроприказов, реализующих в МП определенную операцию.

1.7.2. Цикл команды в МПС

Цикл команды состоит из двух фаз: выборки команды и ее исполнения. Фаза выборки команды одинакова для всех команд микропроцессора. Последняя операция фазы исполнения команды, а именно размещение результата в аккумуляторе или в одном из регистров, также одинакова для целого ряда команд. Классификация команд по типам отражается в самом коде команды, таким образом существенно упрощается схема декодирования команды.

Существенная особенность работы первичного автомата состоит в том, что его алгоритм содержит условный оператор ожидания готовности операнда. Появление такого оператора в алгоритме объясняется тем, что МП приспособлены для работы с различными типами внешней системной памяти (с непосредственным, прямым или последовательным доступом), имеющими разные времена обращения. Кроме того, МП может обращаться за операндом не только к памяти, но и медленно действующим внешним устройствам ввода/вывода. Наличие в схеме алгоритма первичного автомата оператора ожидания готовности операнда является одной из причин того, что последовательность микрокоманд, реализующая некоторую команду, генерируется первичным автоматом не только на основе кода команды, но и под воздействием внешних управляющих сигналов.

Схема управления системой в зависимости от кода текущей команды, состояния ПУА, а также от значения осведомительных сигналов в шине управления вырабатывает управляющие воздействия, которые реализуют процедуры системного обмена информацией.

1.7.3. Тактирование МП и синхронизация МПС

В МП управляющий автомат в зависимости от сложности команды реализует цикл команды за несколько внутренних машинных циклов. В простейшем случае цикл команды реализуется за 1–5 машинных циклов. Один машинный цикл требуется МП для одного обращения к памяти или устройству ввода/вывода (УВВ). Выборка байта команды или каждого байта адреса или данных (а также их условного представления) требует одного машинного

цикла. Аналогичность операций, выполняемых в этих циклах, несмотря на то, что они расположены в различных фрагментах блок-схемы алгоритма работы устройства управления, позволяет реализовывать их в течение цикла команды на одном и том же оборудовании первичного автомата. Эффективность работы управляющего автомата достигается за счет того, что машинные циклы могут быть переменной длины. Так, в МП I8080 каждый цикл может состоять из 3–5 тактов. Тактирование МП от внешнего генератора показано на рис. 1.8.

Каждый такт машинного цикла образует пара сигналов тактирования F1 и F2, поступающих от внешнего генератора. В начале каждого машинного цикла первичный автомат генерирует сигнал синхронизации МП системы SYNC. Каждому такту Т соответствует отдельное состояние первичного автомата управляющего устройства МП. На рис. 1.9 представлена блок-схема алгоритма работы первичного управляющего автомата.

Все такты Т1–Т5 имеют одинаковую длительность. Существуют три исключения из этого положения:

1. Состояние WAIT (Tw), в котором МП находится в ожидании операнда.
2. Состояние HOLD (Twh), в которое МП переходит под воздействием внешних сигналов управления МП системой.
3. Состояние HALT, в которое МП может быть введен командой останова.

Названные состояния МП не связаны с тактовой частотой сигналов F1 и F2, и их продолжительность не определена, так как зависит от внешних по отношению к МП событий. Эти состояния длятся целое число тактов, и выход из них МП тактируется. Таким образом, каждый период тактирования МП соответствует особому состоянию первичного автомата. В стандартном машинном цикле может быть от трех до пяти состояний автомата (Т1-3, Т1-4, Т1-5). Цикл команды содержит от одного до пяти машинных циклов. В зависимости от сложности операций, определяемых командой, цикл команды может быть реализован первичным автоматом с числом переходов по внутренним состояниям от четырех до восемнадцати.

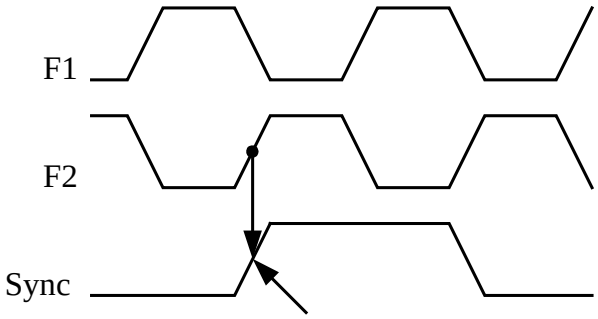


Рис. 1.8. Тактирование МП

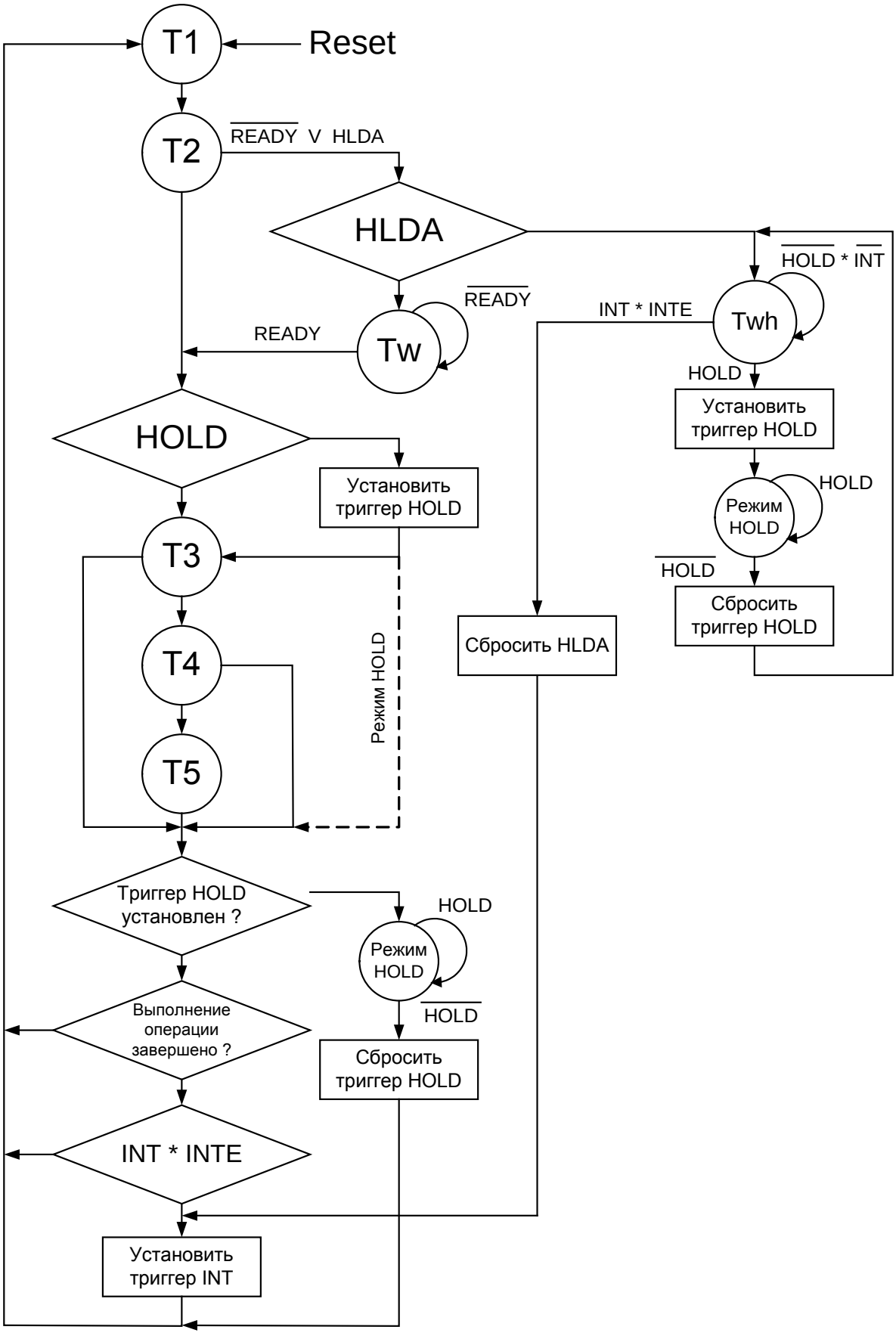


Рис. 1.9. Алгоритм работы первичного управляющего автомата

На предложенной выше блок-схеме первичного автомата (рис. 1.9) можно отметить следующее:

1. Микропроцессор позволяет приступить к анализу запросов на прерывание только после окончания выполнения текущей команды.
2. Из состояния останова HALT МП может быть выведен двумя способами: поступившим внешним сигналом прерывания и соответствующим разрешением на него или сигналом системной установки в исходное состояние RESET, который переводит первичный автомат в состояние T1.

1.7.4. Слово состояния МП как средство управления системой

Для нормального функционирования МП-системы недостаточно управляющих сигналов, генерируемых простейшим микропроцессором. МПС в каждом машинном цикле должна получать полную информацию о состоянии МП. Эта задача решается с помощью применения специального регистра состояния SL (Status Latch). МП I8080 в первом такте каждого машинного цикла генерирует на шине данных слово состояния PSW (Program Status World), информирующее МП-систему о процессах, которые происходят в МП. Так как сигнал синхронизации SYNC вырабатывается в начале каждого машинного цикла, он используется в качестве сигнала, идентифициру-ющего информацию, представленную на шине данных как слово состояния МП. Слово состояния под воздействием сигнала SYNC загружается в SL. Сигналы на выходах SL используются в качестве сигналов управления периферией МПС. Каждый разряд слова PSW имеет свое символьное имя и заводится на соответствующие входы адаптеров или устройство ввода/вывода, определяя тем самым режим их функционирования в соответствии с данным текущим состоянием МП. Таким образом, управление МПС осуществляется генерацией управляющих воздействий на двух уровнях:

1. На уровне микроприказов по шине управления собственно МП в каждом такте работы первичного автомата T.
2. На уровне мини-приказов путем генерации слова состояния PSW в каждом машинном цикле M. Выходы регистра состояния SL образуют шину управления МП-системой.

1.7.5. Управляющее устройство МП. МПС под управлением первичного автомата

Граф состояний устройства управления, а также описание управляющих сигналов МП и слова состояния требуют более детального рассмотрения.

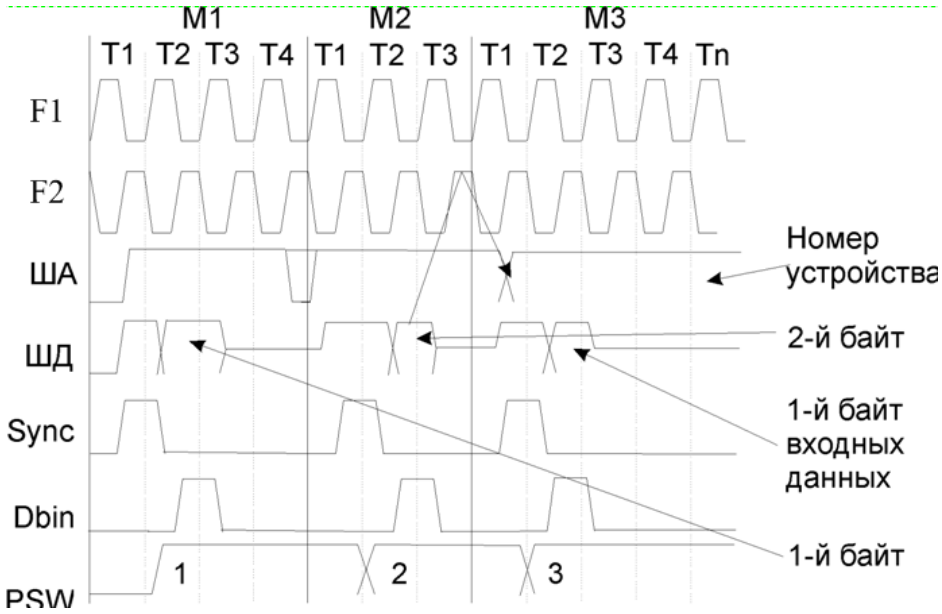


Рис. 1.10. Временная диаграмма выполнения команды ВВОД

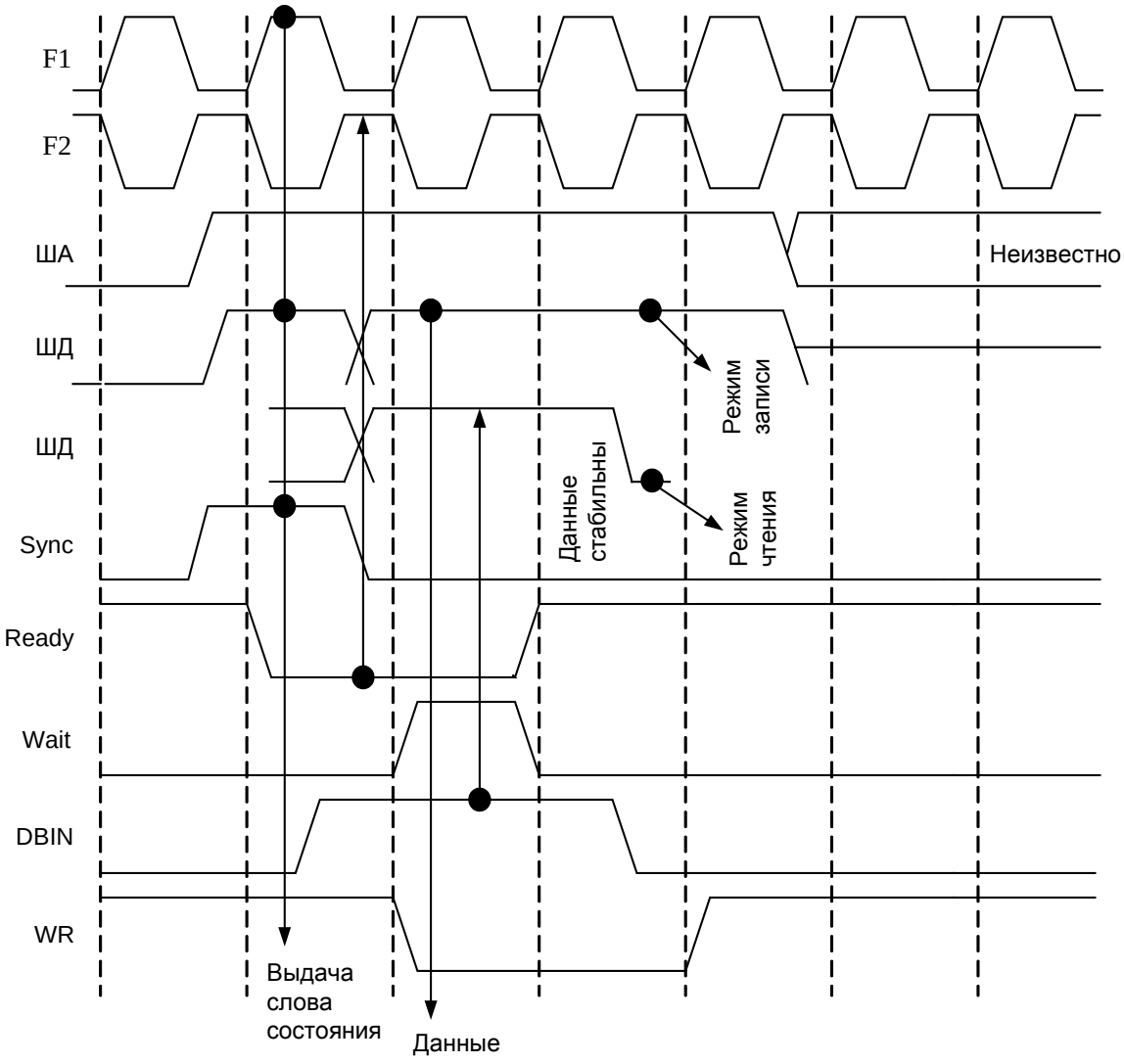


Рис. 1.11. Временная диаграмма основного цикла команды

Временная диаграмма (рис. 1.10) представляет собой основной цикл команды МП в условиях, когда присутствует внешний управляющий сигнал READY, информирующий о готовности периферийного устройства к обмену с МП (воздействие сигналов HOLD и INT будет рассмотрено ниже).

В первом такте синхронизации T1 МП выставляет на шине адреса адрес очередной команды. Начинается цикл выборки команды. Одновременно на линии синхронизации SYNC появляется единичный сигнал, который, во-первых, идентифицирует информацию на шине данных как слово состояния и загружает его во внешний регистр состояния (SL), а во-вторых, свидетельствует о начале машинного цикла. По окончании сигнала синхронизации буферная схема шины данных переводит ШД в режим ввода, о чем свидетельствует сигнал DBIN на шине управления (рис. 1.11).

В такте T2 МП осуществляет проверку готовности внешнего устройства к обмену в том случае, если адаптер внешнего устройства или памяти генерирует сигнал Ready. Первичный автомат управления переходит в состояние ожидания Wait (Tw). В состоянии Tw МП будет находиться до тех пор, пока на линии управления Ready не появится единичный сигнал, который будет свидетельствовать о том, что память или периферийное устройство готовы к обмену. Естественно, что до тех пор, пока автомат находится в состоянии Tw, МП простаивает. Поэтому на этапе проектирования МПС следует обеспечить согласование рабочей частоты МП и типа используемой памяти. Из альтернативных состояний T2 и Tw первичный автомат всегда переходит в состояние T3, в котором происходит чтение или запись данных в память. Состояния первичного автомата T4 и T5 отводятся для реализации операции, заданной кодом команды. Выполнение некоторых сложных команд может требовать от первичного автомата неоднократного перехода по циклу состояний от T1 до T5 (см. рис. 1.10). Цикл команды ввода реализуется за три машинных цикла: M1, M2, M3. В течение машинного цикла M1 производится выборка команды. После загрузки команды в регистр команды первичный автомат по коду команды определяет, что данная команда ввода/вывода двухбайтовая. Включается механизм тандемных пересылок, который влечет за собой переход первичного автомата из машинного цикла M1 к M2, т. е. из состояния T4 автомат вновь переходит в состояние T1. В машинном цикле M2 второй байт команды из памяти считывается по шине данных по сигналу разрешения ввода DBIN. Второй байт команды определяет номер внешнего устройства ввода, от которого следует получить байт входных данных. В машинном цикле M2 на шине адреса присутствует инкрементированное значение содержимого счетчика команд. В состоянии T1 третьего машинного цикла M3 первичный автомат выставляет на шине адреса номер устройства ввода, который был представ-лен во втором байте команды. Под воздействием сигнала DBIN с линии управления и сигналов управления ВУ с регистра состояния выбранное устройство ввода выдает на шину данных

байт данных, который загружается в аккумулятор МП. В начале каждого машинного цикла на регистре состояний процессора фиксируется слово состояния PSW.

1.7.6. Работа первичного управляющего автомата в режиме прерывания

Работа первичного управляющего автомата в режиме прерывания представлена на рис. 1.12. Периферийное оборудование МП-системы может запросить сообщение о прерывании текущей программы у МП путем подачи сигнала INT на вход прерывания. Сигнал прерывания INT может возникнуть в любой момент цикла команды. Обработка прерывания организована таким образом, что запрос на прерывание фиксируется на внутреннем триггере TrINT запроса прерывания только при переходе первичного автомата к циклу M1, т. е. к начальному циклу очередной команды (что свидетельствует об окончании выполнения очередной операции), и только в том случае, если программе было разрешено прерывание (внутренний триггер разрешения прерывания INTE находится в состоянии 1). Выполнение этих условий приведет к тому, что следующий машинный цикл M1 будет циклом обработки запроса на прерывание.

Машинный цикл прерывания, который начинается в такте T1 в условиях разрешенного прерывания, в основном повторяет машинный цикл выборки. В течение времени, определяемом единичным значением сигнала синхронизации SYNC, в регистр состояния выдается сигнал подтверждения прерывания INTA. Этот сигнал используется в периферийном оборудовании для инициирования процедур, определяемых прерыванием.

Отличие машинного цикла прерывания от машинного цикла выборки состоит в том, что содержимое программного счетчика PC не инкрементируется, а запоминается так, чтобы это значение могло быть восстановлено после окончания программы прерывания и возврата к прерванному процессу (программе).

Как уже отмечалось, в цикле прерывания первичный управляющий автомат использует обычный машинный цикл выборки и не производит никаких дополнительных специальных действий. В рассматриваемом случае все специальные процедуры обработки запросов на прерывание возлагаются на периферийную аппаратуру: специальный субпроцессор обработки прерываний или схему определения приоритета прерывания.

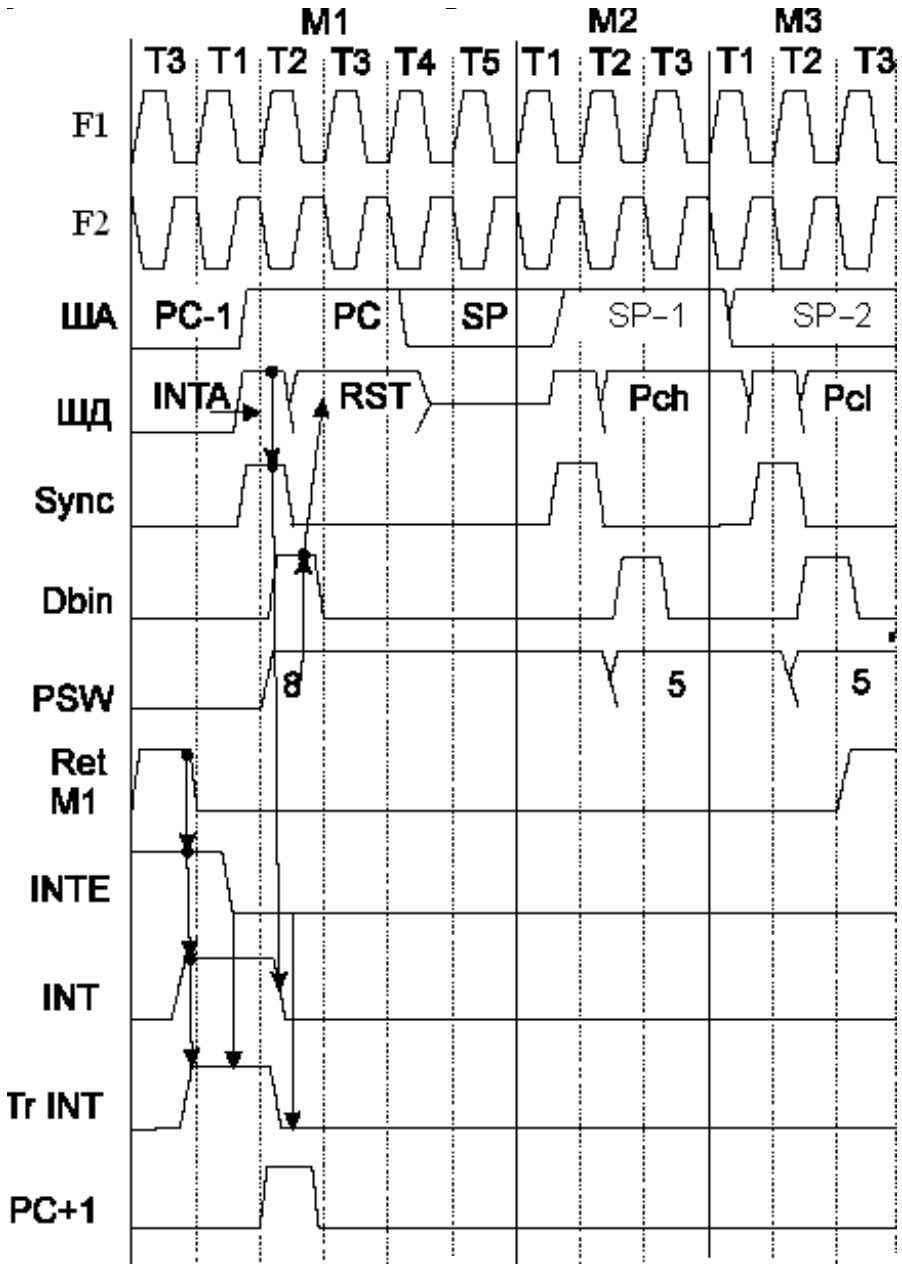


Рис. 1.12. Временная диаграмма работы МП при разрешенном прерывании

Периферийное оборудование подготавливает информацию для процессора. В такте Т3 по шине данных начальная команда программы прерывания RST (Restart), подготовленная периферийным оборудованием, помещается в МП. В МПС типовой конфигурации это означает временное отключение памяти от процессорной шины данных (на временной диаграмме это соответствует тому, что в такте Т3 управляющий сигнал WR имеет единичное значение). Информация, содержащаяся в команде RST, может быть по шине данных передана от периферии в МП, несмотря на то, что на шине адреса все еще присутствует адрес, соответствующий содержимому программного счетчика. В списке команд МП предусмотрена специальная команда вызова процедуры прерывания – команда повторного пуска RST. В формате команды содержится трехразрядное поле для представления вектора преры-

вания. В поле аппаратурой периферийных устройств или субпроцессора обработки прерываний формируется начальный адрес программы обслуживания прерывания, который соответствует приоритету устройства, запросившего прерывание.

Любая программа обслуживания прерывания начинается со специальных команд, которые перегружают содержимое программного счетчика в стек. На временной диаграмме показано, как в машинные циклы М2 и М3 (старший и младший байты) содержимое программного счетчика последовательно загружается в стек. На шине адреса в машинных циклах присутствует декрементируемое значение регистра указателя стека SP. Завершение процесса входа МПС в режим обработки прерывания фиксируется внутренним сигналом возврата к машинному циклу М1.

Особое внимание следует обратить на то, что внутренний триггер разрешения прерывания INTE процедурой входа МП в режим обработки прерывания установлен в 0. Это значит, что никакое прерывание невозможно до тех пор, пока МП не выйдет из режима обработки запроса прерывания или в подпрограмме обработки данного прерывания не будет команды разрешения прерывания с еще более высоким приоритетом.

1.7.7. Работа первичного управляющего автомата в режиме захвата шин

Управляющее устройство МП позволяет выполнять операции с прямым доступом к памяти. По линии управления HOLD периферийное устройство может приостанавливать нормальный вычислительный процесс в МП и кратковременно осуществлять управление по шинам адреса и данных. Первичный автомат при этом вырабатывает микроприказы, переводящие буферные схемы шин адреса и данных в высокоимпедансное состояние. Таким образом реализуется принцип захвата шин МП-системы для ввода/вывода данных.

В режиме ПДП обмен данными между памятью и периферией производится без участия МП. В этом случае в процедурах обмена не участвуют регистры, а следовательно, содержимое МП сохраняется неизменным.

Наиболее целесообразен режим ПДП при обмене специально упакованными блоками данных. Режим ПДП дает возможность МП-системе отказаться от программного управления обменом данными, что приводит к повышению ее эффективности, так как позволяет достаточно простыми средствами осуществить согласование работы медленно действующих периферийных устройств с высокой скоростью обработки данных в МП. На рис. 1.13 представлена временная диаграмма операции чтения с устройства ввода в режиме HOLD. Из диаграммы видим, что в случае готовности символа данных к передаче (Ready) при сигнале HOLD внутренний триггер HOLD переводится в единичное состояние. В результате первичный автомат вырабатывает по линии HLDA шины управления сигнал разрешения.

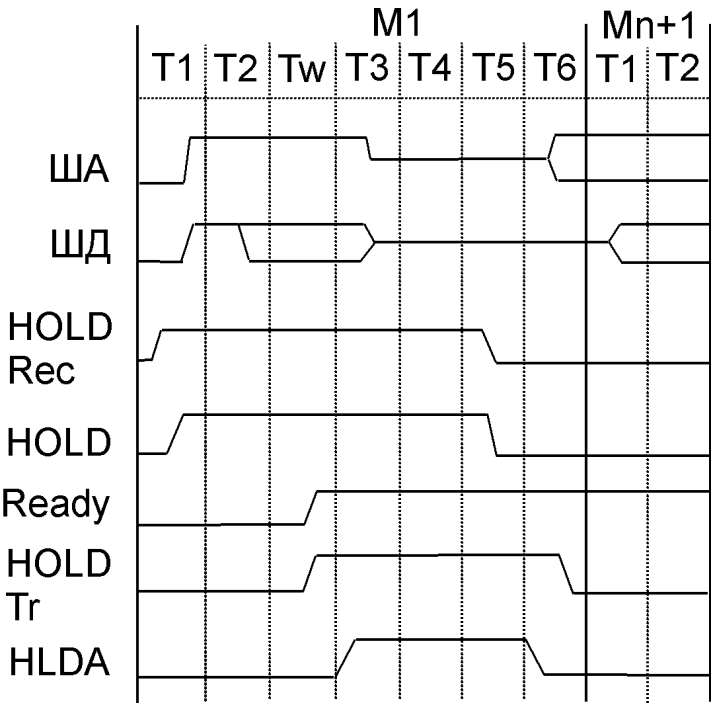


Рис. 1.13. Первичный автомат в режиме ПДП

Логика первичного автомата позволяет согласовывать во времени синхронную работу МП и асинхронные сигналы HOLD и Ready. Процесс захвата шин на один цикл для передачи символа в режиме ПДП начинается после того, как триггер HLDA установится в единичное состояние. Сигнал с единичного выхода триггера HLDA управляет буферными схемами шин адреса и данных и переводит их в высокоимпедансное состояние, т. е. отключает МП от системы по шинам адреса и данных.

Единичный сигнал HLDA, поступивший в адаптер периферийного оборудования, инициирует в нем процедуру управления системными шинами: периферийное оборудование на один цикл захватывает шины адреса и данных и передает по условленному адресу в память, минуя МП, блок информации. Естественно, что наличие режима HOLD в МП вызывает некоторое усложнение логических схем периферийных адаптеров, или контроллеров. По окончании цикла ПДП, т. е. по окончании процедуры передачи одного блока по системным шинам, периферийное оборудование снимает сигнал захвата шин HOLD, в результате чего первичный автомат сбрасывает триггер HOLD, переводит в нулевое состояние сигнал управления HLDA и возвращается к выполнению вычислительного процесса, который был приостановлен на один машинный цикл.

1.8. Методы и способы организации памяти

Общие понятия о памяти

Под памятью цифровых вычислительных устройств понимают запоминающее устройство, предназначенное для приема (записи), хранения и выдачи (считывания) информации, представленной двоичным кодом. Основными характеристиками ЗУ являются:

информационная емкость, определяемая максимальным объемом хранимой информации в битах или байтах;

быстродействие, характеризуемое временем выборки информации из ЗУ и временем цикла обращения к ЗУ с произвольным доступом или временем поиска и количеством переданной в единицу времени информации в ЗУ (или из ЗУ) с последовательным доступом;

энергопотребление, определяемое электрической мощностью, потребляемой ЗУ от источника питания в каждом из режимов работы, а также *надежность, стоимость, масса, габаритные размеры* и др.

Со времени появления больших (по размерам) компьютеров сложилось деление памяти на внутреннюю и внешнюю:

внутренняя память – электронная (полупроводниковая) память, устанавливаемая на системной плате или на платах расширения;

внешняя память – память, реализованная в виде устройств с различными принципами хранения информации и обычно с подвижными носителями.

Обычно это устройство магнитной (дисковой и ленточной) памяти, оптической и магнитооптической памяти.

Для процессора непосредственно доступной является *внутренняя (оперативная) память*.

Основная, или *оперативная, память* (Main Memory) компьютера используется для оперативного обмена информацией (командами и данными) между процессором, внешней памятью (например, дисковой) и периферийными подсистемами (графика, ввод/вывод, коммуникации и т. п.). Ее другое название – ОЗУ – примерно соответствует английскому термину RAM (Random Access Memory) – память с произвольным доступом. Произвольность доступа подразумевает возможность операций записи или чтения любой ячейкой ОЗУ в произвольном порядке.

Требования, предъявляемые к основной памяти:

большой (для электронной памяти) объем, исчисляемый единицами, десятками и даже сотнями мегабайт;

быстродействие и производительность, позволяющие реализовать вычислительную мощность современных процессоров;

высокая надежность хранения данных – ошибка даже в одном бите, в принципе, может привести и к ошибкам вычислений, и к искажению, и потере данных.

Оперативная память является одним из трех основных элементов, на которых держится «компьютерный мир» (процессор, память и периферийные устройства).

Оперативное хранение информации выполняет *динамическая память*, имеющая наилучшее сочетание объема, плотности упаковки, энергопотребления и цены. Однако ей присуще невысокое (по характеристикам современных процессоров) быстродействие. Динамическую память замещает статическая, быстродействие которой выше, но достижимая емкость принципиально ниже, чем у динамической.

Обсудим основные параметры оперативной памяти – быстродействие, производительность, достоверность хранения и методы их улучшения.

Время доступа (access time) определяется как задержка появления действительных данных на выходе памяти относительно начала цикла чтения.

Длительность цикла определяется как минимальный период следующих друг за другом обращений к памяти, причем циклы чтения и записи требуют различных затрат времени.

Производительность памяти характеризуется скоростью потока записываемых или считываемых данных и измеряется в мегабайтах в секунду. Производительность подсистемы памяти зависит от *типа* и *быстродействия* применяемых запоминающих элементов, *разрядности* шины памяти и некоторых особенностей архитектуры.

Разрядность шины памяти – это количество байт (или бит), с которыми операция чтения или записи может быть выполнена одновременно. Разрядность основной памяти обычно согласуется с разрядностью внешней шины процессора (1 байт – для I8088; 2 байта – для I8086, I80286, I386SX; 4 байта – для I386DX, I486; 8 байт – для Pentium и выше). Вполне очевидно, что при одинаковом быстродействии микросхем или модулей памяти производительность блока с большей разрядностью будет выше, чем у малоразрядного. Именно с целью повышения производительности у 32-битного (по внутренним регистрам) процессора Pentium внешняя шина, связывающая его с памятью, имеет разрядность 64 бита.

Банком памяти называют комплект микросхем, или модулей (а также их посадочных мест – «кроваток» для микросхем, слотов для SIMM или DIMM), обеспечивающий требуемую для данной системы разрядность хранимых данных. Работоспособным может быть только полностью заполненный банк. Внутри одного банка практически всегда должны применяться одинаковые (по типу и объему) элементы памяти.

Если устанавливаемый объем памяти набирается несколькими банками, появляется резерв для повышения производительности за счет применения *чередования банков* (bank interleaving). Идея чередования заключается в том, что смежные блоки данных (разрядность такого блока данных соответствует разрядности банка) располагаются поочередно в разных банках. Чем больше банков участвуют в чередовании, тем выше (теоретически) предельная производительность. Чаще всего используется чередование двух или трех банков (two way interleaving, three way interleaving).

Применение *теневой памяти* (shadow memory) позволяет повысить производительность компьютера при интенсивном обращении к относительно медленной памяти модулей ROM BIOS и видеопамяти. Идея метода заключается в «затенении» медленного модуля специальной памяти блоком быстродействующей оперативной памяти.

При использовании Shadow ROM содержимое затеняемой области (ROM) копируется в RAM и при дальнейшем обращении по этим адресам подставляется физическая область RAM, а запись в эту область блокируется. При использовании Shadow RAM запись производится одновременно в физическую память затеняемой области и в оперативную память (RAM), а при чтении обращение идет только к оперативной памяти. Shadow RAM обычно применяется для ускорения работы графических адаптеров.

Разделяемая память адаптера – память, содержимое которой может изменяться как со стороны системной шины (по инициативе процессора или других ее абонентов), так и со стороны адаптера, составной частью которого она является. Примерами разделяемой памяти являются буферная память коммуникационных адаптеров (она в произвольный момент времени может быть заполнена принятым из сети пакетом) и видеопамять адаптеров с графическими сопроцессорами (битовое разложение графического примитива строится в ней внутренним процессором графического адаптера).

Теневая память дает двойной эффект повышения производительности: затеняемые области обычно имеют малую разрядность (ROM BIOS – 8 бит, видеопамять небольшого объема – 8 или 16 бит) и низкое быстродействие (ROM имеет время доступа более 100 нс, а обращение к видеопамяти тормозится конкурирующим процессом – регенерацией изображения).

В процессорах I486 и старше для повышения производительности обмена данными с последовательно расположенными ячейками памяти введен так называемый *пакетный цикл обмена* – Burst Cycle. Обычный цикл обмена имеет фазу адреса и фазу данных. Пакетный цикл предназначен для последовательного обмена обычно с четырьмя соседними элементами (байт, слово, двойное слово,...) памяти. При этом фаза адреса существует только в начале цикла, а следующие три передачи осуществляются без нее: подразумевается автоматическое изменение адреса по определенным правилам.

В любой из многих миллионов ячеек памяти возможен случайный сбой или окончательный отказ, приводящий к ошибке. Вероятность ошибки, естественно, возрастает при увеличении объема памяти.

Отказ ячейки памяти – потеря ее работоспособности, обычно требующая замены элемента памяти. Отказ может быть устойчивым, но возможно и самопроизвольное восстановление работоспособности, например после повторного включения питания. Часто причиной отказов является неисправность контакта или нарушение условий эксплуатации.

Случайный *сбой* может произойти и в исправной микросхеме памяти, например при пролете через нее ионизирующей частицы (по этой причине в условиях высокого уровня радиации обычные электронные элементы не-

работоспособны). После сбоя следующая запись в ячейку произойдет нормально.

В первых моделях персональных компьютеров (PC) обязательно применялся контроль четности. При его использовании каждый байт памяти сопровождался битом паритета (Parity bit), дополняющим количество единиц в байте до нечетного. При обнаружении ошибки паритета схемой контроля вырабатывается немаскируемое прерывание (NMI) и его обработчик обычно выводит на экран сообщение Parity Error Check (ошибка паритета) с указанием адреса сбойной ячейки и останавливает процессор командой Halt.

В компьютерах особо ответственного применения используют память с обнаружением и исправлением ошибок – ECC Memory (Error Checking and Correcting). В этом случае для каждого записываемого информационного слова памяти (а не байта, как при контроле паритета) по определенным правилам вычисляется функция свертки, результат которой разрядностью в несколько бит также хранится в памяти. Функцию контроля и исправления выполняет чипсет, его реакцию на ошибки обычно можно задать опциями BIOS Setup. Возможны различные варианты поведения, например:

автоматически исправлять ошибки, не уведомляя об этом систему;

исправлять однократные ошибки, уведомляя систему только о многократных;

не исправлять ошибки, а только уведомлять об их обнаружении (самый достоверный контроль).

В отличие от памяти с контролем паритета, допускающей побайтное обращение, к ECC-памяти можно обращаться только полноразрядными словами.

1.9. Принципы действия ячеек памяти

1.9.1. Динамическая память

Динамическая память – DRAM (Dynamic RAM) – получила свое название от принципа действия ее запоминающих ячеек, которые выполнены в виде конденсаторов, образованных элементами полупроводниковых микросхем. С некоторым упрощением описания физических процессов можно сказать, что при записи логической единицы в ячейку конденсатор заряжается, при записи нуля – разряжается. Схема считывания разряжает через себя этот конденсатор и, если заряд был ненулевым, выставляет на своем выходе единичное значение, затем подзаряжает конденсатор до прежнего значения. При отсутствии обращения к ячейке со временем за счет токов утечки конденсатор разряжается и информация теряется, поэтому такая память требует постоянного периодического подзаряда конденсаторов (обращения к каждой

ячейке) – память может работать только в динамическом режиме. Этим она принципиально отличается от статической памяти, реализуемой на триггерных ячейках и хранящей информацию без обращений к ней сколь угодно долго (при включенном питании). Благодаря относительной простоте ячейки динамической памяти на одном кристалле удается размещать миллионы ячеек и получать самую дешевую полупроводниковую память достаточно высокого быстродействия с умеренным энергопотреблением, используемую в качестве основной памяти компьютера.

Запоминающие ячейки микросхем DRAM организованы в виде двумерной матрицы. Адрес строки и столбца передается по мультиплексированной шине адреса (MA, Multiplexed Address) и стробируется по спаду импульсов. Поскольку обращения (запись или чтение) к различным ячейкам памяти обычно происходят в случайном порядке, то для поддержания сохранности данных применяется *регенерация* (Memory Refresh – «освежение» памяти) – регулярный циклический перебор ее ячеек (обращение к ним) с холостыми циклами. Регенерация в микросхеме происходит одновременно по всей строке матрицы при обращении к любой из ее ячеек.

Динамическая память, используемая в видеобуферах графических адаптеров, специальных циклов регенерации, как правило, не требует, поскольку частота ее чтения для регенерации изображения вполне достаточна для сохранения информации.

Динамическая память в настоящее время является практически незаменимой в качестве основной (оперативной) памяти компьютеров. Наиболее частые изменения конфигурации PC связаны именно с оперативной памятью – обычно стремятся к увеличению ее объема и повышению производительности. С этим видом памяти, однако, связано и большинство проблем, выражающихся в неустойчивой работе компьютера.

Микросхемы или (и) модули динамической памяти, предназначенные для работы в качестве ОЗУ, в подавляющем большинстве случаев устанавливаются на системной плате с целью максимального приближения к процессору и чипсету. Это приближение (и физическое, и логическое) прежде всего направлено на повышение производительности оперативной памяти.

1.9.2. Статическая память

Статическая память – SRAM (Static Random Access Memory), как следует из ее названия, способна хранить информацию в статическом режиме, т. е. сколь угодно долго при отсутствии обращений (но при наличии питающего напряжения). Ячейки статической памяти реализуются на триггерах – элементах с двумя устойчивыми состояниями. По сравнению с динамической памятью эти ячейки более сложные и занимают больше места на кристалле, однако они проще в управлении и не требуют регенерации. Быстродействие и энергопотребление статической памяти определяется технологи-

ей изготовления и схемотехникой запоминающих ячеек. Самая экономичная КМОП-память (CMOS Memory) имеет время доступа более 100 наносекунд, но зато пригодна для длительного хранения информации при питании от маломощной батареи, что и применяется в памяти конфигурации PC. Самая быстродействующая статическая память имеет время доступа в несколько наносекунд, что позволяет ей работать на частоте системной шины процессора, не требуя от него тактов ожидания. Типовой объем памяти современных микросхем SRAM достигает 1 Мбит. Относительно высокая удельная стоимость хранения информации и энергопотребление при низкой плотности упаковки не позволяют использовать SRAM в качестве основной памяти компьютеров.

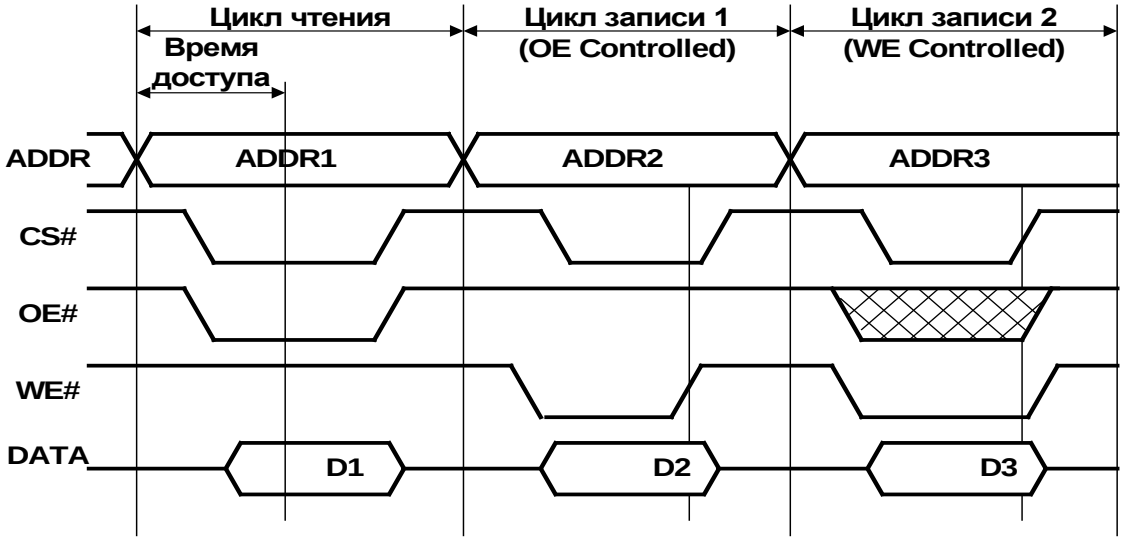


Рис. 1.14. Временные диаграммы
чтения и записи асинхронной статической памяти

Микросхемы этого типа имеют простейший асинхронный интерфейс, включающий шину адреса, шину данных и сигналы управления CS#, OE# и WE#. Микросхема выбирается низким уровнем сигнала CS# (Chip select). Низкий уровень сигнала OE# (Output Enable) открывает выходные буферы для считывания данных, WE# (Write Enable) низким уровнем разрешает запись (рис. 1.14). При операции записи управление выходными буферами может производиться как сигналом OE# (цикл 1), так и сигналом WE# (цикл 2). Для удобства объединения микросхем внутренний сигнал CS# может собираться по схеме «И» из нескольких внешних, например: CSO#, CS1 и CS2#. В таком случае микросхема будет выбрана при сочетании логических сигналов 0, 1, 0 на соответствующих входах.

Время доступа – задержка появления действительных данных на выходе относительно момента установления адреса. У стандартных микросхем SRAM составляет 12, 15 или 20 наносекунд, что позволяет процессору выполнять пакетный цикл чтения 2–1–1–1 (т. е. без тактов ожидания) на частоте

системной шины до 66 МГц. На более высоких частотах цикл будет не лучше 3–2–2–2.

1.9.3. Энергонезависимая память

Энергонезависимая память хранит записанные данные и при отсутствии питающего напряжения в отличие от статической и динамической полупроводниковой памяти. Существует множество типов энергонезависимой памяти: ROM, PROM, EPROM, EEPROM, Flash Memory, различающихся по потребительским свойствам, способам построения запоминающих ячеек и сферам применения. Запись информации в энергонезависимую память, называемую программированием, обычно существенно сложнее и требует больших затрат времени и энергии, чем считывание. Основным режимом работы такой памяти является считывание данных, а некоторые типы после программирования допускают только считывание, что и обусловливает их общее название ROM (Read Only Memory – память только для чтения) или ПЗУ (постоянное запоминающее устройство).

Существует и *полупостоянная память*, которая в основном используется для хранения информации о конфигурации компьютера. Традиционная память конфигурации вместе с часами – календарем (CMOS Memory и CMOS RTC) – имеет объем несколько десятков байт; ESCD (Extended Static Configuration Data) – область энергонезависимой памяти, используемая для конфигурирования устройств Plug & Play, имеет объем несколько килобайт. Сохранность данных полупостоянной памяти при отключении питания компьютера обеспечивается маломощной внутренней батарейкой или аккумулятором. В качестве полупостоянной применяется и *энергонезависимая память* – NV RAM (Non-Volatile RAM), которая хранит информацию и при отсутствии питания.

Запоминающие ячейки энергонезависимой памяти обычно несимметричны по своей природе и позволяют записывать только нули (реже – только единицы) в предварительно стертые (чистые) ячейки. Однократно программируемые микросхемы позволяют изменять только исходное (после изготовления) состояние ячеек. Стирание ячеек выполняется либо для всей микросхемы, либо для определенного блока, либо для одной ячейки (байта). Стирание приводит все биты стираемой области в одно состояние (обычно – во все единицы, реже – во все нули). Процедура стирания обычно существенно дольше записи. В зависимости от способа стирания различают:

микросхемы, стираемые ультрафиолетовым облучением, их обычно называют просто EPROM (Erasable PROM – стираемые микросхемы), или UV-EPROM (Ultra-Violet EPROM, УФРПЗУ);

электрически стираемые микросхемы EEPROM (Electrical Erasable PROM, ЭСПЗУ), в том числе и флэш-память.

Энергонезависимая память в основном применяется для хранения неизменяемой (или редко изменяемой) информации – системного программ-

ного обеспечения (BIOS), таблиц (например, знакогенераторов графических адаптеров), памяти конфигурации устройств (ESCD, EEPROM адаптеров). Эта информация обычно является ключевой для функционирования PC (без BIOS компьютер представляет собой только коробку с дорогими комплектующими).

Важными параметрами энергонезависимой памяти является *время хранения* и устойчивость к электромагнитным воздействиям, а для перепрограммируемой памяти еще и гарантированное *количество циклов перепрограммирования*. Энергонезависимую память, запись в которую производят при регулярной работе, называют NVRAM (Non Volatile Random Access Memory).

Постоянная память – ROM, FROM, EPROM

Масочные постоянные запоминающие устройства – ПЗУ (ROM) – имеют самое высокое быстродействие (время доступа 5–70 нс). Эти микросхемы в МПС широкого распространения не получили из-за сложности модификации содержимого.

Однократно программируемые постоянные запоминающие устройства – ППЗУ (PROM) – имеют аналогичные параметры и благодаря возможности программирования изготовителем оборудования (а не микросхем) находят более широкое применение для хранения кодов BIOS. Программирование этих микросхем осуществляется только с помощью специальных программаторов, в целевые устройства которых они устанавливаются в «кроватки» или запаиваются. Как и масочные, эти микросхемы практически нечувствительны к электромагнитным полям (в том числе и к рентгеновскому облучению), и несанкционированное изменение их содержимого в устройстве исключено (конечно, не считая отказа).

Репрограммируемые постоянные запоминающие устройства – РПЗУ (EPROM) – до недавних пор были самыми распространенными носителями BIOS как на системных платах, так и в адаптерах, а также использовались в качестве знакогенераторов.

Микросхемы EPROM тоже программируются на программаторах, но относительно простой интерфейс записи позволяет их программировать и в устройстве (но не в штатном его режиме работы, а при подключении внешнего программатора). *Стирание* микросхем осуществляется ультрафиолетовым облучением в течение нескольких минут. Специально для стирания микросхемы имеют стеклянные окошки. После программирования эти окошки заклеивают для предотвращения стирания под действием солнечного или люминесцентного облучения. Время стирания зависит от расстояния до источника облучения, его мощности и объема микросхемы (более емкие микросхемы стираются быстрее).

В PC чаще всего применяют микросхемы EPROM.

Отметим *основные свойства* EPROM:

- Стирание информации происходит сразу для всей микросхемы под воздействием облучения и занимает несколько минут. Стертые ячейки имеют единичные значения всех бит.
- Запись может производиться в любую часть микросхемы побайтно, в пределах байта можно маскировать запись отдельных бит, устанавливая им единичные значения данных.
- Защита от записи осуществляется подачей низкого (5 В) напряжения на вход Vpp в рабочем режиме (только чтение).
- Защита от стирания производится заклейкой окна.

Память с электрическим стиранием – EEPROM и флэш-память

Стирание микросхем постоянной памяти возможно и электрическим способом. Однако этот процесс требует значительного расхода энергии. Интерфейс традиционных микросхем EEPROM имеет временную диаграмму режима записи с большой длительностью импульса, что не позволяет непосредственно использовать сигнал записи системной шины. Кроме того, перед записью информации в ячейку обычно требуется предварительное стирание, тоже достаточно длительное. Микросхемы EEPROM относительно небольшого объема широко применяются в качестве энергонезависимой памяти конфигурирования различных адаптеров. Современные микросхемы EEPROM имеют более сложную внутреннюю структуру, в которую входит управляющий автомат. Это позволяет упростить внешний интерфейс, делая возможным непосредственное подключение к микропроцессорной шине, и скрыть специфические (и ненужные пользователю) вспомогательные операции типа стирания и верификации.

Флэш-память, по определению, относится к классу EEPROM, но использует особую технологию построения запоминающих ячеек. Стирание во флэш-памяти производится сразу для целой области ячеек: блоками или полностью всей микросхемы. Это позволило существенно повысить производительность в режиме записи (программирования). Флэш-память обладает сочетанием высокой плотности упаковки (ее ячейки на 30 % меньше ячеек DRAM), энергонезависимого хранения, электрического стирания и записи, низкого потребления, высокой надежности и невысокой стоимости (рис. 1.15).

Расположение выводов распространенных микросхем флэш-памяти приведено на рис. 1.16, *а*, *б*.

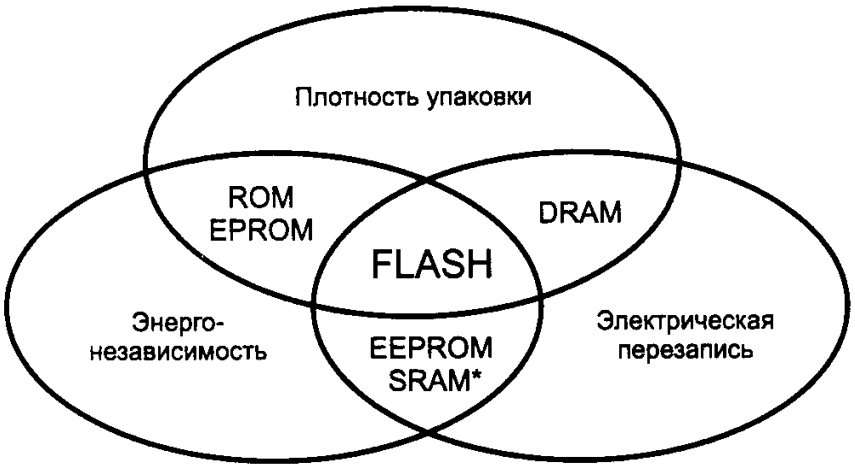


Рис. 1.15. Флэш-память – основные показатели

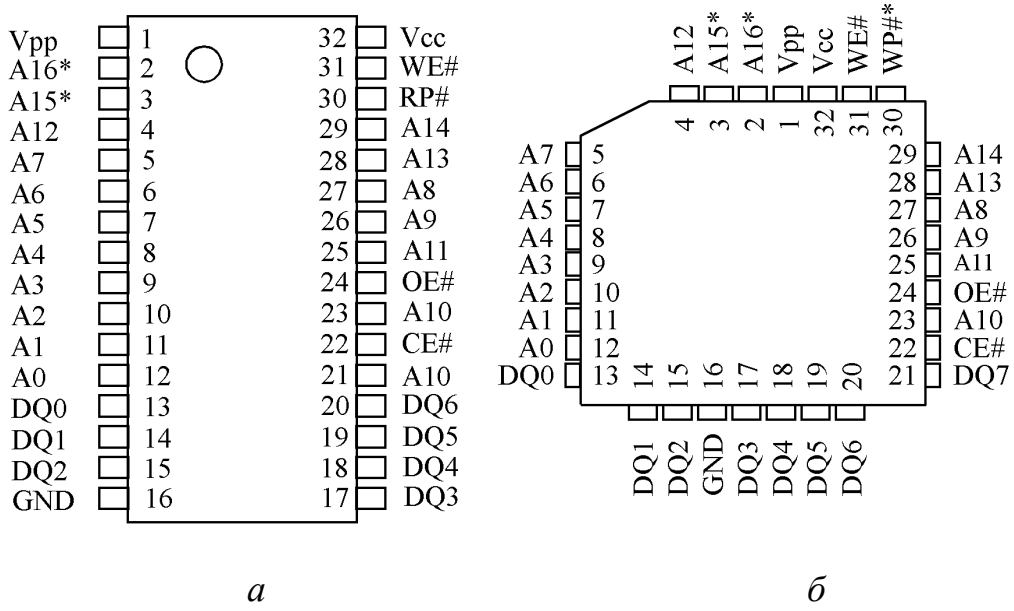


Рис. 1.16. Расположение выводов микросхем флэш-памяти
с 8-битной организацией в корпусах DIP и PLCC: *а* – DIP-32, *б* – PLCC-32

Интерфейс микросхем флэш-памяти хорошо сочетается со стандартными сигналами, используемыми в микропроцессорных системах. *Внутренние циклы* стирания, записи и верификации выполняются автономно от *шинных циклов* внешнего интерфейса, что является существенным преимуществом перед микросхемами EPROM и EEPROM. В режиме чтения они полностью совместимы с EPROM, совпадая с ними и по расположению основных выводов.

1.10. Кэширование

Кэш-память (Cache Memory) – сверхоперативная память (СОЗУ), является буфером между ОЗУ и его «клиентами» – процессором (одним или несколькими) и другими абонентами системной шины. В переводе слово *кэш* (cache) означает «склад» или «тайник» («заначка»). «Тайна» этого склада заключается в его «прозрачности» – для программы он не представляет собой дополнительной адресуемой области памяти. Кэш-память является дополнительным и быстродействующим хранилищем копий блоков информации основной памяти, к которой, вероятно, в ближайшее время будет обращение. Кэш не может хранить копию всей основной памяти, поскольку его объем во много раз меньше объема основной памяти. Он хранит лишь ограниченное количество блоков данных и *каталог* (cache directory) – список их текущего соответствия областям основной памяти. Кроме того, кэшироваться может не вся память, доступная процессору: обычно кэшируется только основная динамическая память системной платы (память, установленная на адаптерах, не кэшируется) и из этой памяти кэшируется только часть (распространенные версии чипсетов для Pentium часто позволяют кэшировать только первые 64 Мбайт ОЗУ).

Кэш в современных компьютерах строится по двух (и более)-уровневой схеме:

Первичный кэш, или *L1 Cache* (Level I Cache), – кэш 1-го уровня, внутренний (Internal, Integrated) кэш процессоров класса I486 и старше, а также некоторых моделей I386.

Вторичный кэш, или *L2 Cache* (Level 2 Cache), – кэш 2-го уровня. Обычно это *внешний* (External) кэш, установленный на системной плате. В Pentium Pro и Pentium II вторичный кэш расположен в одном корпусе с процессором. Дополнительный кэш на системную плату уже не устанавливается. Кэш, установленный на системной плате компьютера с процессором I386, не имеющим внутреннего кэша, является первичным (и единственным).

При каждом обращении к кэшируемой памяти контроллер кэш-памяти по каталогу проверяет, есть ли действительная копия затребованных данных в кэше. Если она там есть, то это случай *кэш-попадания* (cache hit), и обращение за данными происходит только к кэш-памяти. Если действительной копии там нет, то это случай *кэш-промаха* (cache miss), и данные берутся из основной памяти.

Кэш-контроллер должен обеспечивать *когерентность* (coherency) – согласованность данных кэш-памяти обоих уровней с данными в основной памяти, причем обращение к этим данным может производиться не только со стороны процессора (процессоров может быть и несколько и у каждого может быть свой внутренний кэш), но и со стороны других активных (bus-master) адаптеров, подключенных к шинам (PCI, VLB, ISA...).

Контроллер кэша оперирует *строками* (cache line) фиксированной длины. Строка может хранить копию блока основной памяти, размер которого, естественно, совпадает с длиной строки. С каждой строкой кэша связаны ин-

формация об адресе скопированного в нее блока основной памяти и признаки ее состояния. Строка может быть *действительной* (valid) – в текущий момент времени она достоверно отражает соответствующий блок основной памяти, или *недействительной* (пустой). Информация о том, какой именно блок занимает данную строку (т. е. старшая часть адреса или номер страницы), а также ее состояние называется *тегом* (tag), который хранится в связанной с данной строкой ячейке специальной *памяти тегов* (tag RAM).

Возможен и вариант *секторированного* (sectored) кэша, при котором одна строка содержит несколько смежных ячеек – *секторов,* размер которых соответствует минимальной порции обмена данных кэша с основной памятью. При этом в записи каталога, соответствующей каждой строке, должны храниться биты действительности для каждого сектора данной строки. Секторирование позволяет экономить память, необходимую для хранения каталога при увеличении объема кэша.

Поведение кэш-контроллера при операции записи в память, когда копия затребованной области находится в некоторой строке кэша, определяется его *политикой записи* (Write Policy). Существуют два основных алгоритма записи данных из кэша в основную память: *сквозная запись* WT (Write Through) и *обратная запись* WB (Write Back).

Алгоритм WT предусматривает выполнение каждой операции записи (даже однобайтной), попадающей в кэшированный блок, одновременно и в строку кэша, и в основную память.

Алгоритм WB позволяет уменьшить количество операций записи на шине основной памяти. Если блок памяти, в который должна производиться запись, отображен и в кэше, то физическая запись сначала будет произведена в эту действительную строку кэша и отмечена как *грязная* (dirty), или модифицированная, т. е. требующая выгрузки в основную память. Только после этой выгрузки (записи в основную память) строка станет *чистой* (clean) и ее можно будет использовать для кэширования других блоков без потери целостности данных.

Различают три архитектуры кэш-памяти: *кэш прямого отображения* (direct-mapped cache), *полностью ассоциативный кэш* (fully associative cache) и их комбинация – *частично* или *наборно-ассоциативный кэш* (set-associative cache).

В кэш-памяти прямого отображения адрес памяти, по которому происходит обращение, однозначно определяет строку, в которой может находиться отображение требуемого блока.

Наборно-ассоциативная архитектура кэша позволяет каждому блоку кэшируемой памяти претендовать на *одну из нескольких* строк кэша, объединенных в *набор* (set). Эту архитектуру можно рассматривать как несколько параллельно и согласованно работающих каналов прямого отображения, где контроллеру кэша приходится еще и принимать решение о том, в какую из строк набора помещать очередной блок данных.

В отличие от предыдущих у *полностью ассоциативного кэша любая* его *строка* может отображать *любой блок* памяти, что существенно повышает эффективность использования его ограниченного объема. При этом все биты ад-

реса кэшированного блока за вычетом бит, определяющих положение (смещение) данных в строке, хранятся в памяти тегов. В такой архитектуре для определения наличия затребованных данных в кэш-памяти необходимо сравнение со старшей частью адреса тегов *всех строк,* а не одной или нескольких.

1.11. Карта памяти. Критерии и способы распределения адресного пространства

Карта памяти вычислительной системы – это графическое представление распределения адресного пространства системы, разбиение таблицы доступных адресов по функциональным и программно-аппаратным признакам.

При построении микропроцессорных устройств важнейшими задачами проектировщика являются распределение адресного пространства системы; определение и распределение доступных системных адресов между ОЗУ, ПЗУ и ВУ; формирование протоколов предоставления адреса и оптимизация аппаратно-программной части обмена адресом.

В первых моделях микропроцессоров использовалось *линейное распределение* адресного пространства, при котором все доступное адресное пространство системы разбивалось по линейному принципу. В младших адресах располагались системные векторы прерываний, а прочая память делилась по принципу фон Неймана, отображая адреса программ и данных в единое адресное пространство системы, при этом выделялись адреса внешних устройств. С момента разработки процессора I8086 и по настоящее время применяется принцип *сегментированного разбиения,* при котором область программ и данных разбивается на несколько сегментов (страниц) по функциональным признакам. Так, в микропроцессорной системе, организованной на базе МП I8086, присутствуют четыре сегмента (кода, данных, стековый и дополнительный).

Позднее при увеличении доступного адресного пространства появились более сложные, но вместе с тем и более эффективные способы организации системной памяти. Так, начиная с процессора I80286, принят принцип *дескрипторной адресации,* при котором сегменты в памяти адресуются посредством дескрипторов – специальных таблиц в памяти, отвечающих за распределение и доступ к адресам системы.

Основными критериями распределения адресного пространства системы являются:

- максимально доступное адресное пространство;
- минимум аппаратных затрат на реализацию необходимых системных функций;

- максимальное быстродействие системной памяти в сочетании с контролем достоверности информации;
- обеспечение выбранной модели памяти необходимыми системными программными и аппаратными ресурсами;
- удобочитаемость карты распределения, подразумевающая иерархический, сегментированный или иной способ описания адресного пространства, при котором разработчик достаточно легко проводит анализ адресов аппаратной части системы;
- наращиваемость, возможность комбинационного проектирования адресного пространства.

Вышеперечисленные требования не отражают всех возможных критериев и задач распределения адресного пространства, которые могут возникнуть в процессе разработки микропроцессорной системы, но являются одними из основных.

Контрольные вопросы к главе 1

1. Дайте определение понятиям «автомат», «программа», «команда» и «память программ».
2. Приведите основные исторические сведения о развитии микропроцессоров.
3. Перечислите критерии классификации микропроцессоров.
4. Перечислите компоненты простейшей микропроцессорной системы, организованной по архитектуре «с тремя шинами».
5. Каковы основные принципы построения MPP- и SMP-систем?
6. Приведите общий алгоритм выполнения команды процессором.
7. Дайте определение понятиям «системная синхронизация», «машинный такт», «машинный цикл» и «цикл команды».
8. Каковы алгоритмы функционирования микропроцессорной системы в режиме прерывания и прямого доступа к памяти?
9. Приведите основные характеристики запоминающих устройств.
10. Каковы принципы функционирования динамической, статической и энергонезависимой памяти? Назовите методы и способы организации кэш-памяти.
11. Что такое карта памяти? Перечислите основные критерии и способы распределения адресного пространства вычислительных систем.

2. МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ С ДАТЧИКАМИ

2.1. Общие сведения

Датчиком (sensor) называется устройство, вырабатывающее выходной сигнал в ответ на входной электрический сигнал или механическое действие. Иначе датчиками называют *преобразователи (transducer)* одного типа сигнала в другой.

Датчики применяются для измерения различных физических свойств материалов и сред (температуры, силы, давления, позиции, интенсивности света и др). Эти входные воздействия задают возбуждение датчика, который входит в систему измерения данного параметра. Такой системой является совокупность аналоговых и (или) цифровых модулей управления/анализа каким-либо процессом.

Различают *активные* и *пассивные* датчики. *Активный датчик* использует внешние цепи возбуждения, например датчики на резисторах. Такие датчики изменяют свое сопротивление в зависимости от состояния окружающей среды датчика, но для его функционирования необходим источник тока, к которому он подключен.

Пассивные датчики могут сами формировать выходной сигнал без использования внешнего источника тока или напряжения, например фотодиоды. Фотодиод генерирует фотодиодный ток в зависимости от уровня освещенности, который не зависит от внешних цепей.

В табл. 2.1 приведены виды типичных датчиков.

Выходной сигнал датчиков, как правило, достаточно мал (миллиамперы, милливольты, пикофарады и т. п.), в связи с этим сигнал должен быть усилен для приема, оцифровки и дальнейшей обработки цифровой системой.

Цепи усиления, фильтрации, трансформации и преобразования называются *цепями формирования сигнала.*

Помимо низкого выходного сигнала выход датчика, как правило, достаточно не линеен. Другими словами, датчики далеко не всегда выдают прямо пропорциональную зависимость выходного сигнала от входного возбуждения. Таким образом, цепи формирования сигнала должны содержать модули линеаризации датчика.

Таблица 2.1

Типичные датчики

Измеряемый параметр	Наименование датчика	Активный или пассивный	Выход датчика
Температура	Термоэлемент Тиристор Резистивный термометр Термистор	Пассивный Активный Активный Активный	Напряжение Напряжение/ток Сопротивление Сопротивление
Сила/давление	Тензометр Пьезокварцевый датчик	Активный Пассивный	Сопротивление Напряжение
Ускорение	Акселерометр	Активный	Емкость
Позиция	Преобразователь перемещения	Активный	Переменное напряжение
Интенсивность света	Фотодиод	Пассивный	Ток

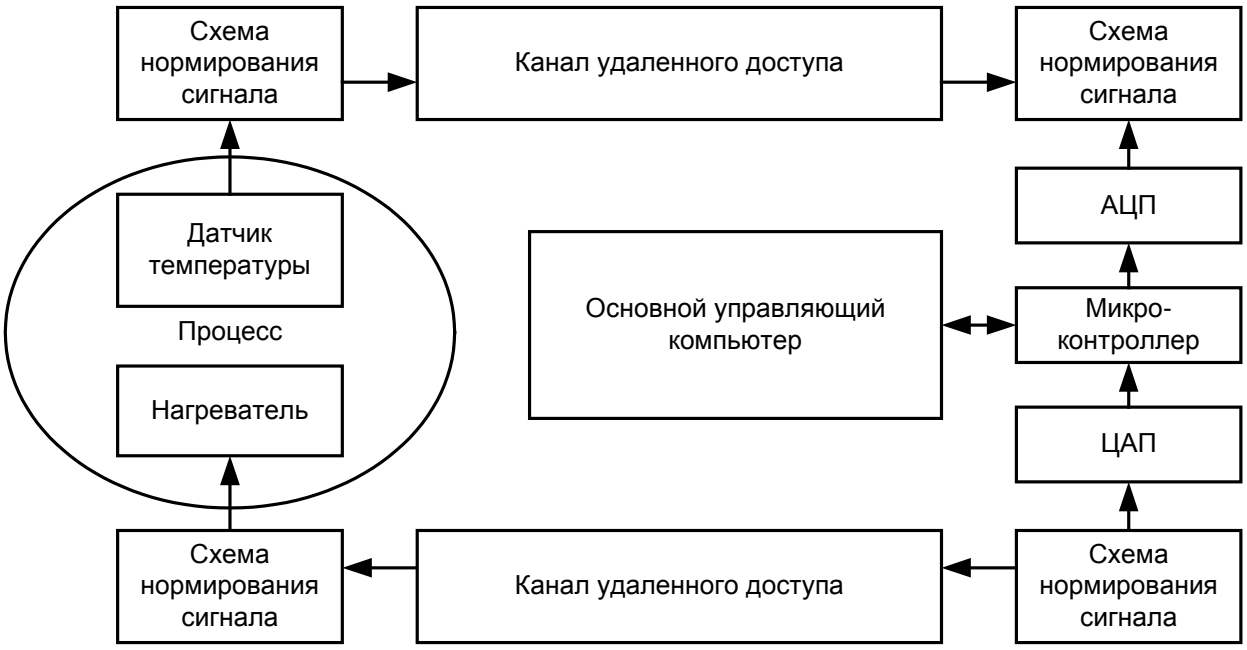


Рис. 2.1. Схема управления термопроцессом

Рассмотрим типичную схему микропроцессорной системы для анализа и контроля температуры среды некоторого процесса (рис. 2.1).

Температурный датчик помещен в исследуемую среду (процесс). Выходной сигнал датчика согласовывается (нормируется) и поступает на вход аналого-цифрового преобразователя (АЦП). Микропроцессор (микроконтроллер) управляет работой АЦП и воспринимает преобразованный в соот-

ветствующую цифровую величину сигнал, получаемый с датчика. Исполнительным устройством системы контроля температуры является нагреватель, управляемый микропроцессором при помощи цифроаналогового преобразователя и схем согласования. Основываясь на данных о температуре, микропроцессорная система поддерживает уровень приложенного напряжения (протекающего тока) через нагреватель для поддержания заданного значения температуры процесса.

Если эту или аналогичную систему объединить в один конструктив, мы получим *интеллектуальный датчик* (*smart sensor*), который имеет функции самонастройки, автолинииеризации и пр. Если расширить интеллектуальный датчик возможностями передачи данных по стандартизированной локальной сети, то мы получим *интегрированную систему сбора и обработки информации*. Такие системы выпускаются ведущими мировыми производителями (Analog Devices, Texas Instruments, Philips и др.) Они интегрируют на одном конструктиве (или даже кристалле) высокопроизводительные АЦП, ЦАП, микроконтроллеры, flash-память, различные стандартные контроллеры последовательной передачи данных и др.

2.2. Резистивные датчики

Наиболее распространенными датчиками являются резистивные элементы. В табл. 2.2 приведены различные резистивные датчики.

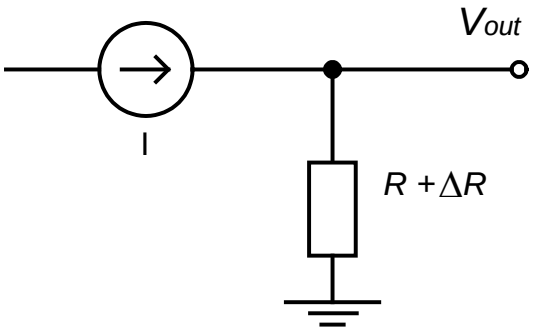
Таблица 2.2

Резистивные датчики

Наименование датчика	Диапазон сопротивлений
Тензометрические датчики	120 Ом, 350 Ом, 3500 Ом
Динамометры	350–3500 Ом
Датчики давления	350–3500 Ом
Датчики относительной влажности	100 кОм – 10 МОм
Термометры сопротивления	100–1000 Ом
Термисторы	100 Ом – 10 МОм

Как видно из таблицы, первые три позиции имеют малый диапазон измерения сопротивления, следовательно, при использовании этих датчиков особенно необходимо очень точно отслеживать малейшие изменения сопротивления.

Обычная схема подключение резистивного датчика приведена на рис. 2.2.

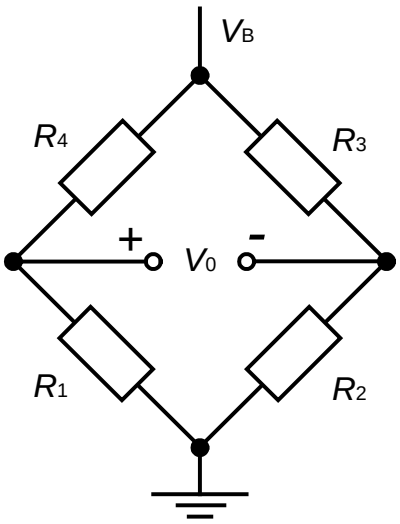


$$V_{out} = I(R + \Delta R)$$

Рис. 2.2. Резистивный датчик

Следует заметить, что при данном включении необходимо учитывать нелинейность как самого датчика, возникающую при его самонагреве, так и возможную нестабильность работы источника тока. Для избежания этих искажений рекомендуется использовать источник тока малой величины.

Более сложными, но более информативными и распространенными схемами включения резистивных датчиков являются различные *мостовые* схемы, например мост Уитстона (рис. 2.3).



$$V_{out} = \frac{R_1}{R_1 + R_4} V_в - \frac{R_2}{R_2 + R_3} V_в = \frac{\frac{R_1}{R_4} - \frac{R_2}{R_3}}{\left(1 + \frac{R_1}{R_4}\right)\left(1 + \frac{R_2}{R_3}\right)} V_в$$

$$V_0 = 0, \text{если} \quad \frac{R_1}{R_4} = \frac{R_2}{R_3}$$

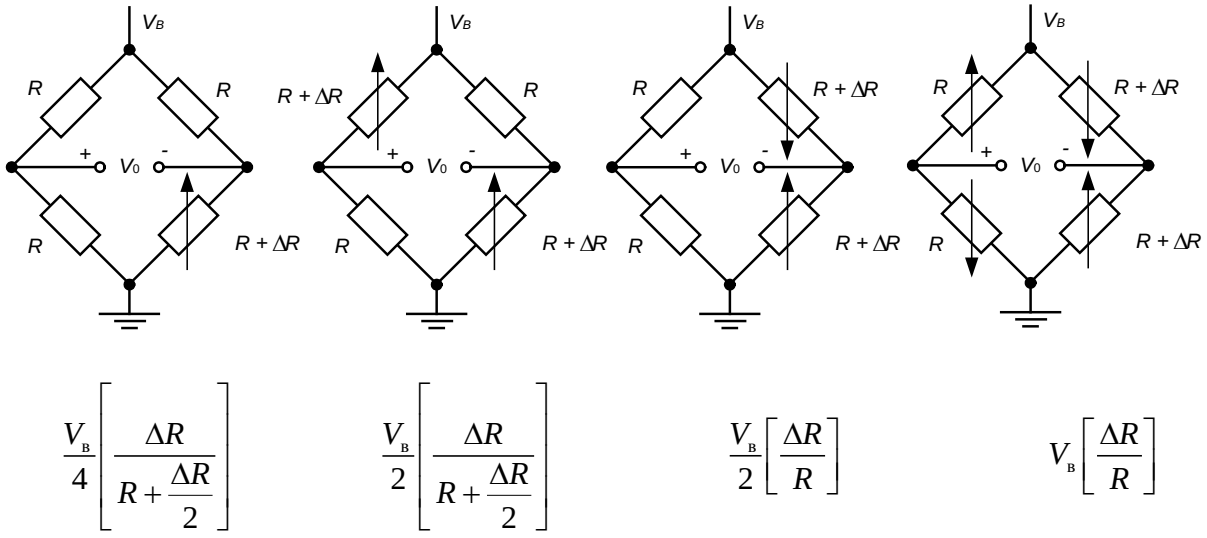
Рис. 2.3. Мост Уитстона

Если плечи моста равны ($R_1/R_4 = R_2/R_3$), то такой мост называют *сбалансированным* (нулевым).

Пусть R_1 – датчик, определим $R_2/R_3 = K$, при этом будем механически вводить мост в нулевое положение при помощи подстроечного резистора R_4, который имеет шкалу (например, реостат), таким образом получим систему управления/подстойки параметров датчика R_1. Такую систему можно использовать, например, для определения уровня поднятия заслонки при подаче жидкостей через электромеханический вентиль.

В мостовых схемах можно применять более одного датчика и определять их сопротивление по напряжению диагонали моста (рис. 2.4). Как указывалось выше, следует учитывать, что относительное изменение выходного напряжения такого моста будет достаточно невелико (десятки милливольт при $V_в$ = 10 В).

Чувствительность моста – это отношение максимально ожидаемого изменения выходного напряжения (выхода) к напряжению возбуждения (возбуждению). Так, если максимальный выход составляет 10 мВ, а возбуждение 10 В, то чувствительность равна 1 мВ/В.

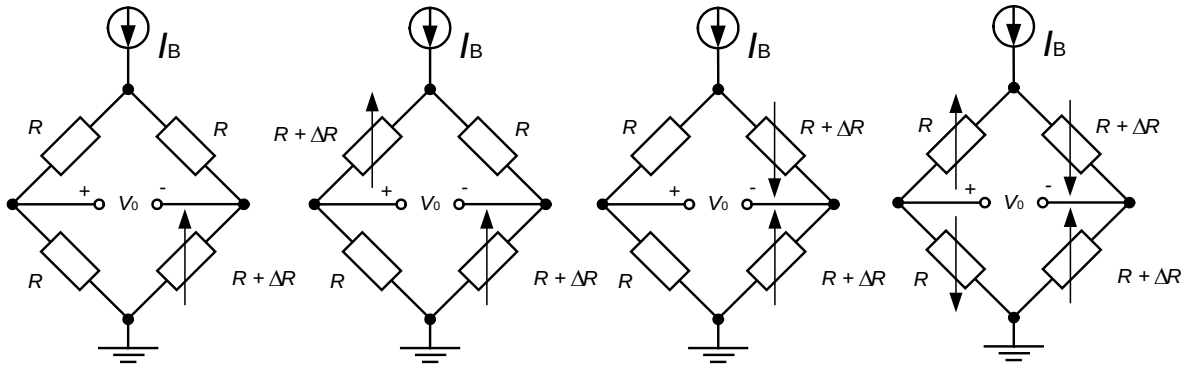


$$\frac{V_в}{4}\left[\frac{\Delta R}{R+\frac{\Delta R}{2}}\right] \qquad \frac{V_в}{2}\left[\frac{\Delta R}{R+\frac{\Delta R}{2}}\right] \qquad \frac{V_в}{2}\left[\frac{\Delta R}{R}\right] \qquad V_в\left[\frac{\Delta R}{R}\right]$$

Рис. 2.4. Конфигурации мостов (возбуждение напряжением)

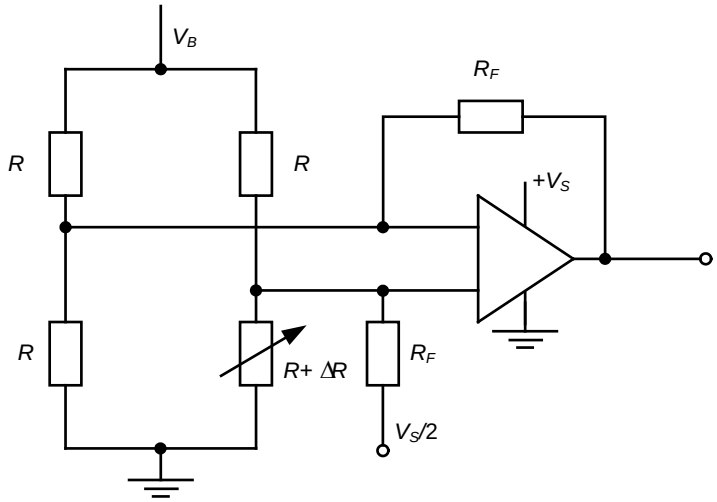
Если мост располагается достаточно далеко от системы принятия сигнала, то за счет сопротивления связывающих их проводником могут возникнуть дополнительные искажения. В этом случае мосты лучше питать не постоянным напряжением, а током (рис. 2.5), тогда нелинейным будет только четверть моста.

В качестве схемы усиления выходных сигналов моста рассмотрим схему, предложенную на рис. 2.6. Усиление сигнала происходит операционным усилителем (ОУ). Такая схема требует использования высокоточных резисторов R_F для обеспечения высокого коэффициента ослабления синфазной составляющей сигнала (КОСС). Выход схемы не линеен. Однако она достаточно проста и в ней применен однополярный источник питания.



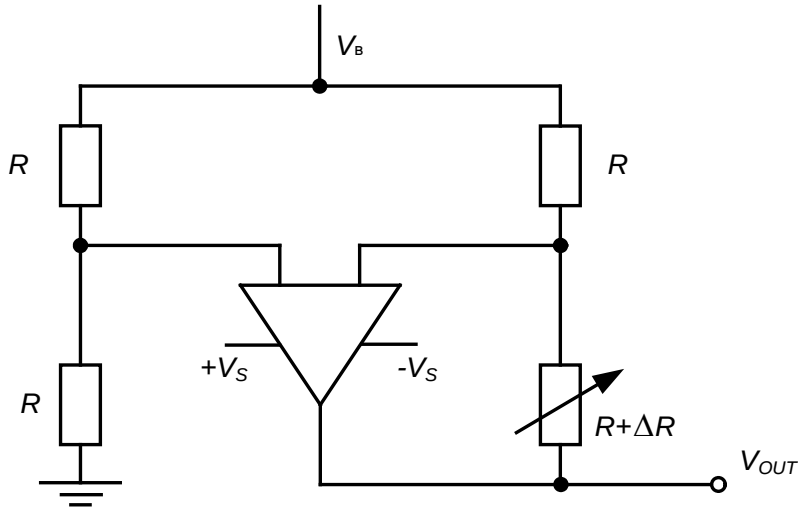
$$\frac{I_в R}{4}\left[\frac{\Delta R}{R+\frac{\Delta R}{4}}\right] \qquad \frac{I_в}{2}\Delta R \qquad \frac{I_в}{2}\Delta R \qquad I_в \Delta R$$

Рис. 2.5. Конфигурации мостов (возбуждение током)



$$V_{out} = \frac{V_в}{4}\left[\frac{\Delta R}{R+\frac{\Delta R}{2}}\right]$$

Рис. 2.6. Усиление выходного сигнала выхода четвертьмостового датчика



$$V_{out} = -V_{в}\left[\frac{\Delta R}{2R}\right]$$

Рис. 2.7. Линеаризация четвертьмостового датчика

В качестве схемы линеаризации выходного сигнала рассмотрим схему, представленную на рис. 2.7. Здесь операционный усилитель принудительно устанавливает нуль в измерительной диагонали путем подачи компенсирующего напряжения обратной полярности в измеряющее плечо моста. При этом амплитуда выходного сигнала мостового датчика в два раза больше, чем при стандартном включении, и сигнал линеен даже при большом изменении величины чувствительного элемента (ΔR). Однако в данной схеме используется двухполярный источник питания.

2.3. Тензометрические датчики

Тензодатчики применяются для измерения величины силы (давления).

Различают резистивные, полупроводниковые и пьезоэлектрические тензометрические датчики. Измерение силы тензодатчиком происходит косвенно – путем измерения деформации калиброванного элемента, вызванной действием данной силы. Для измерения давления его преобразуют в силу и измеряют тензометрическим методом. Тензодатчики также применяют для измерения скоростей потока различных жидкостей и сред (воздуха, газов). Такие измерения производят, используя дифференциальный метод измерения.

В табл. 2.3 приведены направления и техника применения тензодатчиков.

Таблица 2.3

Применение тензодатчиков

Направление	Тензодатчик
Деформация	Тензодатчик, пьезоэлектрический преобразователь
Сила	Элемент нагрузки (динамометр)
Давление	Диафрагма преобразует в силу, измеряемую тензодатчиком
Поток	Методы измерения дифференциального давления

Резистивный тензодатчик меняет свои размеры при действии на него силы (сжимается или растягивается), таким образом он изменяет свое сопротивление.

В простейшем случае резистивный тезодатчик представляет собой тензопроволочку, натянутую между двумя стойками-контактами (рис. 2.8). Сила, воздействуя на проволочку (площадью сечения A, длиной L, удельным сопротивлением ρ), вызовет удлинение или сжатие последней, что приведет к пропорциональному увеличению или уменьшению ее сопротивления:

$$R = \frac{\rho L}{A},$$

$$\frac{\Delta R}{R} = GF\frac{\Delta L}{L},$$

где GF – это тензочувствительность (значение 2.0–4.5 – для металлов, более 150 – для полупроводников).

Значение силы, приложенной к проволочке, определяется величиной $\Delta L/L$, выражается в единицах относительной деформации (е. о. д.). Таким образом, чем больше тензочувствительность, тем больше величина изменения сопротивления и, следовательно, выше чувствительность датчика.

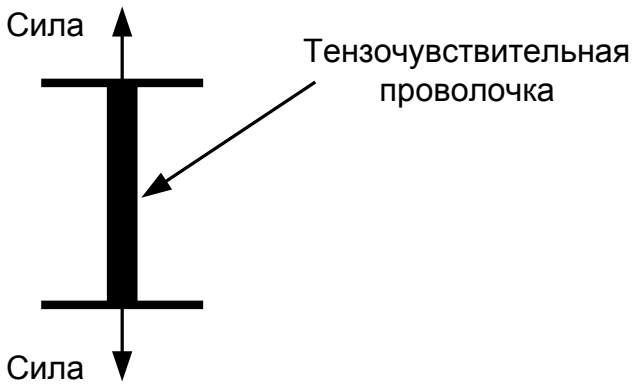


Рис. 2.8. Проволочный тензодатчик

Если проволочку или проводящую тензофольгу закрепить на специальной подвижной пластине или основании, мы получим наклеиваемый тензодатчик. Такой датчик устанавливается вдоль направления измеряемой силы (рис. 2.9).

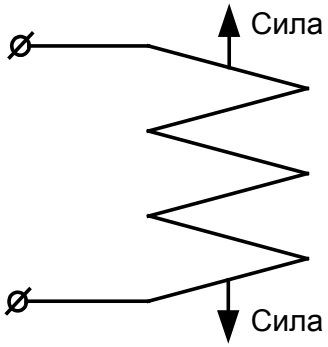


Рис. 2.9. Наклеиваемый проволочный тензодатчик

Наклеиваемые датчики изготовляют из тех же металлов, что и проволочные (константан, нихром, сплав никеля с железом и т. д.). Наибольшее распространение получили фольговые датчики, изготавливаемые методом фототравления.

Таблица 2.4

Сравнение металлических и полупроводниковых тензодатчиков

Параметр	Металлический тензодатчик	Полупроводниковый тензодатчик
Диапазон измерения	0.1–40,000 με	0.001–3000 με
Тензочувствительность	2.0–4.5	50–200
Сопротивление, Ом	120, 350, 600, ..., 5000	1000–5000
Допуск резисторов	0. 1 %–0.2 %	1–2 %
Размер, мм	0.4–150 (стандарт 3–6)	1–5

Проволочные датчики имеют малую поверхность связи с образцом (основанием), что уменьшает токи утечки при высоких температурах и дает большее напряжение изоляции между чувствительным элементом и образцом. Кроме того, фольговые чувствительные элементы имеют большое отношение площади поверхности к площади поперечного сечения (чувствительность) и более стабильны при критических температурах и длительных нагрузках. Большая площадь поверхности и малое поперечное сечение также обеспечивают хороший температурный контакт чувствительного элемента с образцом, что уменьшает саморазогрев датчика.

Полупроводниковые материалы (например, кремний и германий), имеющие пьезорезистивный эффект, используют для изготовления тензодатчиков большой чувствительности. Однако они трудно поддаются компенсации и имеют нелинейное изменение сопротивления (табл. 2.4).

2.4. Применение тензодатчиков для измерения силы

Рассмотрим применение тензодатчиков для измерения силы (рис. 2.10). В рассматриваемом примере используются четыре тензорезистора, устновленных на балке и включенных по полномостовой схеме.

Как правило, тензодатчики – это устройства с достаточно низким сопротивлением, поэтому для получения приемлемых уровней выходного напряжения они требуют приложения значительной мощности возбуждения. Пусть тензомостовой датчик имеет импеданс 350 Ом, тогда его чувствительность будет выражаться в милливольтах полной шкалы на вольт напряжения возбуждения. В качестве примера рассмотрим динамометр, состоящий из четырех тензорезисторов (рис. 2.11).

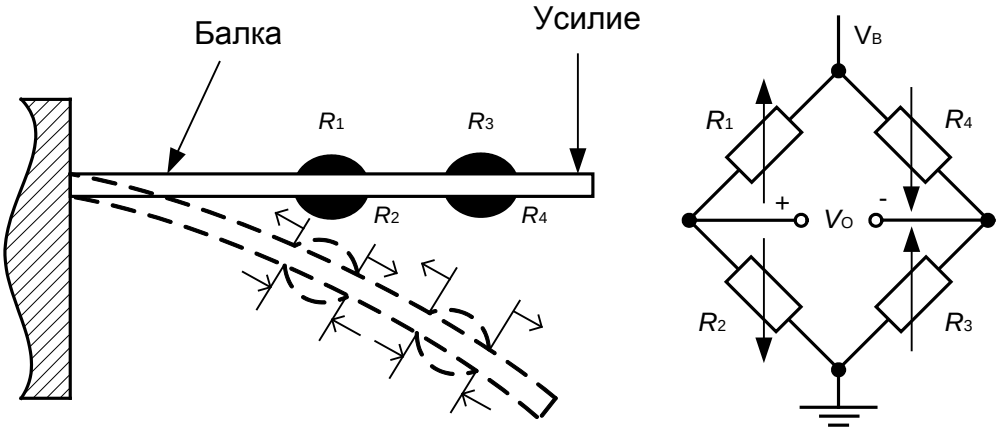


Рис. 2.10. Балочный динамометр

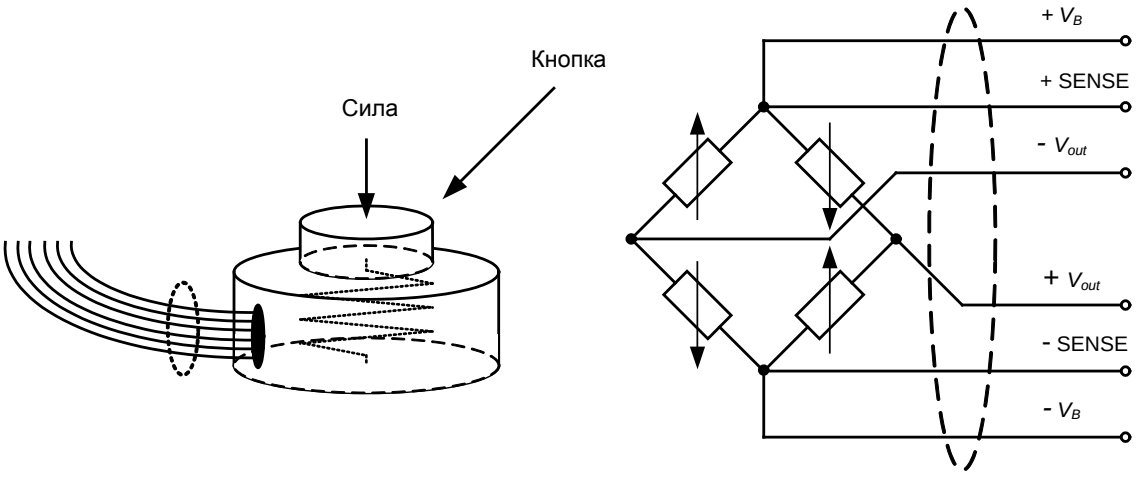


Рис. 2.11. Шестипроводной динамометр

В данной ситуации при возбуждении 10 В и коэффициенте преобразования 3 мВ/В верхний предел шкалы составит 30 мВ.

Для измерения давления в жидкостях и газах используются различные механические преобразователи давления, подключенные к тензодатчикам.

Для высокочастотных измерений давления (например, в гидроаккустике) применяются пьезоэлектрические преобразователи давления.

2.5. Измерение потоков жидкостей и газов

При измерении потоков жидкостей и газов наиболее часто измеряют *количество* протекающего вещества. Самым простым способом такого измерения является измерение объема. Однако данный метод возможен, только если плотность жидкости или газа постоянна.

Поток можно измерить, узнав дифференциальное давление между двумя точками в протекающей среде: одна статическая, другая в потоке [2].

Для таких измерений используют *Трубки Пито* (рис. 2.12) или устройства, основанные на эффекте *вентури* (эффект заключается в том, что на пути потока помещают сужающее устройство).

Можно поместить в поток изгибающуюся лопасть с закрепленным на ней тензодатчиком для измерения скорости потока (рис. 2.13).

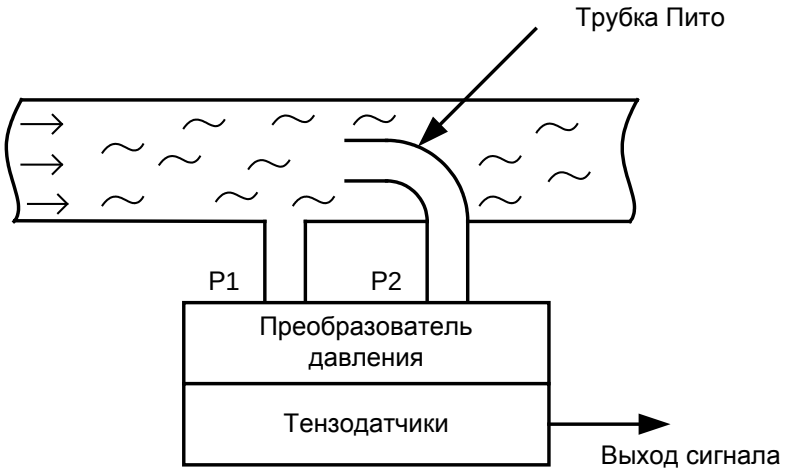


Рис. 2.12. Трубка Пито, используемая для измерения скорости потока

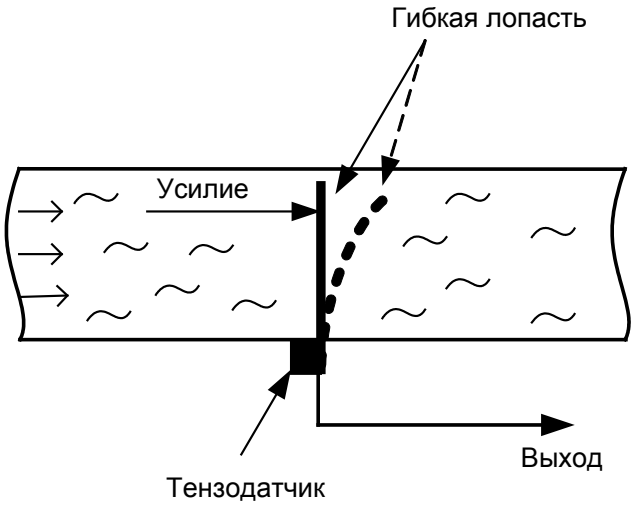


Рис. 2.13. Использование сгибающейся лопасти с тензодатчиком для определения скорости потока

Ниже приведено несколько примеров применения тензодатчиков, предложенных в [2].

2.6. Измерение деформации

Полномостовая цепь для измерения деформации при испытании материала на усталость показана на рис. 2.14. Мост является интегральным устройством и может быть закреплен на поверхности, деформацию или изгиб которой необходимо измерить. В схеме используется генератор тока возбуждения для выполнения дистанционных измерений. Устройство OP177 питает мост током 10 мА, для чего используется источник опорного напряжения 1.235 В. Тензодатчик дает выходной сигнал 10.25 мВ/1000 е. о. д. Сигнал усиливается инструментальным усилителем AD620 с коэффициентом усиления 100. Величину напряжения верхнего предела (полной шкалы) можно устанавливать, подстраивая потенциометр 100 Ом так, чтобы для деформации 3500 е. о. д. выход составлял 3.500 В, а для деформации +5000 е. о. д. – +5.000 В. Далее сигнал можно преобразовать с помощью АЦП с верхним пределом по входу 10 В. Конденсатор 0.1 мкФ на входе инструментального усилителя совместно с сопротивлением моста 1 кОм составляют низкочастотный фильтр для радиочастотных помех. Частота среза НЧ-фильтра около 1.6 КГц.

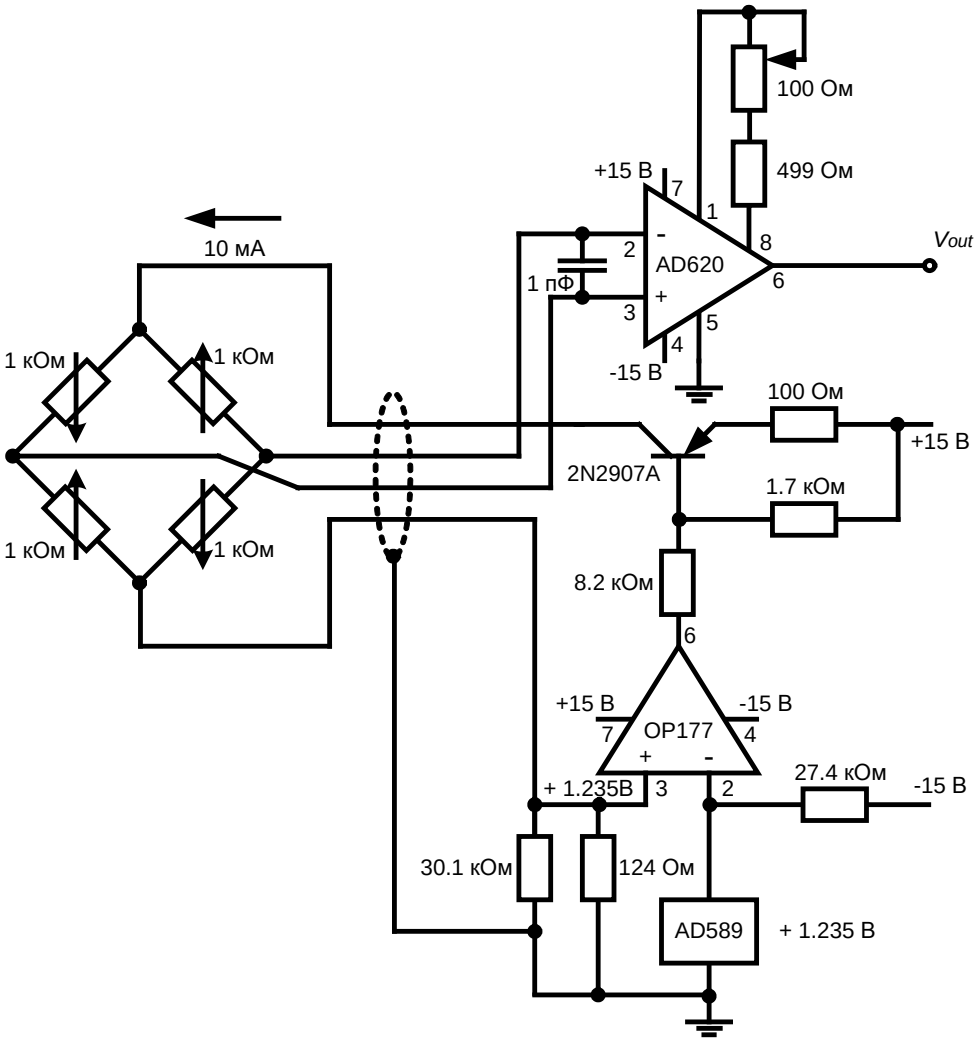


Рис. 2.14. Прецизионный усилитель для тензометрического датчика

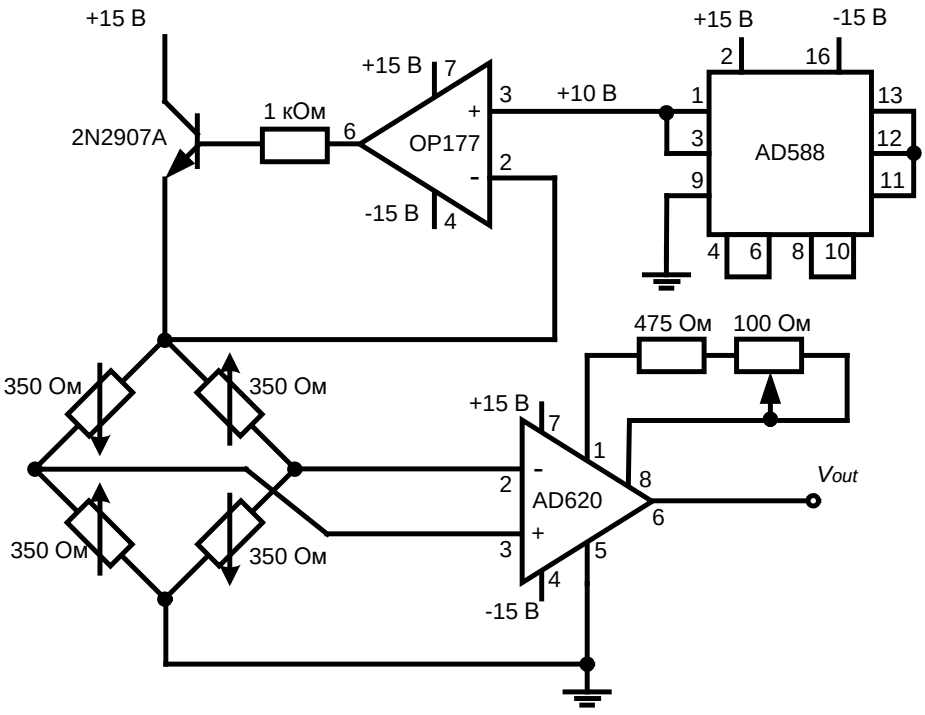


Рис. 2.15. Прецизионный усилитель для динамометра

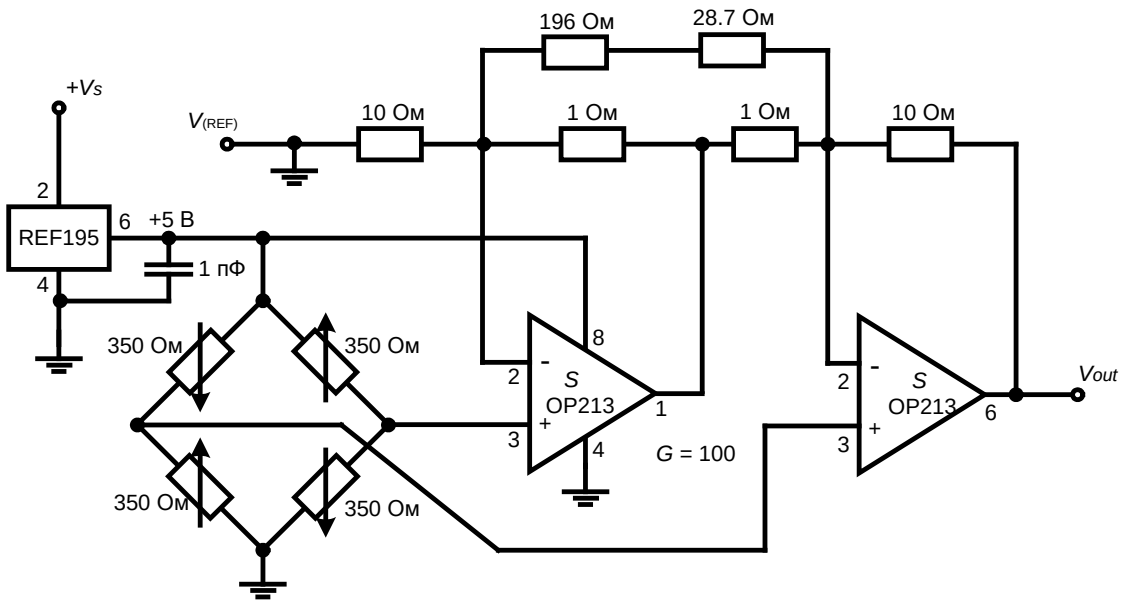


Рис. 2.16. Усилитель с однополярным питанием для элемента нагрузки

На рис. 2.15 показан другой пример цепи – усилитель динамометра (элемента нагрузки). Типовое сопротивление моста 350 Ом. 10.000 В возбуждение моста получают с помощью источника опорного напряжения на AD588, OP177 и транзистора 2N2219A, обеспечивающего ток 28.57 мА. Для сохранения высокой линейности используется инструментальный усилитель. Схема содержит минимальное количество критичных резисторов и усилителей, что обеспечивает точность, стабильность и малую стоимость. Единственным требованием является низкий температурный коэффициент резистора 475 Ом и потенциометра 100 Ом для обеспечения низкого температурного дрейфа.

Как отмечалось ранее, прецизионный динамометр обычно представляет собой измерительный мост 350 Ом. На рис. 2.16 показан прецизионный усилитель динамометра с однополярным питанием. Прецизионный пятивольтный ИОН REF195 с высокой нагрузочной способностью (30 мА) используется для питания моста.

Сдвоенный операционный усилитель OP213 образует ИУ на двух ОУ с коэффициентом усиления 100. Усиление задается резисторами:

$$G = 1 + \frac{10\ \text{кОм}}{1R\ \text{Ом}} + \frac{20\,\text{кОм}}{196\,\text{Ом} + 28.7\ \text{Ом}} = 100.$$

2.7. Датчики температуры

2.7.1. Общие сведения

Различают следующие датчики и направления при построении температурных контрольно-измерительных устройств:

для мониторирования (наблюдения)

- портативного оборудования,
- температуры центрального процессора,
- температуры аккумуляторной батареи,
- температуры окружающей среды;

для компенсации

- дрейфа генератора в сотовых телефонах,
- температуры холодного спая термопар;

для управления

- зарядом аккумуляторной батареи,
- процессом удержания температуры.

Практически все температурные датчики достаточно нелинейны, исключение составляют интегральные датчики. Резистивные датчики достаточно точны, но требуют внешнего тока возбуждения и, следовательно, оптимальной схемой включения такого датчика будет мостовая схема. Термисторы наиболее чувствительны, но и наиболее нелинейны. Полупроводниковые датчики температуры являются самыми точными, но имеют узкий диапазон применения (от –55 °С до +150 °С).

Таблица 2.5

Типы датчиков температуры

Термопары	РДТ	Термисторы	Полупроводниковые датчики температуры
Самый широкий диапазон температур (от –184 °С до +2300 °С) Высокая точность и повторяемость Необходимость компенсации холодного спая Низкое выходное напряжение	Диапазон от –200 °С до +850 °С Высокая линейность Требует внешнего возбуждения Низкая стоимость	Диапазон от 0 °С до +100 °С Низкая линейность Требует внешнего возбуждения Высокая чувствительность	Диапазон от –55 °С до +150 °С Линейность 1 °С Точность 1 °С Требует внешнего возбуждения Типовой выходной сигнал 10 мВ/К, 20 мВ/К или 1 мА/К

В табл. 2.5 приведены наиболее популярные типы температурных датчиков.

2.7.2. Термопары и компенсация холодного спая

Термопары являются относительно недорогими датчиками, причем они функционируют в широком диапазоне температур, а при измерении высоких температур (до + 2300 °С) и в агрессивных средах термопары практически незаменимы. Тем не менее они дают на выходе милливольтные сигналы и требуют точного усиления для проведения дальнейшей их обработки. Еще одним недостатком при применении термопар является компенсация температуры холодного спая (см. ниже). Как правило, термопары достаточно линейны. Наиболее известные термопары приведены в табл. 2.6, а на рис. 2.17 показаны кривые зависимости напряжения от температуры для трех широко используемых термопар.

Термопары изготавливают из следующих металлов: железо, платина, родий, рений, вольфрам, медь, алюмель (сплав никеля и алюминия), хромел (сплав никеля и хрома) и константан (сплав меди и никеля).

Рассмотрим основы функционирования термопар. Известно, что при соединении двух разнородных металлов при температуре выше абсолютного нуля между ними появляется разность потенциалов (термоЭДС), которая является функцией температуры спая (соединения) (рис. 2.17, *а*). Другими словами, *каждая пара разнородных металлов, находящихся в контакте друг с другом, генерирует термоэлектрическую ЭДС.*

Таблица 2.6

Термопары

Материал контакта	Типовой температурный диапазон, °С	Номинальная чувствительность, мкВ/°С	Обозначение по ANSI
Платина (6 %) Родий-платина (30 %) Родий	От 38 до 1800	7.7	*B*
Вольфрам (5 %) Рений-вольфрам (26 %) Рений	От 0 до 2300	16	C
Хромел-константан	От 0 до 982	76	E
Железо-константан	От 0 до 760	55	J
Хромел-алюмель	От –184 до 1260	39	K
Платина (13 %) Родий-платина	От 0 до 1593	11,7	R
Платина (10 %) Родий-платина	От 0 до 1538	10,4	S
Медь-константан	От 184 до 400	45	T

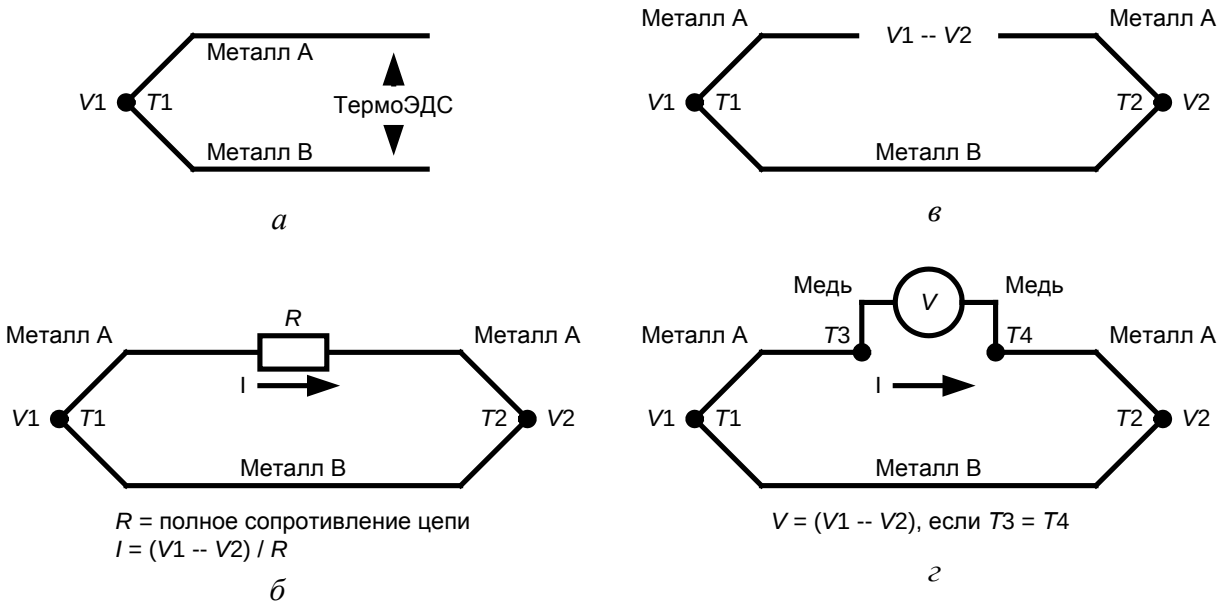


Рис. 2.17. Основы работы термопары: *а* – термоэлектрическое напряжение; *б* – термопара; *в* – измерение с помощью термопары; *г* – измерение с помощью термопары

Для того чтобы сформировать два спая (рис. 2.17, *б*), соединим между собой два термопарных провода с обоих концов. Если оба спая находятся при различных температурах, то в цепи появится результирующая ЭДС и потечет ток, определяемый величиной ЭДС и полным сопротивлением цепи. Если разорвать один из проводов, то напряжение в точках разрыва будет равно величине результирующей термоЭДС в цепи, и если измерить это напряжение, то можно использовать его для расчета разности температур двух спаев (рис. 2.17, *в*) [8].

Примечание. Термопара измеряет разницу температур двух спаев, а не абсолютную температуру одного из спаев.

Замер температуры на измерительном спае можно проводить только в том случае, если известна температура другого спая (называемого часто опорным или холодным спаем).

Подключим вольтметр к цепи термопар (рис. 2.17, *г*). В местах подключения образуются дополнительные термопары из проводов вольтметра и проводов цепи. Если эти подключения находятся при разных температурах, то они будут вносить ошибки.

Следовательно, необходимо, чтобы все пары контактов в цепи, содержащей термопару, находились при одной и той же температуре, кроме, разумеется, измерительных контактов термопар.

Термопары не требуют внешнего возбуждения. Как правило, для измерения используются два спая (рис. 2.18): $T1$ – измерительный спай и $T2$ – опорный (холодный) спай. Если $T1 = T2$, то $V1 = V2$ и выходное напряжение $V = 0$. Выходные напряжения термопар определяются по отношению к температуре опорного спая при 0 °С (см. рис. 2.18).

Именно отсюда произошел термин *холодный спай или спай точки таяния льда.* Из вышеизложенного следует, что термопара дает выходное напряжение 0 В при температуре измерительного спая 0 °С. Однако необязательно иметь температуру холодного спая, равную 0 °С. Достаточно знать его текущую температуру (рис. 2.19). На рисунке показано, что вместо ванны таящего льда используется другой температурный датчик, который измеряет температуру холодного спая, и его сигнал используется для введения напряжения в измерительную цепь термопары. Этот сигнал компенсирует разницу между действительной температурой холодного спая и ее идеальной величиной (0 °С).

На практике используют компесационные коэффициенты термопар для определения величины выходного напряжения ($V = K \cdot T2$). Таким образом, расчет напряжения термопары с температурой ее измерительного спая T °С и опорного спая при температуре 0 °С производится при помощи полинома $V = K1 \cdot T + K2 \cdot -T^2 + K3 \cdot T^3 + \ldots$. Следует учесть, что величины коэффициентов $K2$, $K3$ и т. д. весьма малы для большинства известных типов термопар.

Как правило, для компенсации холодного спая свободные концы термопары устанавливаются в специальном изотермическом блоке (рис. 2.20).

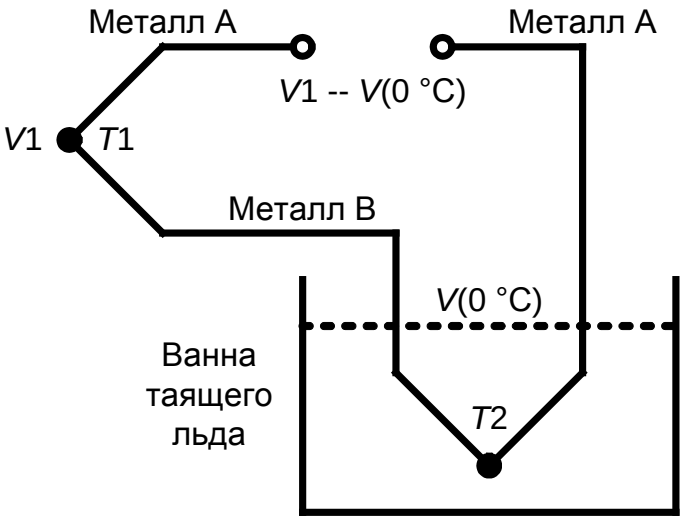
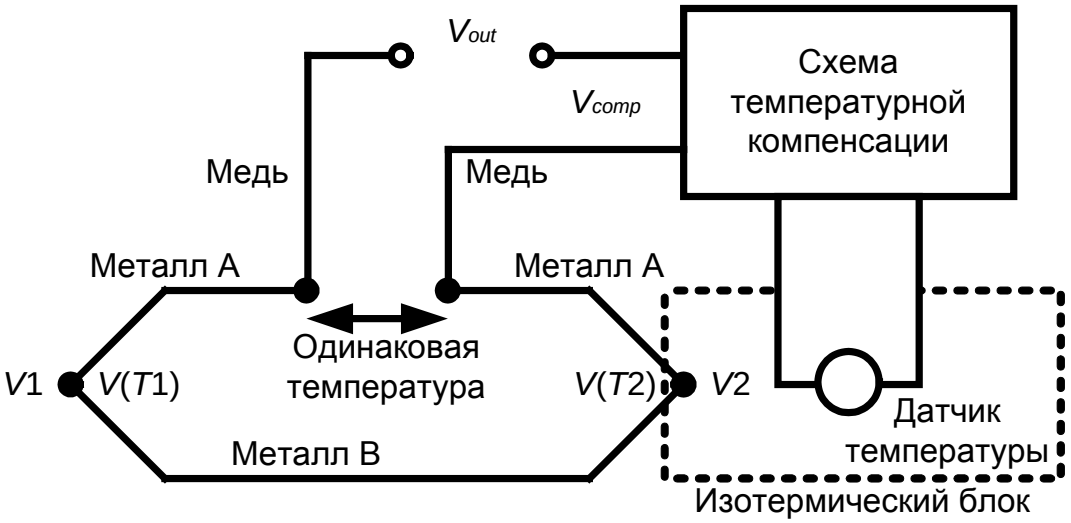


Рис. 2.18. Классическая компенсация температуры холодного спая при использовании опорного спая, находящегося при температуре таяния льда (0 °С)



$V_{comp} = f(T2)$
$V_{out} = V(T1) - V(T2) + V_{comp}$
если $V_{comp} = V(T2) - V(0°C)$, то $V_{out} = V(T1) -- V(0°C)$

Рис. 2.19. Использование датчика температуры
для компенсации холодного спая

Рассмотрим схему сопряжения микропроцессорной системы с термопарой типа К (рис. 2.21). Здесь обеспечивается компенсация холодного спая для температур от 0 °С до 250 °С [9]. Схема работает от одного источника питания от +3,3 В до +12 В и формирует передаточную характеристику выходного напряжения 10 мВ/°С. Термопара типа К имеет коэффициент Зеебека приблизительно 41 мкВ/°С (см. ниже), поэтому на холодном спае устанавливается датчик температуры с температурным коэффициентом 10 мВ/°С – микросхема TMP35. Он используется совместно с делителем *R*1 и *R*2 для того, чтобы ввести компенсирующий температурный коэффициент холодного спая противоположного знака величиной –41мкВ/°С.

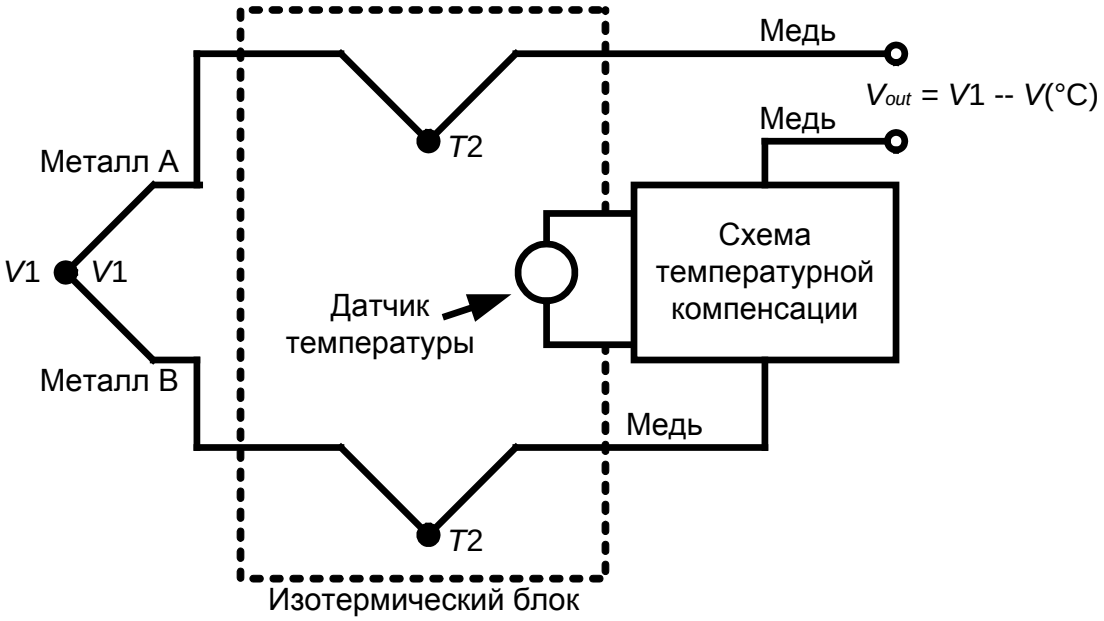


Рис. 2.20. Установка термопарных проводников
непосредственно в изотермическом блоке

Указанное включение препятствует появлению ошибки измерения температуры, обусловленной непосредственным соединением между проводниками термопары и трассами печатных проводников платы. Данная компенсация работает исключительно хорошо в диапазоне температур окружающей среды от 20 °С до 50 °С.

По диапазону измерения 250 °С термопара дает изменение выходного напряжения в 10.151 мВ. Поскольку требуемое изменение выходного сигнала по верхнему пределу составляет 2.5 В, усиление в цепи будет 246.3. Выбор *R*4 = 4.99 кОм даст величину К5 = 1.22 МОм. Поскольку ближайшая величина 1 % резистора *R*5 составляет 1.21 МОм, используется дополнительный потенциометр 50 кОм для точной настройки выходного напряжения по верхнему пределу. Интегральная схема OP193 является операционным усилителем с однополярным питанием, его выходной каскад не работает в режиме от питания до питания, и его выходной сигнал доходит только до потенциала +0.1 В относительно земли. По этой причине для смещения выходного напряжения приблизительно на 0.1 В устанавливается дополнительный резистор *R*3 на источник питания 5 В. Это напряжение смещения (10 °С) вычитается при расчетах результатов измерений. Резистор *R*3 также обеспечивает определение обрыва цепи термопары, устанавливая величину выходного напряжения больше 3 В, если термопара оборвана. Резистор *R*7 балансирует входной импеданс операционного усилителя OP193, а пленочный конденсатор 0.1 мкФ уменьшает величину шума на неинвертирующем входе.

Зарубежной промышленностью выпускаются интегральные схемы инструментальных усилителей с компенсацией холодного спая. Например, ИС AD594/AD595 от Analog Devices (рис. 2.22) [9]. Он включает в себя

компенсатор холодного спая на температуру таяния льда и калиброванный усилитель с непосредственным подключением к выходу термопары и выходным сигналом высокого уровня (10 мВ/°С). Переключение перемычек на выводах установки режима позволяет использовать ИС в качестве линейного усилителя-компенсатора или релейного регулятора температуры, использующего фиксированное значение или дистанционное управление точкой установки температуры. ИС можно использовать для прямого усиления напряжения компенсации, получая тем самым отдельный преобразователь температуры в градусах Цельсия с выходным сигналом 10 мВ/°С. Важно помнить, что ЧИП ИС был при той же самой температуре, что и холодный спай термопары, что обычно достигается установкой их обоих в непосредственной близости друг от друга и изолированием от источников тепла.

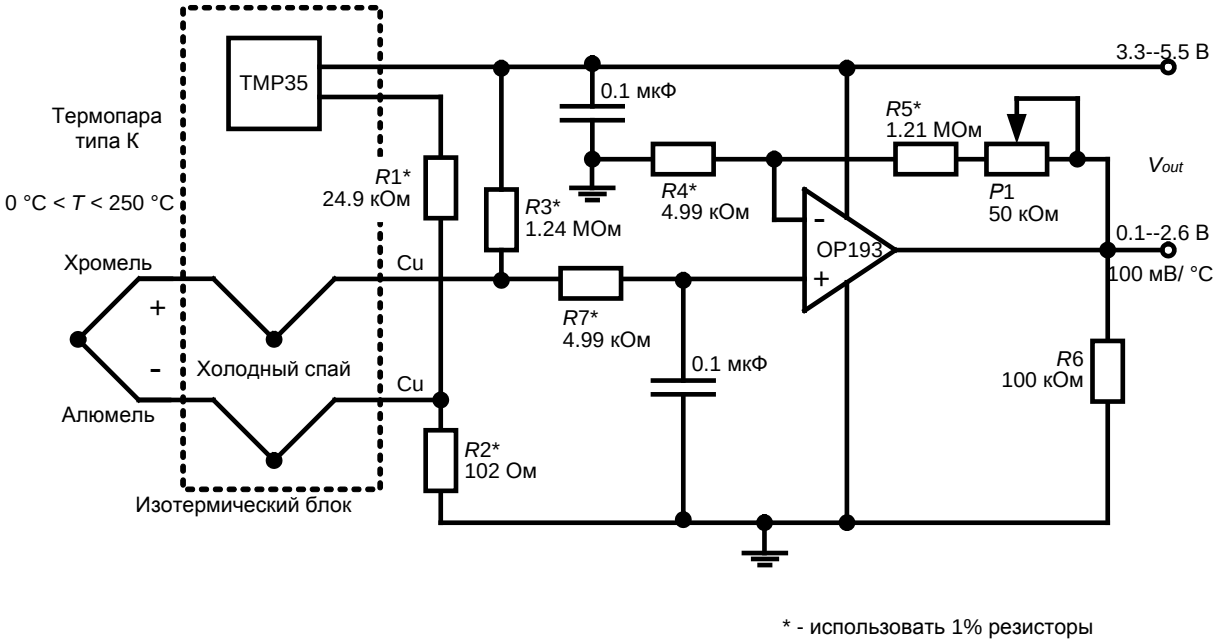


Рис. 2.21. Использование датчика температуры (TMP35)
для компенсации холодного спая термопары

Более совершенные ИС AD596/AD597 являются релейными регуляторами с установкой температуры, которые используются при высоких температурах, например в устройствах, связанных с управлением печами. Для получения внутреннего сигнала, пропорционального температуре, устройство выполняет компенсацию холодного спая и усиливает сигналы термопар типа J/K [10].

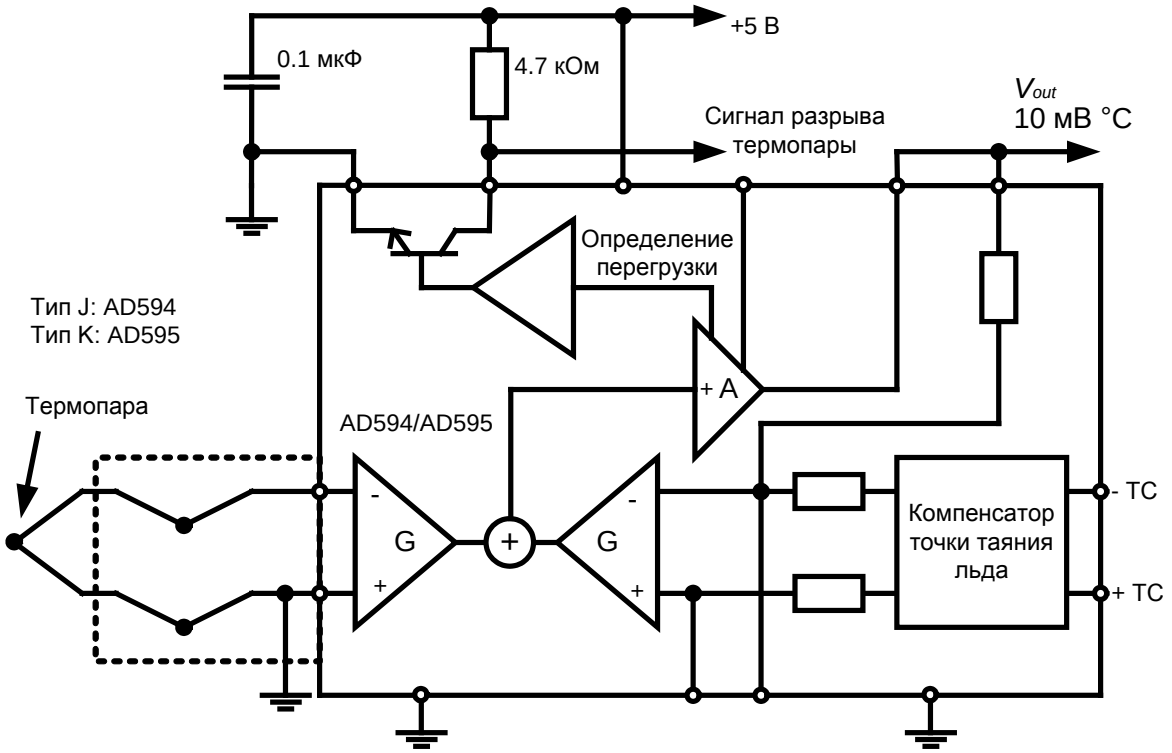


Рис. 2.22. Монолитные усилители термопар AD594/AD595
с компенсацией холодного спая

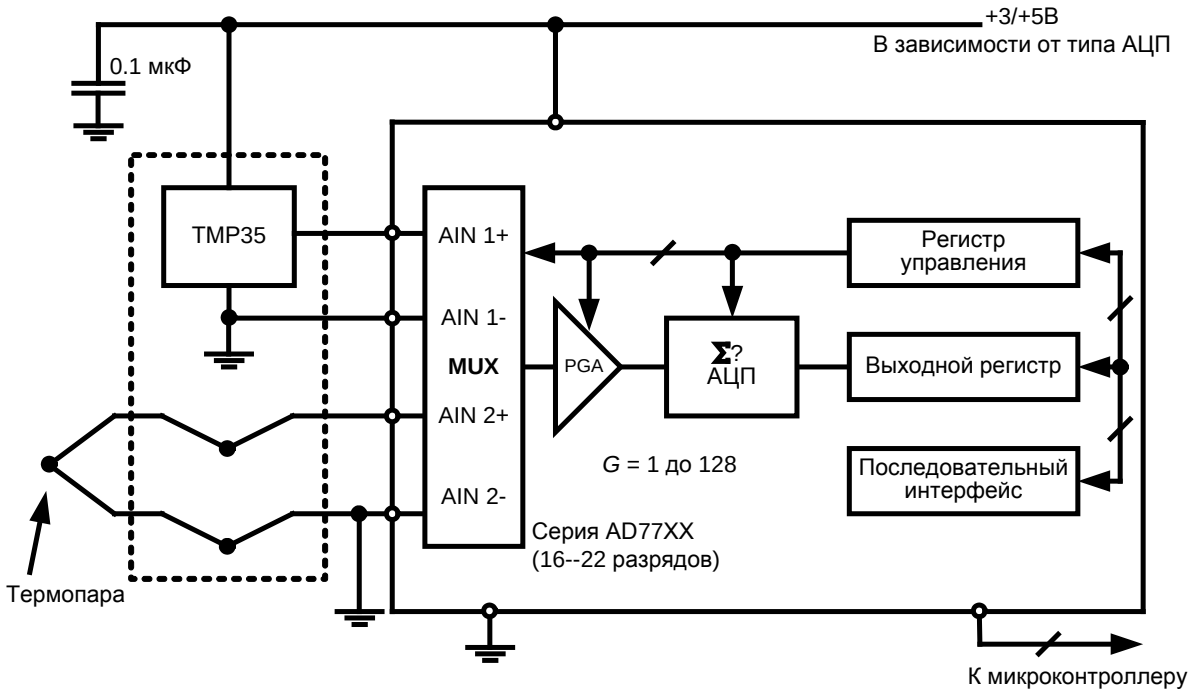


Рис. 2.23. АЦП семейства AD77XX,
используемый совместно с температурным датчиком TMP35
для компенсации температуры холодного спая

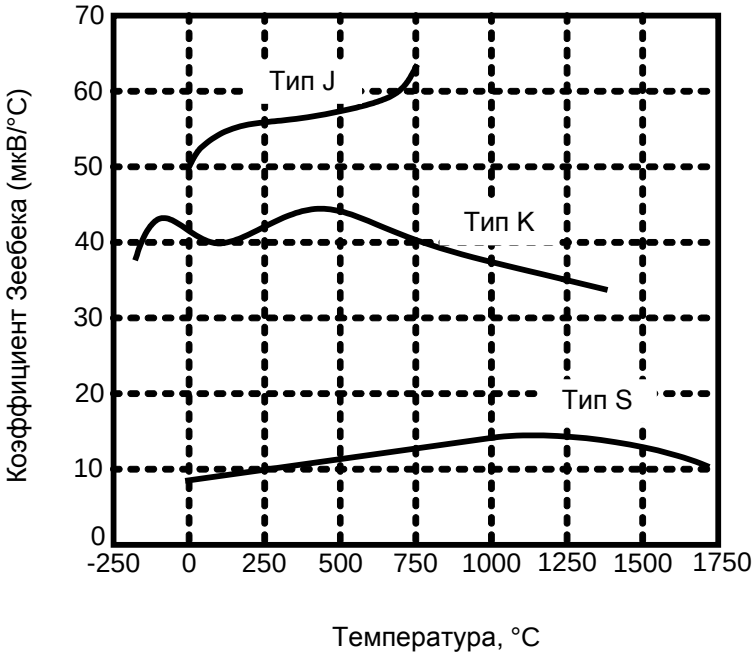


Рис. 2.24. Выходные напряжения для термопар типов J, K, S

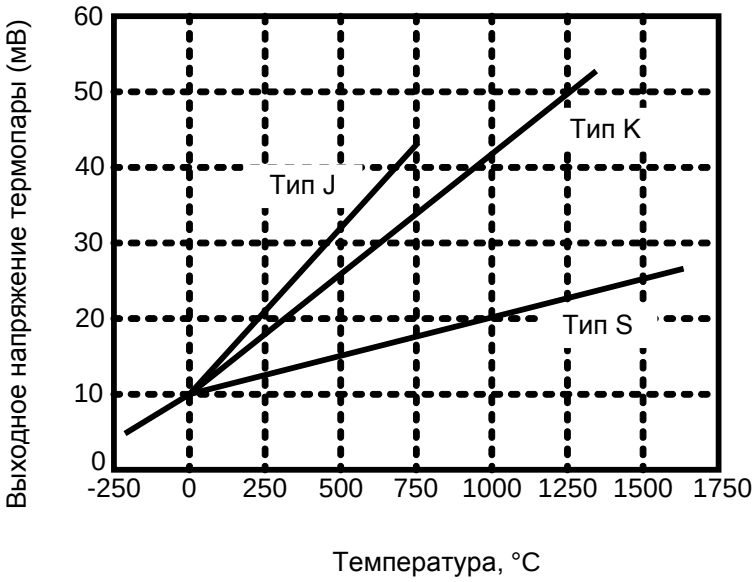


Рис. 2.25. Зависимость коэффициента Зеебека для термопары от температуры

Однако вышеперечисленные устройства не производят компенсацию нелинейности термопар (рис. 2.24). Для компенсации нелинейности термопары рекомендуется использовать высокоточный АЦП и последующую программную обработку сигнала. На рис. 2.23 представлена микропроцессорная система, предназначенная для квантования напряжения с термопары. Для анализа выхода температурного датчика холодного спая используются два мультиплексных входа АЦП. Входной усилитель программируется на усиление от 1 до 128, а разрешение АЦП составляет от 16 до 22 разрядов (в зависимости от выбранного конкретного АЦП). Микроконтроллер выполняет

арифметические действия по компенсации температуры холодного спая и линеаризации характеристики термопары [11].

Известно, что коэффициент Зеебека (изменение выходного напряжения при изменении температуры чувствительного спая) меняется с температурой измерительного спая, поэтому при выборе термопары для выполнения измерений в заданном диапазоне температур необходимо выбирать термопару, коэффициент Зеебека которой в меньшей степени меняется в заданном рабочем диапазоне (рис. 2.25) [8].

Например, для измерения температуры в диапазоне от 200 °С до 500 °С необходимо применять термопару J-типа, так как она имеет коэффициент Зеебека, меняющийся менее чем на 1 мкВ/°С в данном промежутке.

2.7.3. Резистивные датчики температуры

У резистивных датчиков температуры (РДТ) сопротивление меняется с изменением температуры. Не следует путать РДТ с терморезисторами, о которых речь пойдет ниже. Для изготовления РДТ применяется дорогостоящий платиновый провод, наматываемый на керамический каркас. Резистивные датчики температуры имеют величину сопротивления от 100 Ом до 1000 Ом. Типовой температурный коэффициент РТД составляет порядка 0.385 Ом/°С для 100 Ом платинового РДТ. РДТ более точны и линейны, чем термопары. На рис. 2.26 для сравнения показан температурный коэффициент 100 Ом РДТ и коэффициент Зеебека термопары типа S. Как видно из рисунка, по диапазону от 200 до 800 °С РДТ более линейны.

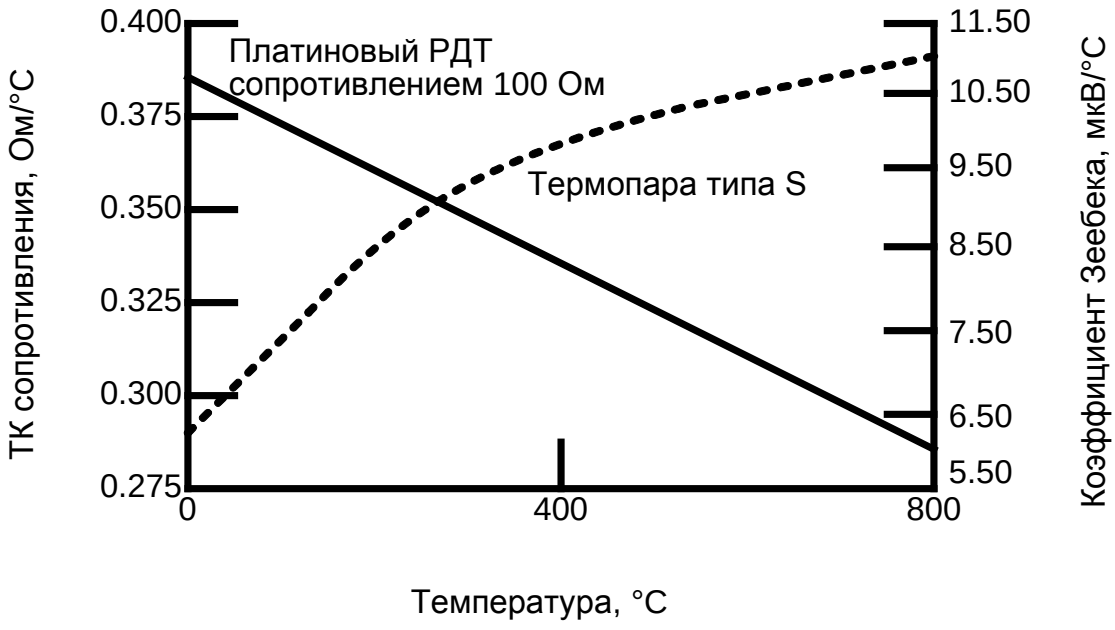


Рис. 2.26. Резистивные датчики температуры (РДТ)

Резистивные датчики температуры являются пассивными датчиками и требуют наличия тока возбуждения. Поскольку ток, текущий через РТД, нагревает его, саморазогрев изменяет температуру РДТ и проявляется ошибка измерения. Таким образом, при разработке схем сопряжения с РДТ следует учитывать величины саморазогрева. Она не должна превышать 0.5 °С.

Изготовители различают ошибки, связанные с саморазогревом для различных номиналов и размеров РДТ в воздушном потоке и без него. Для того чтобы уменьшить ошибки из-за саморазогрева, следует использовать минимально возможные токи возбуждения для достижения требуемого разрешения системы и выбирать РДТ с наибольшими номиналами, дающими, однако, приемлемый по величине временной отклик.

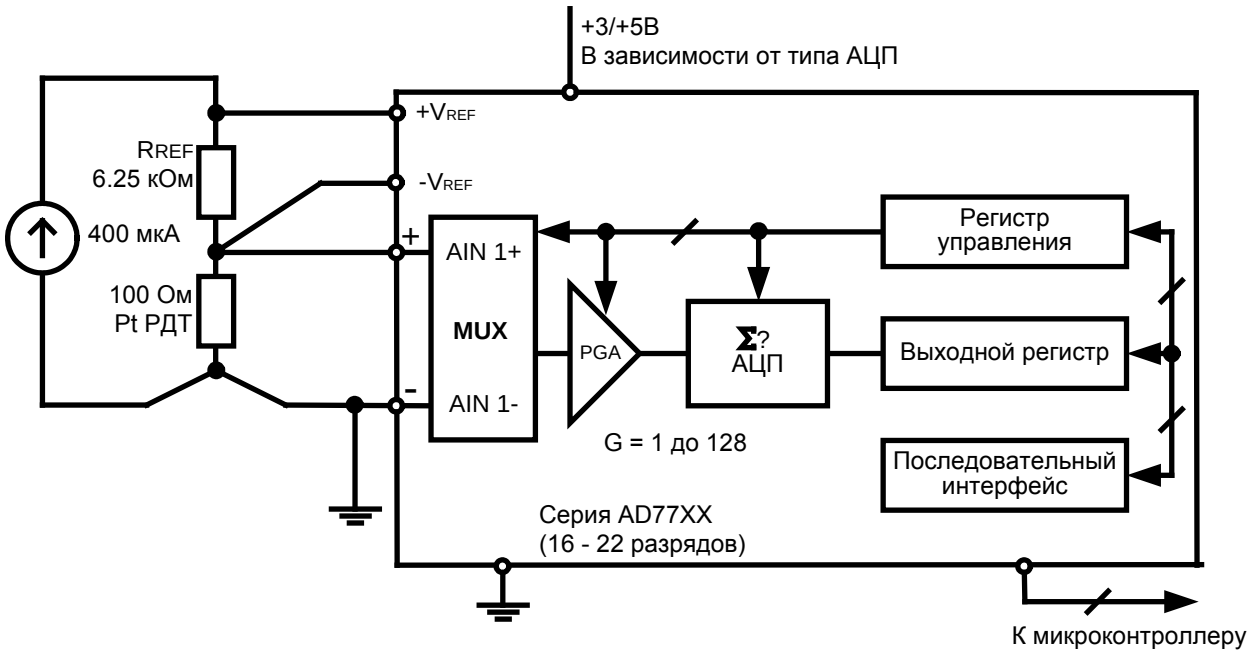


Рис. 2.27. Подключение ПТ РДТ к АЦП с высоким разрешением

Как правило, РДТ подключают по мостовой схеме. Выход моста усиливается. На рис. 2.27 показан ПТ РДТ 100 Ом, питаемый током от источника тока возбуждения 400 мкА. Выходной сигнал датчика квантуется АЦП семейства AD77XX. Отметим, что источник тока возбуждения РДТ также создает опорное напряжение 2.5 В для АЦП, используя резистор 6.25 кОм. Изменение тока возбуждения не влияет на точность схемы, поскольку как входное напряжение, так и опорное напряжение измеряются относительным образом. В то же время резистор 6.25 кОм должен обладать как можно меньшим температурным коэффициентом, с тем, чтобы избежать ошибок измерения. Применение АЦП с высоким разрешением, в составе которого имеется усилитель с программируемым усилением (усиление от 1 до 128), исключает необходимость использования дополнительной нормирующей цепи [10].

2.7.4. Термисторы

Термисторы – это недорогие температурно-чувствительные резисторы. Их изготавливают из полупроводниковых материалов, которые имеют как положительный, так и отрицательный температурный коэффициент. На рис. 2.28 показана зависимость сопротивления термистора с отрицательным температурным коэффициентом (ОТК) [Negative Temperature Coefficient – NTC] от температуры. Термистор является наиболее нелинейным устройством из рассмотренных ранее, но в то же время он наиболее чувствителен [11].

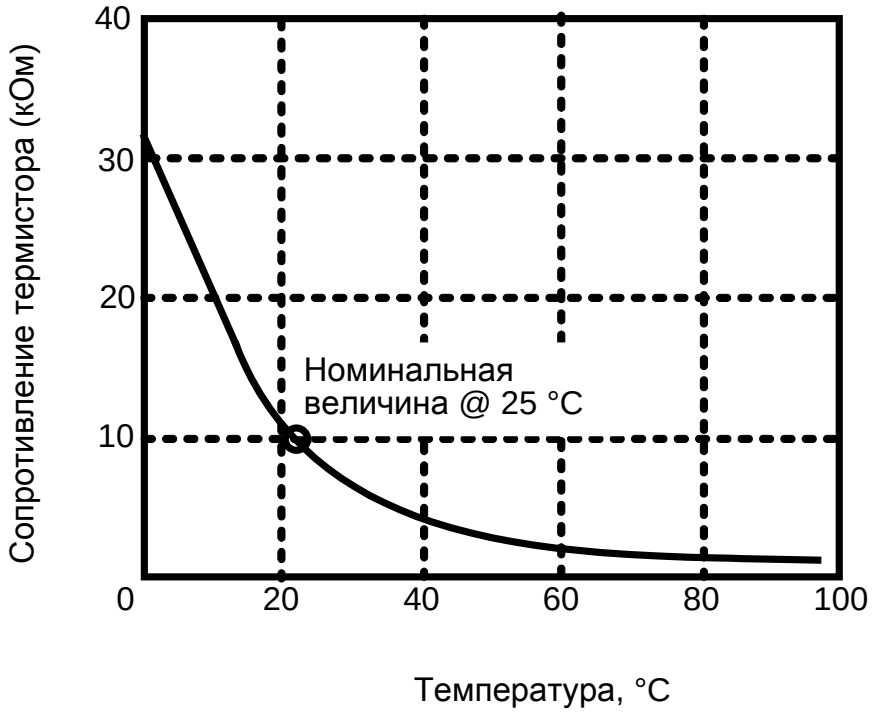


Рис. 2.28. Поведение сопротивления термистора 10 кОм с ОТК

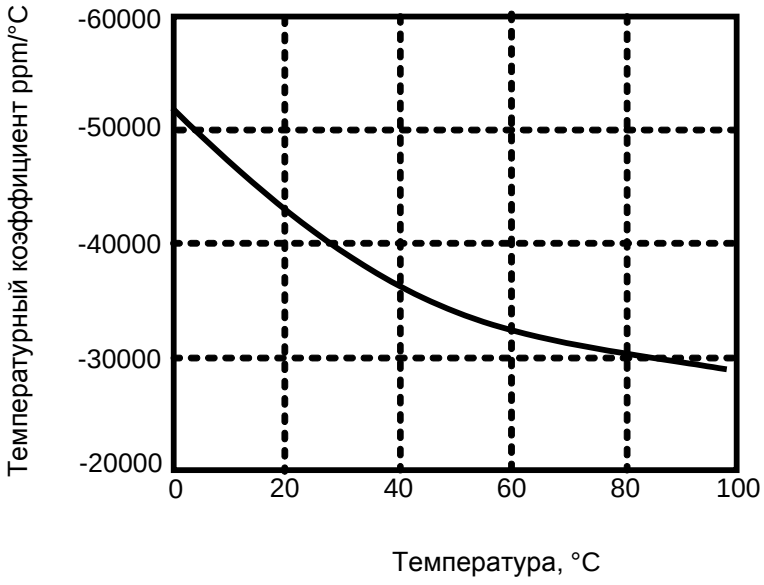


Рис. 2.29. Температурный коэффициент 10 кОм термистора с ОТК

Поскольку термисторы обладают высокой чувствительностью, они практически незаменимы для высокоскоростного определения температуры. Тем не менее следует учитывать, что термистор достаточно нелинеен (рис. 2.29), по этой причине требуется выполнение линеаризации для всех величин температуры, исключая только весьма узкий диапазон измерений. Как следствие, термисторы применяются в узком диапазоне измерений.

Самый простой метод линеаризации термисторов – установка параллельного шунтирующего резистора (рис. 2.30, рис. 2.31).

Величина этого дискретного резистора рассчитывается из равенства

$$R = \frac{RT2 \cdot (RT1 + RT3) - 2RT1 \cdot RT3}{RT1 + RT3 - 2RT2},$$

где $RT1$ – сопротивление термистора при температуре $T1$, нижний предел температурного диапазона;

$RT3$ – сопротивление термистора при температуре $T3$, верхний предел температурного диапазона;

$RT2$ – сопротивление термистора при температуре $T2$, средняя точка температурного диапазона, $T2 = (T1 + T3)/2$.

Например, для термистора 10 кОм с ОТК $RT1$ = 32650 Ом при 0 °С; $RT2$ = 6532 Ом при 35 °С и $RT3$ = 1752 Ом при 70 °С. Это приводит к величине R = 5.17 кОм.

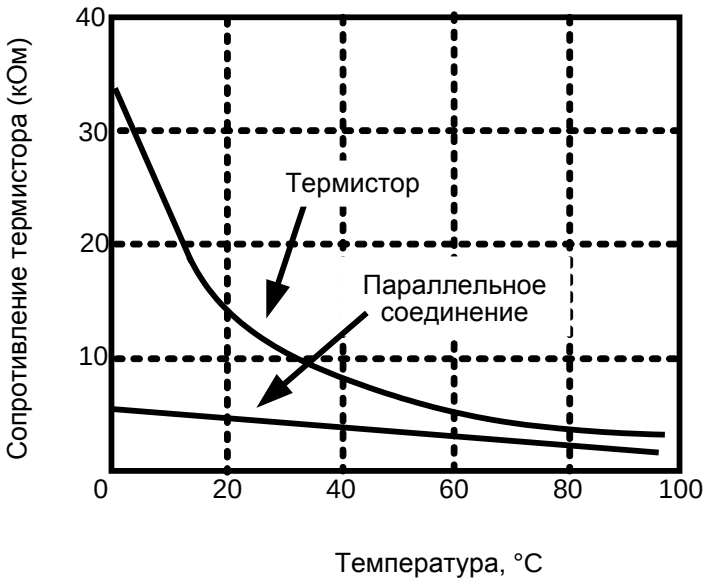


Рис. 2.30. Линеаризация термистора с ОТК
путем подключения параллельного резистора 5.17 кОм

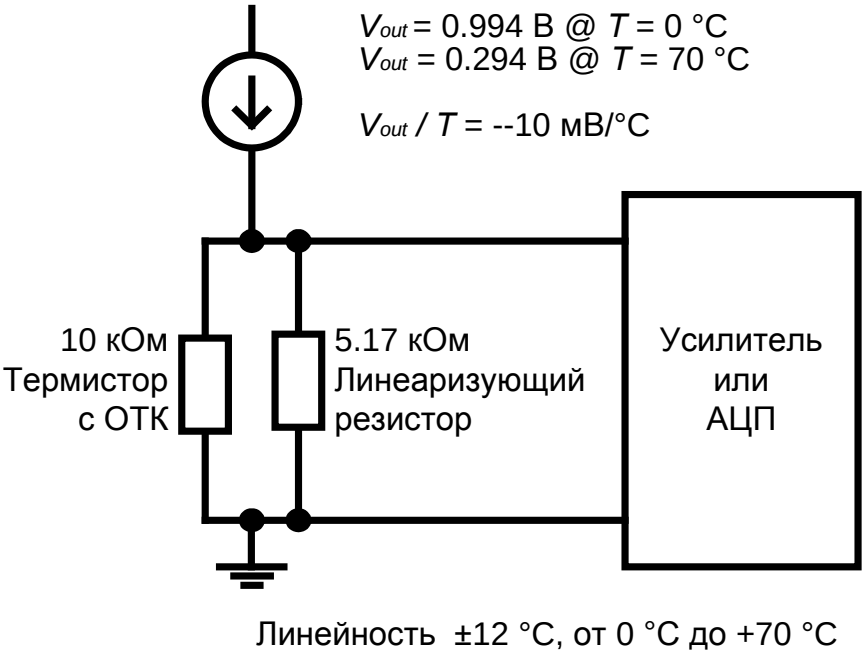


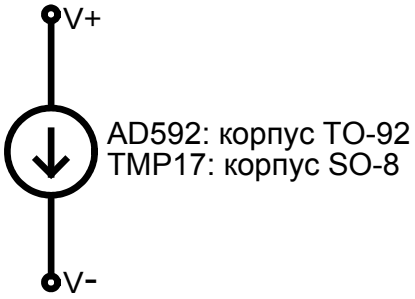
Рис. 2.31. Усилитель с линеаризованным термистором

Точность, необходимая для нормирующей схемы, зависит от линейности цепи. Для приведенного выше примера цепь дает нелинейность от –2.3 °C до +2.0 °C. Для дальнейшей линеаризации сигнал с выхода подается на АЦП (рис. 2.31). Отметим, что выходной сигнал цепи с термистором имеет величину около –10 мВ/°C, при этом разрешения 12-разрядного АЦП более чем достаточно.

2.7.5. Полупроводниковые датчики температуры

Интегральные полупроводниковые датчики температуры имеют самую высокую точность и линейность, однако ограничены диапазоном рабочих температур (от –55 °C до +150 °C). Эти датчики имеют встроенные усилители и могут масштабировать выходные сигналы, приводя их к удобным величинам (например, 10 мВ/°C). Примером таких датчиков может служить ИС AD592 от Analog Devices или ИС TMP17 от Texas Instruments (рис. 2.32). Это датчики с токовым выходом, которые имеют коэффициент преобразования 1 мкА/К, они не требуют внешней калибровки и имеют несколько градаций по точности [13].

На рис. 2.33 представлена микропроцессорная система, использующая в качестве датчика температуры ИС ADT45 (или ADT50), имеющего выход в виде относительного напряжения.



Коэффициент преобразования	1 мкА/К
Номинальный выходной ток @ 25 °C	298.2 мкА
Диапазон рабочего напряжения	4–30 В
Максимальная ошибка @ 25°C	±0.5 °C, ±1 °C во всем диапазоне
Типовая нелинейность (AD592CN)	±0.1 °C
Максимальная ошибка @ 25°C	±2.5 °C, ±3.5 °C во всем диапазоне
Типовая нелинейность (TMП17P):	±0.5 °C
AD592 специфицируется	от –25 °C до +105 °C
TMP17 специфицируется	от –40 °C до +105 °C

Рис. 2.32. Датчики с токовым выходом: AD592, TMP17

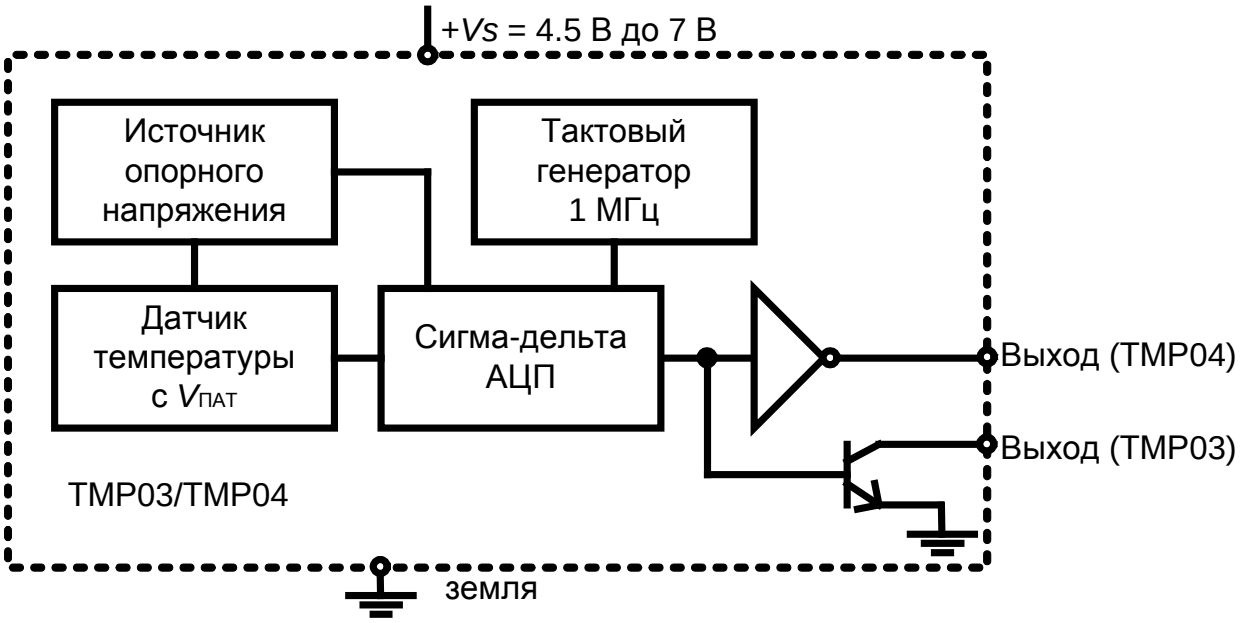
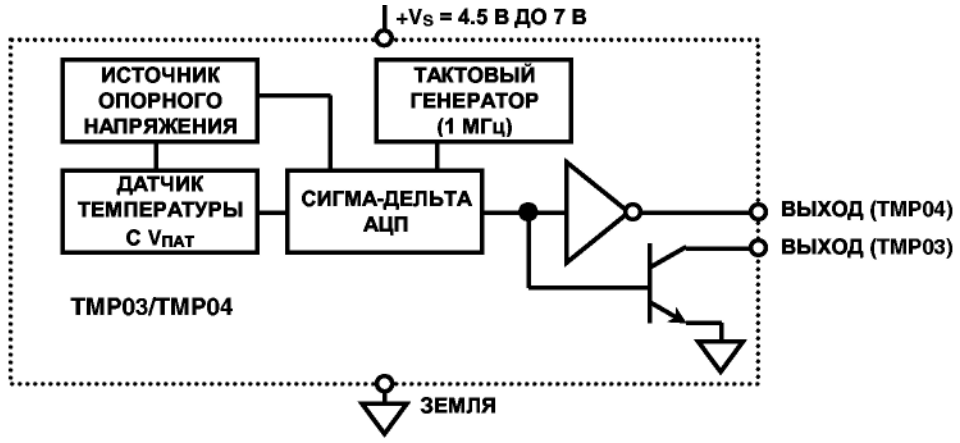


Рис. 2.33. Датчики с выходом в виде относительного напряжения

Для снижения уровня высокочастотных помех (поскольку датчики температуры работают с очень малым током потребления) выводы подключения питания данного датчика заблокированы керамическим конденсатором 0.1 мкФ, имеющим весьма короткие выводы (предпочтительно элемент поверхностного монтажа), он должен быть расположен настолько близко к выводам питания, насколько это возможно [14].

2.7.6. Датчики температуры с цифровым выходом

Датчики температуры с цифровым выходом имеют встроенный на кристалл АЦП. Примером такого датчика может служить датчик TMP03/TMP04, который содержит опорный источник напряжения, сигма-дельта АЦП и тактовый генератор (рис. 2.34). Встроенный АЦП обеспечивает 12-разрядную точность при весьма малых размерах схемы. Выходной сигнал сигма-дельта модулятора кодируется, используя соответствующую схему, которая дает на выходе последовательный цифровой код в виде частотно-модулированного сигнала (рис. 2.35). Данный сигнал весьма просто декодируется с помощью любого микропроцессора, в значениях температуры в градусах Цельсия или Фаренгейта, и всегда передается по одному проводу. Номинальная выходная частота составляет 35 Гц при +25 °С. Устройство работает с фиксированной длительностью импульса *T*1, составляющей 10 мс [14].



Основные характеристики TMPO3/TMP04:

Номинальная длительность импульса *T*1	10 мс
Ошибка в диапазоне температур	±1.5 °С
Типовая нелинейность	±0.5 °С
Рабочий диапазон	от –40 °С до +100 °С
Номинал отношения *T*1/*T*2 при 0 °С	60 %
Номинальная частота при +25 °С	35 Гц
Потребляемая мощность по 5 В	6.5 МВт
Тип корпуса: TO-92 SO-8 или TSSOP	

Рис. 2.34. Датчики с цифровым выходом TMPO3/04

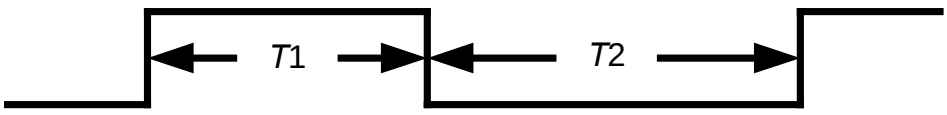


Рис. 2.35. Формы выходного сигнала для TMPO3/04

Выходной сигнал TMP03/TMP04 представляет собой поток импульсов, температура определяется выражениями:

$$\text{Температура } (°\text{C}) = 235 - \left(\frac{400 \times T1}{T2}\right);$$

$$\text{Температура } (°\text{C}) = 455 - \left(\frac{720 \times T1}{T2}\right).$$

Рассмотрим подключение вышеописанного датчика к микропроцессорной системе управления. В качестве такой системы может использоваться любой микроконтроллер, имеющий в своем составе таймеры, с помощью которых можно очень просто декодировать частотно-модулированный сигнал с TMP03/TMP04. Типовой интерфейс к микроконтроллеру Intel 80C51 показан на рис. 2.36.

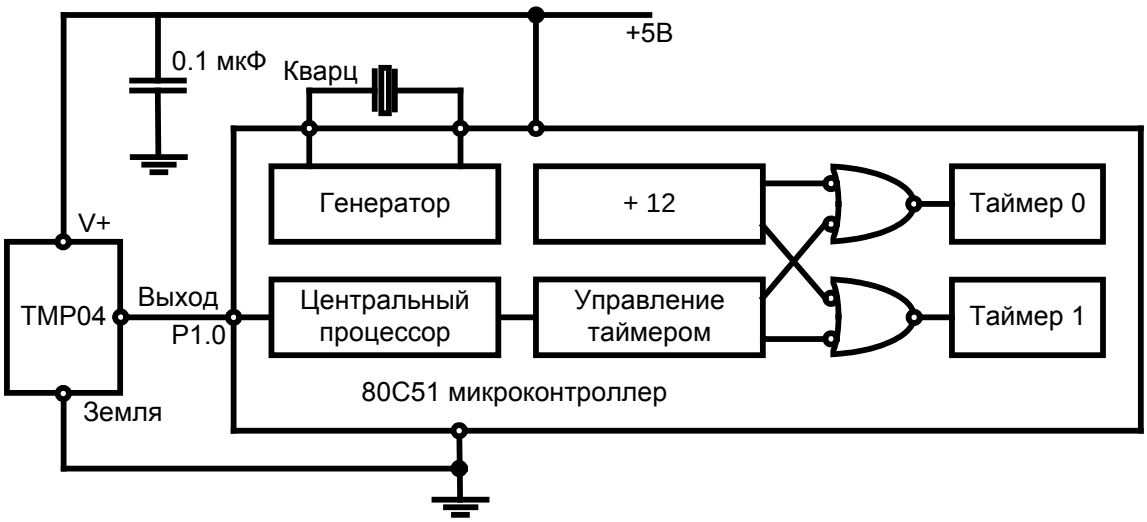


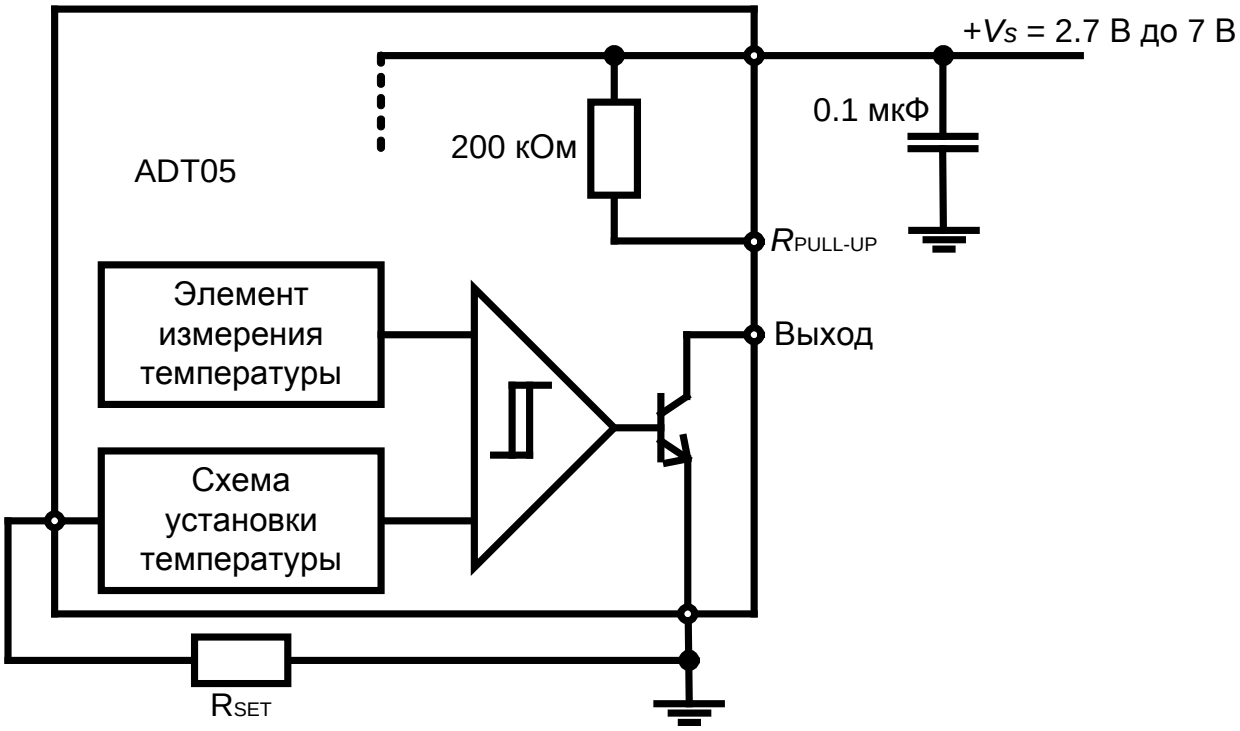
Рис. 2.36. Интерфейс TMP04 к микроконтроллеру

Два таймера, обозначенные как таймер 0 и таймер 1, имеют 16 разрядов. Системная частота микроконтроллера, деленная на 12, является входом для таймеров. Микроконтроллер настраивает порт P1.0 и запускает таймер 0 по положительному перепаду выходного сигнала датчика. Микроконтроллер останавливает таймер 0 и запускает таймер 1 по отрицательному перепаду выходного сигнала датчика.

Когда выходной сигнал снова придет в высокое состояние, содержимое таймеров $T1$ и $T2$ переписывается в регистры таймер 0 и таймер 1 соответственно. Далее для расчета температуры подпрограммы используют равенства, приведенные выше [15].

2.7.7. Термореле и регуляторы с установкой температуры

Если датчик температуры подключить к компаратору, то мы получим термореле. Термореле срабатывает при достижении температурного порога. Примером термореле могут служить недорогая ИС ADT05 или ADT22/23, ADT14 от Analog Devices или TMP01 от Texas Instruments. ADT05 с помощью единственного внешнего резистора позволяет устанавливать температуру переключения с точностью до 2 °C в диапазоне от –40 °C до +150 °C (рис. 2.37). ADT05 предназначена для работы с однополярным питанием в диапазоне от +2.7 В до +7 В, что позволяет использовать ее в приложениях с батарейным питанием, а также в индустриальных системах управления. Вследствие низкой рассеиваемой мощности (200 мкВт при 3.3 В) ошибки из-за саморазогрева минимальны, а время работы даже от аккумуляторной батареи максимально. В ИС включен резистор подключения выхода к питанию для управления такими нагрузками, как входы КМОП.



Точность установки	±2 °C
Внутренний гистерезис	4 °C
Специфицированный рабочий диапазон	от –40 °C до +150 °C
Рассеиваемая мощность	200 мкВт при 3.3 В

Рис. 2.37. Регулятор термостата

Величина резистора установки рабочей температуры определяется равенством [16, 17].

$$R_{SET} = \frac{39\text{МОм}\ ^\circ\text{C}}{T_{SET}\,(^\circ\text{C}) + 281.6\,^\circ\text{C}} - 90.3\text{кОм}.$$

2.7.8. Аналого-цифровые преобразователи с датчиком температуры на одном кристалле

Цифровые датчики температуры помимо встроенного температурного датчика и АЦП имеют, как правило, контроллер последовательного обмена данными (SPI™ и QSPI™ и MICROWIRE™ фирмы National Semiconductor), например, датчики серии AD7816/7817/7818. Функциональные блок-схемы AD7816, AD7817, AD7818 показаны на рис. 2.38, рис. 2.39, рис. 2.40.

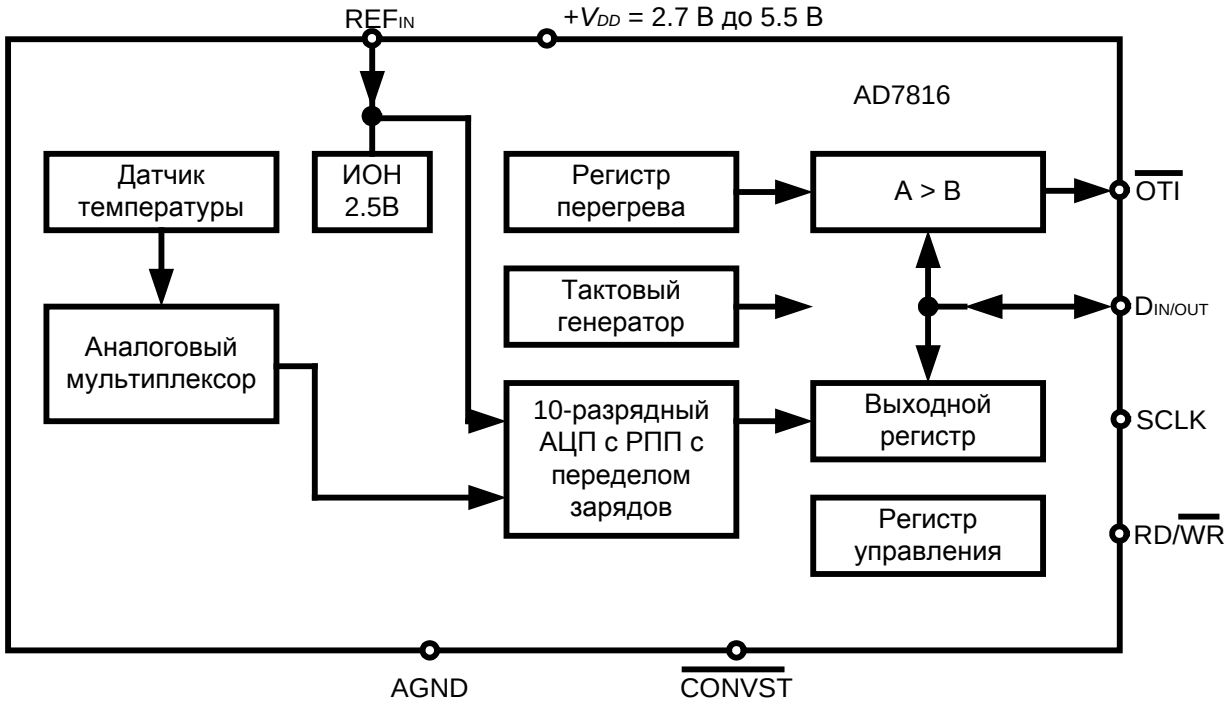


Рис. 2.38. 10-разрядный цифровой датчик температуры с последовательным интерфейсом (AD7816)

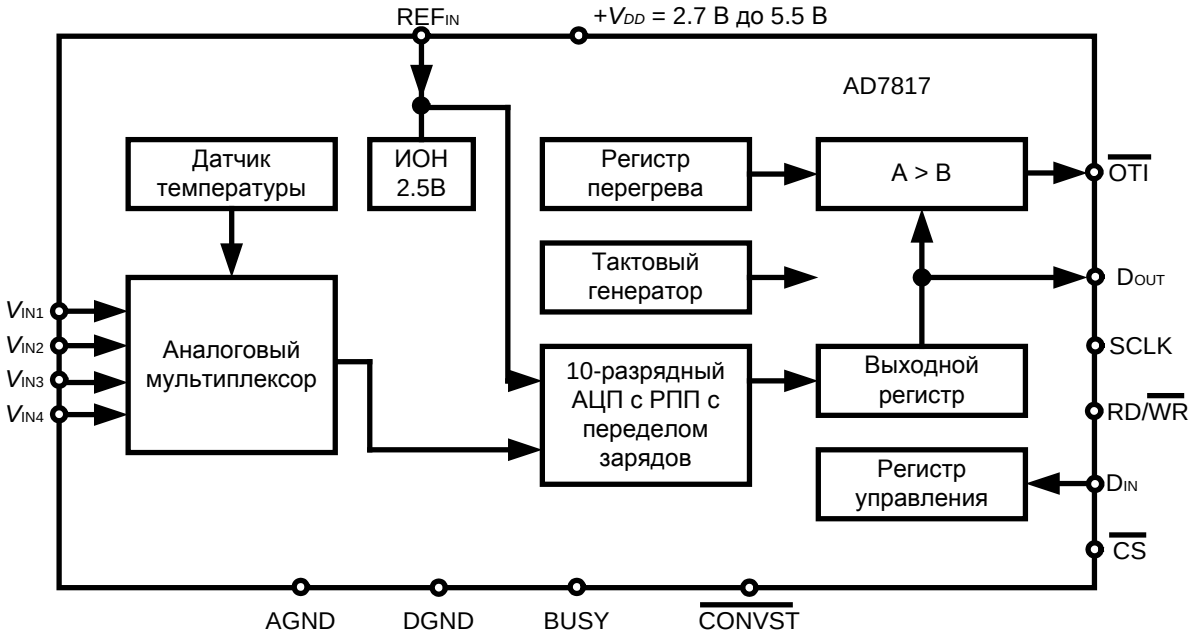


Рис. 2.39. 10-разрядный АЦП
с мультиплексированными входами и датчиком температуры (AD7817)

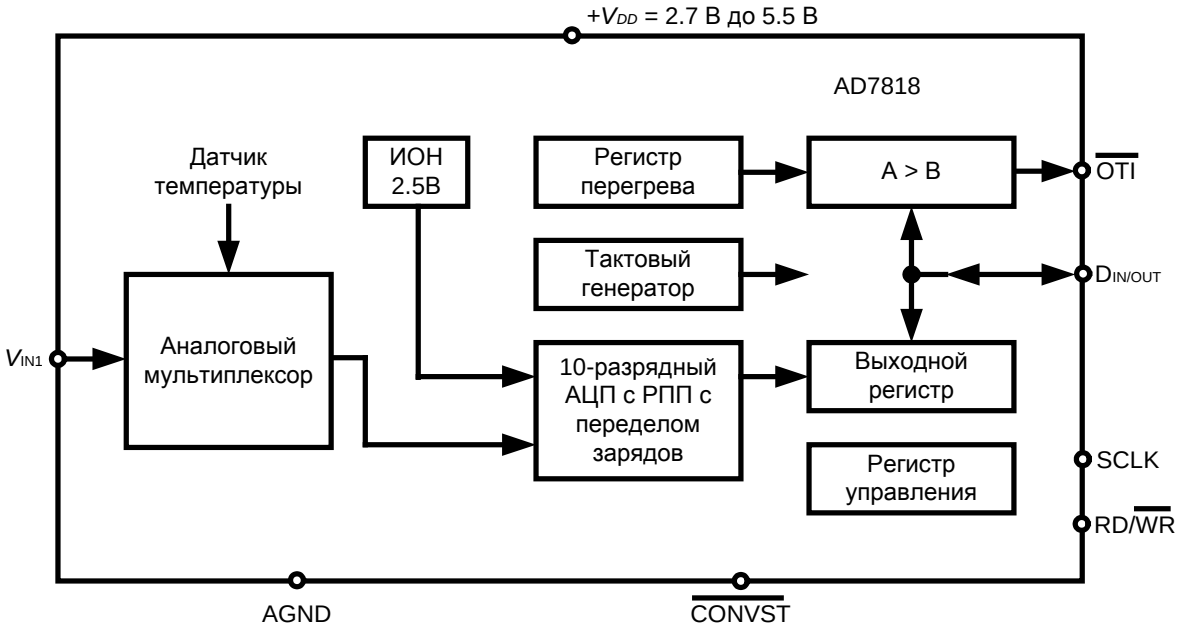


Рис. 2.40. 10-разрядный АЦП
с одним входом и датчиком температуры (AD7818)

10-разрядный АЦП с временем преобразования 10 мкс;
гибкий последовательный интерфейс (Intel 8051, SPI™, QSPI™, MICROWIRE™);
наличие на кристалле датчика температуры от –55 °С до +125 °С;
точность измерения температуры +2 °С (от –40 °С до +85 °С);

наличие встроенного опорного источника 2.5 В ± 1 %;
диапазон напряжения питания от (+2.7 В до +5.5 В);
рассеиваемая мощность 4 мВт на частоте выборок 10 Гц;
режим автопонижения питания после завершения преобразования;
выход «прерывания» по перегреву;
аналоговые входы, четыре для AD7817, один для AD7818;
AD7416/AD7417/AD7418 подобны перечисленным, но имеют 1^2C интерфейс [16].

2.8. Сети датчиков, интеллектуальные датчики

Технологии объединения микропроцессорных систем с датчиками в вычислительную сеть открывают более широкие возможности для систем сбора и анализа данных. Такие системы управления используют в качестве линий передачи данных различные промышленные стандарты: токовую петлю 4–20 мА, интерфейсы и протоколы 1-Ware, CAN, I2C, Ehternet, Lonwork и др.

Различные датчики и исполнительные устройства имеют встроенные на кристалл или конструктив контроллеры этих интерфейсов. Их создают специально под данный режим управления.

2.8.1. Токовая петля

На рис. 2.41 показано, как дистанционный исполнительный механизм управляется с помощью токовой петли со стороны помещения центрального пульта управления. Отметим, что выход передатчика на исполнительный механизм управляется ЦАП, в данном случае AD420. Весь процесс находится под управлением центрального компьютера, который подключается к микроконтроллеру и AD420. На этой схеме показан только один исполнительный механизм, однако реальная система индустриального управления содержит обычно значительное число исполнительных механизмов и датчиков. Отметим, что выход нуля шкалы ЦАП составляет 4 мА (а не нуль), а его верхний предел (полная шкала) – 20 мА. Выбор ненулевого выходного тока для нулевой точки позволяет передатчику определять факт разрыва цепи и одновременно позволяет питать дистанционный преобразователь через ту же самую петлю, если ток последнего менее 4 мА [15].

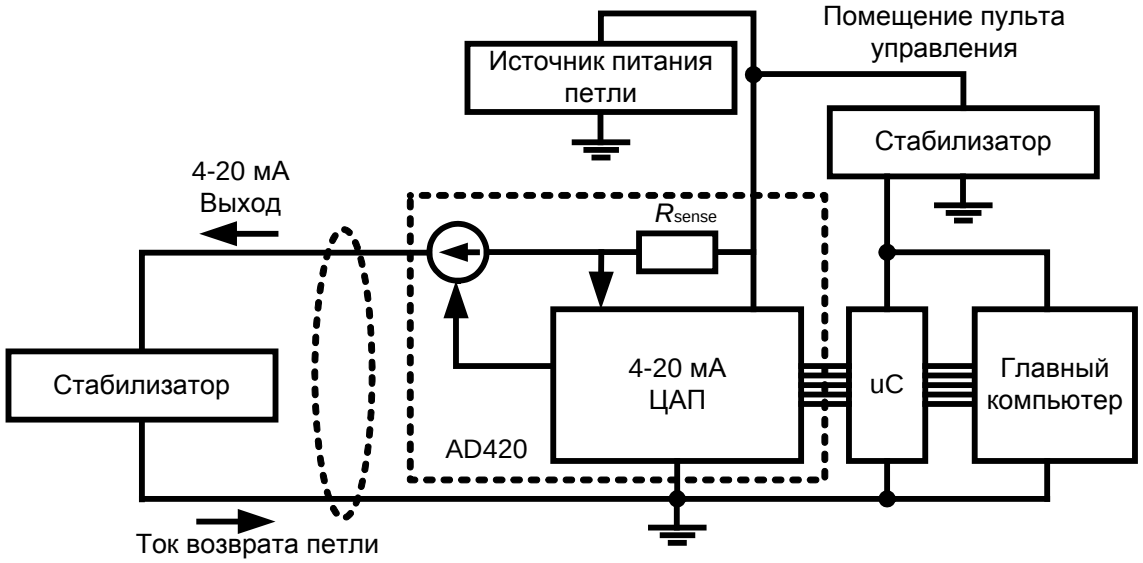


Рис. 2.41. Использование токовой петли 4–20 мА
для управления дистанционным исполнительным механизмом

Многие из цепей в помещении пульта управления питаются непосредственно от источника питания петли, напряжение которого лежит в пределах от 12 до 36 В. Однако часто это напряжение необходимо стабилизировать для питания таких устройств, как усилители, АЦП и микроконтроллеры. Ток петли измеряется с помощью резистора RSENSE, который фактически входит в состав ИС AD420. Внутренний ЦАП AD420 представляет собой 16-разрядный сигма-дельта ЦАП. Наличие последовательного цифрового интерфейса позволяет легко сопрягать его с микроконтроллером.

На рис. 2.42 показан интеллектуальный датчик с выходом 4–20мА с питанием от петли. Для того чтобы данная схема работала, полный суммарный ток всех элементов ее схемы должен быть не более 4 мА. Ядром этой схемы является ИС AD421, 16-разрядный ЦАП, питающийся от токовой петли. Ток внутреннего ЦАП 4–20мА, а также оставшаяся часть тока возврата, требующаяся для питания AD421 и других элементов схемы, протекает через измерительный резистор RSENSE. Измерительная цепь компенсирует эту оставшуюся часть тока возврата и гарантирует, что полный ток возврата будет равен току ЦАП, который соответствует коду, установленному на нем микроконтроллером. Выход датчика квантуется сигма-дельта АЦП AD7714/AD7715. Отметим, что полный ток, потребляемый цепью, менее требуемого максимума 4 мА. Устройство AD421 содержит цепь стабилизатора питания, который управляет затвором внешнего ДМОП – полевого транзистора и устанавливает напряжение питания из ряда 3, 3.3 или 5 В. Таким образом, максимальное напряжение в петле ограничивается только напряжением пробоя ДМОП транзистора [15].

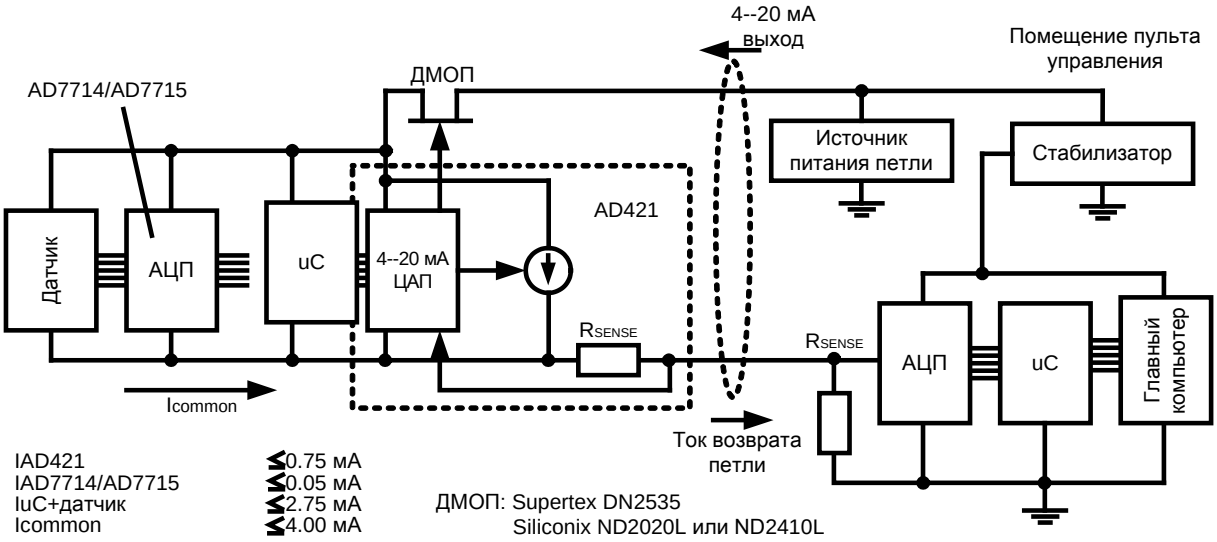


Рис. 2.42. Интеллектуальный датчик, питаемый от токовой петли 4–20 мА

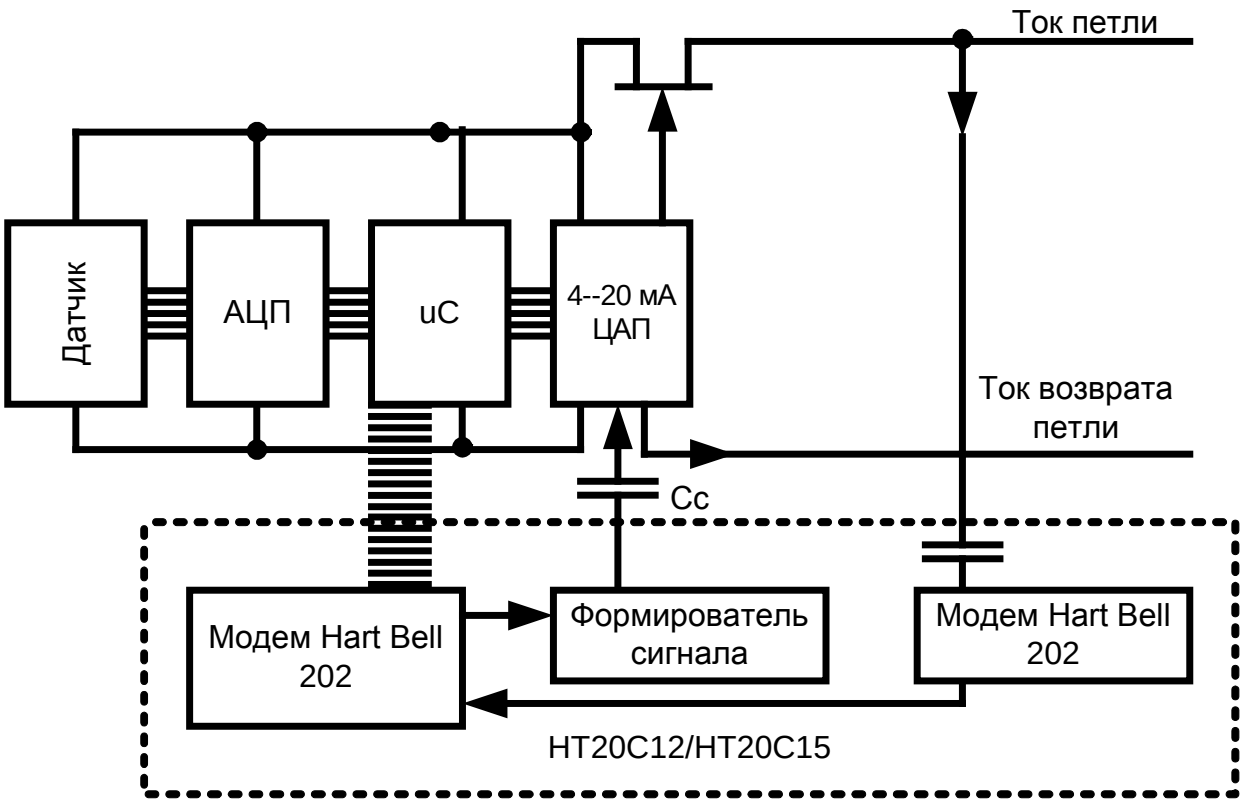


Рис. 2.43. Дистанционный интеллектуальный передатчик
с протоколом HART, использующий AD421 ЦАП с токовой петлей 4–20 мА

Протокол HART использует метод частотной модуляции в соответствии с коммуникационным стандартом (Bell202), который является одним из нескольких стандартов, используемых при создании систем передачи цифровых сигналов по телефонным линиям. Этот метод используется для наложения сигналов цифровой связи на токовую петлю 4–20 мА, соединяющую по-

мещение пульта управления с дистанционным передатчиком. Для представления двоичной 1 и 0 в протоколе используются две различные частоты 1200 и 2200 Гц соответственно. Эти гармонические сигналы низкого уровня со средней величиной, равной нулю, накладываются на сигнал постоянного тока. Данная схема позволяет одновременно использовать аналоговую и цифровую подсистемы связи.

При этом никаких компонент постоянного тока не добавляется к существующему току петли 4–20 мА, не считая цифровых данных, которые передаются по данной линии. Фаза сигнала частотной модуляции непрерывна. Таким образом, в петле 4–20 мА не будет наведенных высокочастотных компонент (обязанных процессу модуляции). Следовательно, имеющиеся аналоговые схемы будут продолжать нормально работать в системе, которая использует протокол HART, поскольку низкочастотная фильтрация (и без того обычно существующая) эффективно режектирует (исключает) цифровой сигнал. Низкочастотный однополюсный фильтр с частотой среза 10 Гц уменьшает величину наводок от связного сигнала до ±0.01 % от верхнего предела шкалы. Протокол HART предписывает, чтобы ведущие устройства (главная система управления) передавали в линию сигнал напряжения в то время, как ведомое (или локальное, периферийное, цеховое) устройство должно возвращать токовый сигнал. Токовый сигнал преобразуется в соответствующее напряжение резистором нагрузки петли в помещении пульта управления. На рис. 2.43 показана блок-схема интеллектуального информационно-измерительного передатчика. Информационно-измерительный передатчик – это такой передатчик, в котором функции его микропроцессора делятся между выполнением первичных измерений с генерацией измерительного сигнала и управлением подсистемой связи, которая позволяет устанавливать двустороннюю связь по тем же самым линиям, по которым передается измерительная информация. Интеллектуальный передатчик, содержащий протокол HART, является примером такого интеллектуального информационно-измерительного передатчика [16].

Данные, передаваемые в соответствии с HART-протоколом в токовую петлю, показанную на рис. 2.43, принимаются передатчиком с использованием полосового фильтра и модема и далее поступают в асинхронный последовательный порт микроконтроллера или в порт модема. В обратном направлении тоновые сигналы с HART-модема формируются и через разделительный конденсатор C_c подаются на выход AD421. Блок, содержащий модем BELL202, формирователь сигнала и полосовой фильтр, выпускается в виде законченной конструкции фирмой Symbios Logic, фирмой Inc – модель 20C15 и фирмой SMAR Research Corporation – модель HT2012.

2.8.2. Объединение датчиков в сеть

Рассмотрим сетевое объединение датчиков. Разумеется, что в данном случае следует рассматривать не дискретные датчики, а системы на кристалле (SOC), т. е. интеллектуальные датчики (рис. 2.44).

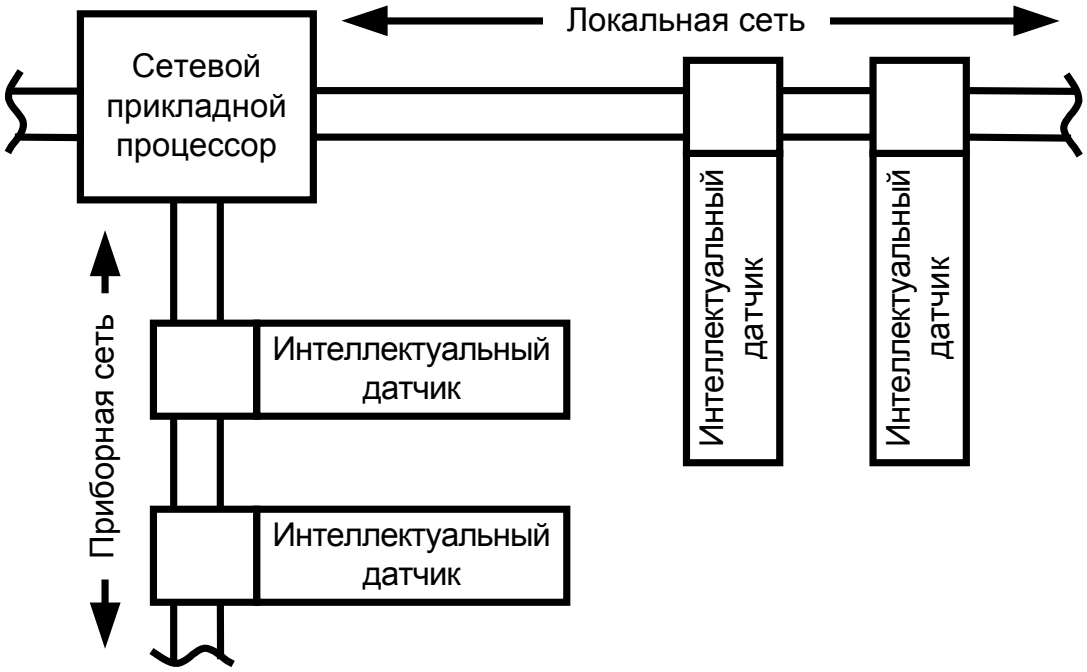


Рис. 2.44. Индустриальная цепь

Такие индустриальные сети могут принимать различные конфигурации. Цеховая сеть на рис. 2.44 представляет собой широкополосную распределенную сеть, например Ethernet или Lonwork. Эта цеховая сеть в обычном виде не предназначена для прямого подключения интеллектуальных датчиков. Большинство приборных сетей (таких как ASI-bus, CAN-bus и HART), кроме того, подают питание на интеллектуальные датчики по той же самой линии, по которой передаются последовательные данные.

Некоторые из стандартов индустриальных сетей, наиболее популярных в настоящее время, перечислены ниже. Каждый из них обладает собственными преимуществами и недостатками и каждый имеет свою собственную аппаратуру и последовательный протокол обмена. Это означает, что интеллектуальный датчик, предназначенный для работы в одной индустриальной сети, не обязательно будет работать в другой.

- Ethernet
- Foundation Fieldbus
- Lonwork
- Profibus
- Interbus-S
- CAN-Bus
- Device-Net
- World FIP
- P-NET
- HART

Universal Serial Bus (USB) ASI

Так как предприятия и многие другие объекты с сетями часто имеют набор разных сетей и подсетей, для них наиболее правильным (гибким) решением будет использование датчиков в режиме автоконфигурации («установи и работай»), совместимых с различными цеховыми и приборными сетями. Заслуга интерфейсного стандарта IEEE 1451.2 состоит в том, что он сделал реальностью существование датчиков, независимых от сети.

На рис. 2.45 показаны основные компоненты системы, совместимой с IEEE 1451.2. Интеллектуальный датчик (или интеллектуальное исполнительное устройство) здесь называется как STIM (Smart Transducer Interface Module) (интерфейсный модуль интеллектуального преобразователя – ИМИП). Он содержит один или более датчиков и/или исполнительных устройств с устройствами нормирования сигналов, АЦП или ЦАП для согласования датчиков/исполнительных устройств с резидентным микроконтроллером. Микроконтроллер имеет доступ к неразрушаемой памяти, которая содержит в себе поле TEDS (Transducer Electronic Data Sheet) (электронное описание преобразователя – ЭОП), которое содержит описания датчика/исполнительного устройства и которое можно прочитать через сеть. NCAP (Network Capable Application Processor) (сетевой прикладной процессор – СПП) представляет собой узел сети, в который будет подключаться STIM.

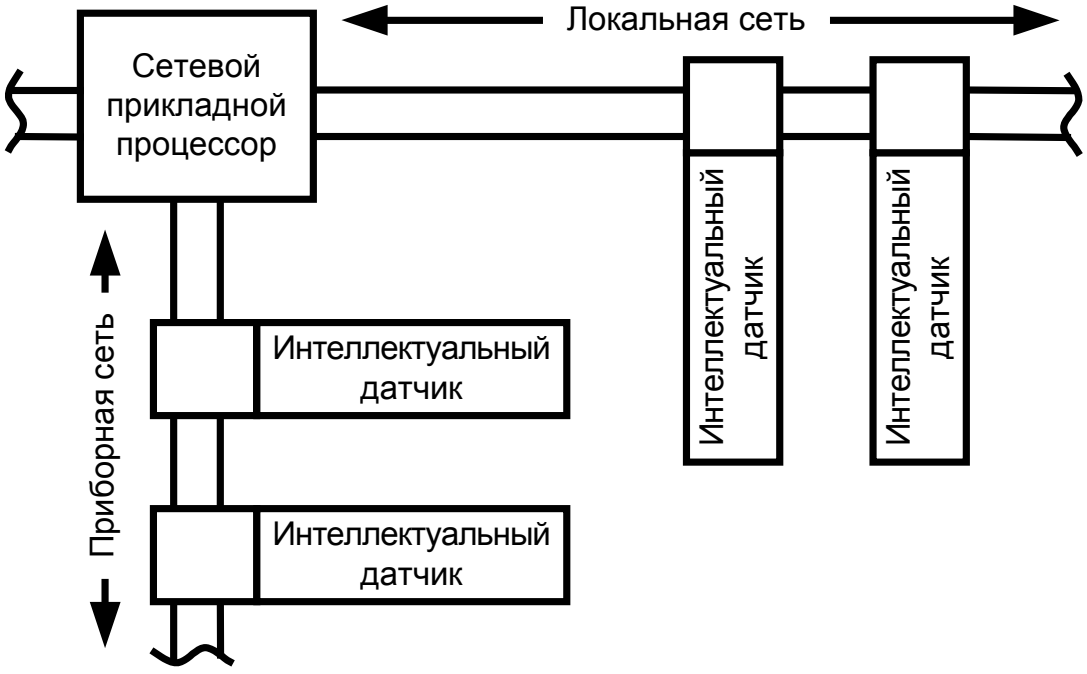


Рис. 2.45. Стандарт подключения датчика IEEE1451.2

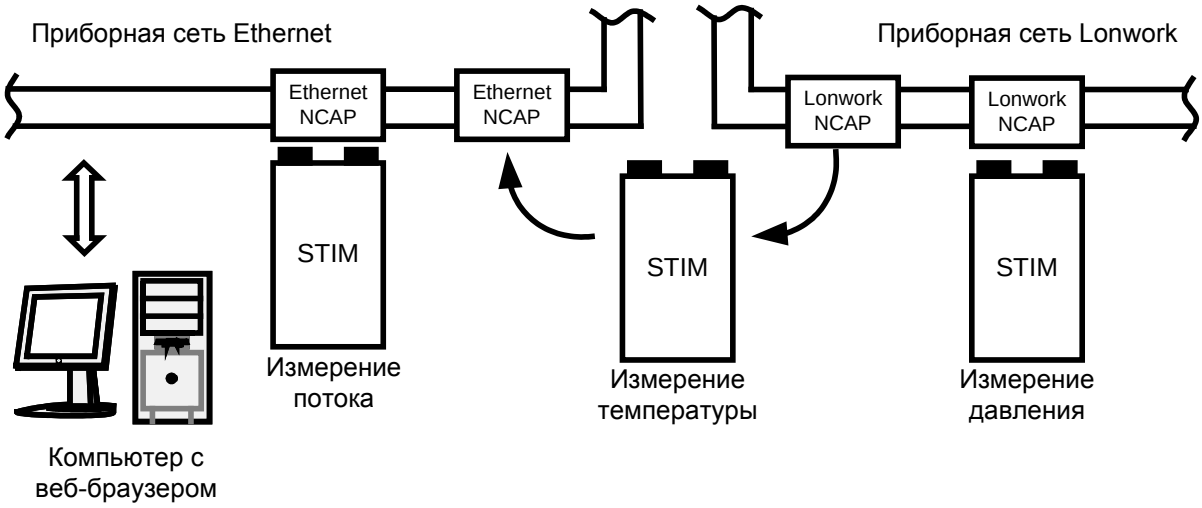


Рис. 2.46. Режим автоконфигурации, plug & play

Основой IEEE1451.2 является стандартный 10-проводной последовательный интерфейс между датчиком и узлом сети, называемый ТП (Transducer Independent Interface) (интерфейс независимый от преобразователя – ИНП). На объектах с разветвленными сетями интерфейс (ТП) позволяет устанавливать любой модуль STIM на любой узел NCAP любой сети, как показано на рис. 2.46. Когда модуль STIM первый раз подключается к новому узлу NCAP, цифровая информация модуля, включая его таблицы TEDS, становится доступной для данной сети. Сеть идентифицирует, какой тип датчика или исполнительного устройства был только что подключен, какие из его данных доступны и каковы размерности входных и выходных данных (кубические метры в секунду, градусы Кельвина, килопаскали и т. д.), какова специфицированная точность устройства (например, ±2 °C), и прочую информацию, касающуюся датчика или исполнительного устройства. Такой прием исключает необходимость выполнения программных шагов по конфигурированию сети, которые требуются при замене или добавлении датчика в систему, реализуя тем самым работу в режиме «устанавливай и работай» вне зависимости от сети.

Большинство интеллектуальных датчиков (не ограниченных модулями под 1451.2) содержат следующие основные компоненты:

микроконтроллер,
АЦП высокого разрешения,
прецизионный усилитель,
датчики.

2.8.3. MicroConverter™

Семейство изделий MicroConverter™ от фирмы Analog Devices – первые устройства, которые содержат все указанные компоненты на одном кристалле (табл. 2.7).

Таблица 2.7

Микроконвертеры Analog Devices

ADuC816	ADuC812	AduC810
Сдвоенный ZA АЦП РПП > 1 6 разрядов С/Ш (р-р) > 100 дБ Дифферент. входы Усилитель с ПУ Самокалибровка	8-канальный АЦП с РПП 12 разрядов, 5 мкс < 1/2 МЗР INL Наличие режима ПДП Самокалибровка	8-канальный АЦП с 10 разрядов < 1/2 МЗР INL
12-разрядный ЦАП Вольтовый выход < 1/2 МЗР DNL	Два 12-разрядных ЦАП Вольтовый выход < 1/2 МЗР DNL	12-разрядный ЦАП Вольтовый выход < 1/2 МЗР DNL
Наличие встроенного ИОН	Наличие встроенного ИОН	Наличие встроенного ИОН
Наличие встроенного датчика температуры	Наличие встроенного датчика температуры	Наличие встроенного датчика температуры

Три основные составляющие каждого устройства из серии MicroConverter™: высокое разрешение при аналого-цифровом и цифроаналоговом преобразовании, наличие неразрушаемой постоянной памяти (FLASH EEPROM) программ и данных и наличие микроконтроллера. Все три устройства содержат 12-разрядный ЦАП с выходом в виде напряжения, прецизионный источник опорного напряжения по запрещенной зоне и встроенный датчик температуры.

Контрольные вопросы к главе 2

1. Дайте определение понятию «датчик», перечислите основные типы датчиков.
2. Приведите примеры резистивных датчиков и систем нормализации сигнала с помощью моста Уитстона.
3. Рассмотрите известные датчики для измерения величины силы (давления).
4. Перечислите основные типы датчиков для измерения температуры.
5. Раскройте сущность метода компенсации холодного спая.
6. Проведите сравнительный анализ резистивных, полупроводниковых датчиков температуры и термисторов.
7. Приведите примеры использования датчика температуры с цифровым выходом.
8. Рассмотрите промышленные стандарты сетей датчиков.

3. МИКРОПРОЦЕССОРНЫЕ СИСТЕМЫ НА ОСНОВЕ МИКРОКОНТРОЛЛЕРОВ ATMEL

3.1. Общие сведения

Фирма Atmel Corp. (США), основанная в 1984 г. и имеющая достаточно широкий спектр деятельности в областях науки и техники, одним из основных направлений считает развитие вычислительных систем управления и контроля.

В настоящий момент Atmel является одной из ведущих корпораций в области микроэлектроники таких направлений, как микроконтроллеры, производство энергонезависимых схем памяти, микросхем программируемой логики.

Таблица 3.1

Микроконтроллеры Atmel AVR серия Classic

Наименование	Flash ROM	EEPROM	RAM	Внешняя RAM	ISP	I/O (Pins)	8/16-bit Timer	AC	ADC	Вн. RC-ген-р	WDT	BDC	UART	SPI	RTC	Кол. команд	Vcc (V)	Частота (МГц)
AT90S1200	1 KB	64 B			+	15	1/-	+		+	+					89	2,7-6,0	0–12
AT90S2313	2 KB	128 B	128 B		+	15	1/1	+			+		+			118	2,7-6,0	0–10
AT90LS2323	2 KB	128 B	128 B		+	3	1/-				+					118	2,7-6,0	0–4
AT90S2323	2 KB	128 B	128 B		+	3	1/-				+					118	4,0-6,0	0–10
AT90LS2343	2 KB	128 B	128 B		+	5	1/-			+	+					118	2,7-6,0	0–4
AT90S2343	2 KB	128 B	128 B		+	5	1/-			+	+					118	4,0-6,0	0–10
AT90LS4433	4 KB	256 B	128 B		+	20	1/1	+	6		+	+	+	+		118	2,7-6,0	0–4
AT90S4433	4 KB	256 B	128 B		+	20	1/1	+	6		+	+	+	+		118	4,0-6,0	0–8
AT90S8515	8 KB	512 B	512 B	64K B	+	32	1/1	+			+		+	+		118	2,7-6,0	0–8
AT90C8534	8 KB	512 B	256 B			7	1/1		6							118	3,3-6,0	0–1,5
AT90LS8535	8 KB	512 B	512 B		+	32	2/1	+	8		+		+	+	+	118	2,7-6,0	0–4
AT90S8535	8 KB	512 B	512 B		+	32	2/1	+	8		+		+	+	+	118	4,0-6,0	0–8

Таблица 3.2

Микроконтроллеры Atmel AVR серия ATMega

Наименование	Flash ROM	EEPROM	RAM	Доп. внешн. RAM	ISP	Self Prog Mem	JTAG	I/O (Pins)	8/16 Tmr	AC	АЦП	Вн. RC-ген-р	WDT	UART	SPI	I2C	RTC	Каналы ШИМ	Умножитель	Кол-во команд	Питание Vcc, В	Частота, МГц
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
ATmega8515L	8 KB	512 B	512 B	до 64К	+	+		35	1/1	+		+	+	1	+		+	3	+	130	2,7–5,5	0–8
ATmega8515	8 KB	512 B	512 B	до 64К	+	+		35	1/1	+		+	+	1	+		+	3	+	130	4,5–5,5	0–16
ATmega8535L	8 KB	512 B	512 B		+	+		35	2/1	+	8	+	+	1	+	+	+	4	+	130	2,7–5,5	0–8
ATmega8535	8 KB	512 B	512 B		+	+		35	2/1	+	8	+	+	1	+	+	+	4	+	130	4,5–5,5	0–8
ATmega8L	8 KB	512 B	1024 B		+	+		23	2/1	+	6/8	+	+	1	+	+	+	3	+	130	2,7–5,5	0–8
ATmega8	8 KB	512 B	1024 B		+	+		23	2/1	+	6/8	+	+	1	+	+	+	3	+	130	4,0–5,5	0–16
ATmega16L	16 KB	512 B	1024 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	2,7–5,5	0–8
ATmega16	16 KB	512 B	1024 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	4,0–5,5	0–16
ATmega161L	16 KB	512 B	1024 B		+	+		35	2/1	+			+	2	+		+		+	130	2,7–5,5	0–4
ATmega161	16 KB	512 B	1024 B		+	+		35	2/1	+			+	2	+		+		+	130	4,0–5,5	0–8
ATmega162V	16 KB	512 B	1024 B	до 64 К	+	+	+	35	2/2	+		+	+	2	+		+	4	+	130	1,8–3,6	0–1
ATmega162U	16 KB	512 B	1024 B	до 64 К	+	+	+	35	2/2	+		+	+	2	+		+	4	+	130	2,4–4,0	0–8
ATmega162L	16 KB	512 B	1024 B	до 64 К	+	+	+	35	2/2	+		+	+	2	+		+	4	+	130	2,7–5,5	0–8
ATmega162	16 KB	512 B	1024 B	до 64 К	+	+	+	35	2/2	+		+	+	2	+		+	4	+	130	4,0–5,5	0–16

Окончание табл. 3.2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
ATmega163L	16 KB	512 B	1024 B		+	+		32	2/1	+	8	+	+	1	+	+	+	3	+	130	2,7–5,5	0–4
ATmega163	16 KB	512 B	1024 B		+	+		32	2/1	+	8	+	+	1	+	+	+	3	+	130	4,0–5,5	0–8
ATmega169L	16 KB	512 B	1袱B		+	+	+	53	2/1	+	8	+	+	1	+		+	4	+	130	1,8–3,6	0–4
ATmega169V	16 KB	512 B	1袱B		+	+	+	53	2/1	+	8	+	+	1	+		+	4	+	130	2,7–3,6	0–1
ATmega323L	32 KB	1024 B	2048 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	2,7–5,5	0–4
ATmega323L	32 KB	1024 B	2048 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	4,0–5,5	0–8
ATmega32L	32 KB	1024 B	2048 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	2,7–5,5	0–8
ATmega32	32 KB	1024 B	2048 B		+	+	+	32	2/1	+	8	+	+	1	+	+	+	4	+	130	4,0–5,5	0–16
ATmega64L	64 KB	2048 B	4096 B	до 64 К	+	+	+	53	2/2	+	8	+	+	2	+	+	+	8	+	130	2,7–5,5	0–8
ATmega64	64 KB	2048 B	4096 B	до 64 К	+	+	+	53	2/2	+	8	+	+	2	+	+	+	8	+	130	4,0–5,5	0–16
ATmega103L	128 KB	4096 B	4000 B	до 64 К	+			48	2/1	+	8		+	1	+		+			121	2,7–3,6	0–4
ATmega103	128 KB	4096 B	4000 B	до 64 К	+			48	2/1	+	8		+	1	+		+			121	4,0–5,5	0–6
ATmega128L	128 KB	4096 B	4096 B	до 64 К	+	+	+	53	2/2	+	8	+	+	2	+	+	+	8	+	133	2,7–5,5	0–8
ATmega128	128 KB	4096 B	4096 B	до 64 К	+	+	+	53	2/2	+	8	+	+	2	+	+	+	8	+	133	4,0–5,5	0–16

Таблица 3.3

Микроконтроллеры Atmel AVR серия Tiny

Наименование	Flash ROM	EEPROM	RAM	ISP	I/O (Pins)	8/16-bit Timer	AC	USI	Каналы ADC	Вн. RC-цепочка	WDT	BDC	Кол-во команд	ПитаниеVcc, В	Частота МГц
ATtiny11L	1 KB			+	6	1/-	+			+	+		90	2.7–5.5	0–2
ATtiny11	1 KB			+	6	1/-	+			+	+		90	4.0–5.5	0–6
ATtiny12 V	1 KB	64 B		+	6	1/-	+			+	+		90	1.8–5.5	0–1
ATtiny12L	1 KB	64 B		+	6	1/-	+			+	+		90	2.7–5.5	0–4
ATtiny12	1 KB	64 B		+	6	1/-	+			+	+		90	4.0–5.5	0–8
ATtiny13L	1 KB	64 B	64 B	+	6	1/-	+			+	+		90	2.7–5.5	0–4
ATtiny15L	1 KB	64 B		+	6	2/-	+		4	+	+	+	90	2.7–5.5	1.6
ATtiny26L	2 KB	128 B	128 B	+	16	2/-	+	+	11	+	+	+	118	2.7–5.5	0–8
ATtiny26	2 KB	128 B	128 B	+	16	2/-	+	+	11	+	+	+	118	4.5–5.5	0–16
ATtiny28 V	2 KB				20	1/-	+			+	+		90	1.8–5.5	0–1
ATtiny28L	2 KB				20	1/-	+			+	+		90	2.7–5.5	0–4

Изначально появившиеся микроконтроллеры серии AT89 были довольно удачной копией Intel MSC51, но впоследствии оказалось, что копия была более чем удачной в основном за счет пониженного энергопотребления и все больше разработчиков стали ориентироваться именно на Atmel.

В научно-исследовательском центре Atmel в Норвегии группой разработчиков (Alf Bogen и Vergard Wollan) была предложена новая концепция 8-разрядных МК, базирующаяся на RISC-архитектуре, эти идеи легли в основу создания RISC AVR-микроконтроллеров. Результатом разработок стала серия AT90. Контроллеры разрабатывались не только для создания программ на языке низкого, но и высокого уровня. В качестве языка высокого уровня был взят Си, поскольку именно Си является самым широкораспространенным языком высокого уровня для разработки приложений для микроконтроллеров. Результатом проведения специальных научных исследований стало то, что получаемый объектный код для контроллера при трансляции с языка Си практически не обладает избыточностью. Флеш-память микро-

контроллеров, созданных по КМОП-технологии, можно программировать посредством ISP, что допускает многократное перезаписывание памяти без каких-либо дополнительных устройств.

Микроконтроллеры AVR являются достаточно мощными, низкопотребляющими и гибкими устройствами, что в совокупности с низкой стоимостью определило их широкое распространение на мировом и российском рынках. В настоящее время AVR-контроллеры используются практически во всех областях производства: от Smart Card для персональных компьютеров до спутниковых навигационных систем.

Различают следующие группы AVR-контроллеров:

Mega AVR (префикс ATmegaXXX);

Classic AVR (префикс AT90SXXX);

Tiny AVR (префикс ATtinyXXX);

AVR для Smart Cards (префикс AT90SCC).

Mega AVR имеют наибольшие объемы памяти, наибольшее количество выводов и наиболее полный набор периферийных узлов.

Classic AVR – группа, которая содержит микроконтроллеры с различным сочетанием периферийных узлов, имеет разные объемы встроенной памяти и количество выводов.

Tiny AVR – дешевые кристаллы в 8-выводных корпусах, способные работать от источника пониженного напряжения и при этом обладающие такими функционально важными периферийными узлами, как, например, АЦП.

AVR для Smart Cards – группа специализированных кристаллов, предназначенных для работы в составе периферийных и сетевых адаптеров.

В табл. 3.1, табл. 3.2, табл. 3.3 приведены характеристики основных групп МК Atmel AVR.

3.2. Организация ядра AVR-контроллеров

Улучшенная RISC (enhanced RISC) архитектура AVR-микроконтроллеров (рис. 3.1) объединяет комплекс решений, направленных на повышение быстродействия микропроцессорного ядра AVR. Арифметико-логическое устройство (ALU), в котором выполняются все вычислительные операции, имеет доступ к 32 оперативным регистрам, объединенным в регистровый файл. Выборка содержимого регистров, выполнение операции и запись результата обратно в регистровый файл выполняются за один машинный цикл.

Основной идеей всех RISC (Reduced Instruction Set Computer), как известно, является увеличение быстродействия за счет сокращения количества операций обмена с памятью программ. Для этого каждую команду стремятся уместить в одну ячейку памяти. При ограниченной разрядности ячейки памяти это неизбежно приводит к сокращению набора команд микропроцессора.

У AVR-микроконтроллеров в соответствии с этим принципом практи-

чески все команды (исключая те, у которых одним из операндов является 16-разрядный адрес) также упакованы в одну ячейку памяти программ. Но сделать это удалось не за счет сокращения количества команд процессора, а путем расширения ячейки памяти программ до 16 разрядов. Такое решение является причиной богатства системы команд AVR по сравнению с другими RISC-микроконтроллерами.

Организация памяти AVR выполнена по схеме гарвардского типа, в которой разделены не только адресные пространства памяти программ и памяти данных, но также и шины доступа к ним.

Вся программная память AVR-микроконтроллеров выполнена по технологии FLASH и размещена на кристалле. Она представляет собой последовательность 16-разрядных ячеек и имеет емкость от 512 слов до 64 Кслов в зависимости от типа кристалла.

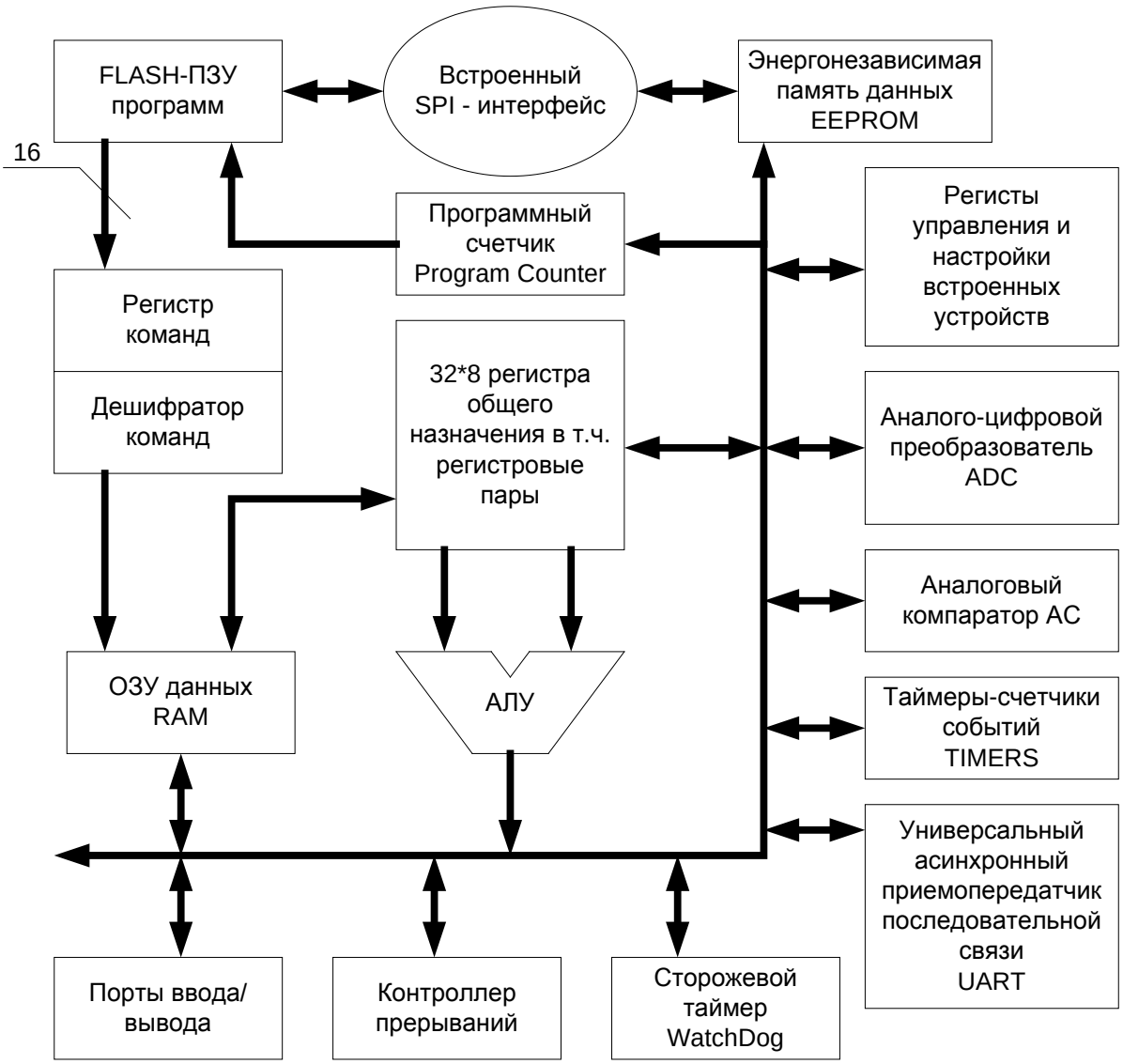


Рис. 3.1. Внутренняя организация AVR-контроллеров

Во FLASH-память кроме программы могут быть записаны постоянные данные, которые не изменяются во время функционирования микропроцессорной системы. Это различные константы, таблицы знакогенераторов, таблицы линеаризации датчиков и т. п. Достоинством технологии FLASH является высокая степень упаковки, а недостатком – то, что она не позволяет стирать отдельные ячейки. Поэтому всегда выполняется полная очистка всей памяти программ, а для AVR гарантируется, как минимум, 1000 циклов перезаписи FLASH-памяти.

Кроме того, для хранения данных AVR-микроконтроллеры могут иметь, в зависимости от типа кристалла, внутреннюю (от 0 до 4 Кбайт) и внешнюю (от 0 до 64 Кбайт) оперативную SRAM-память и энергонезависимую внутреннюю EEPROM-память (от 0 до 4 Кбайт).

Разделение шин доступа к FLASH-памяти и SRAM-памяти дает возможность иметь шины данных для памяти данных и памяти программ различной разрядности, а также использовать технологию конвейеризации. Конвейеризация заключается в том, что во время исполнения текущей команды программный код следующей уже выбирается из памяти и дешифрируется.

Для сравнения вспомним, что у микроконтроллеров семейства MCS-51 выборка кода команды и ее исполнение осуществляются последовательно, что занимает один машинный цикл, который длится 12 периодов кварцевого резонатора.

В случае использования конвейера приведенную длительность машинного цикла можно сократить. Например, у PIC-микроконтроллеров фирмы Microchip за счет использования конвейера удалось уменьшить длительность машинного цикла до четырех периодов кварцевого резонатора. Длительность же машинного цикла AVR составляет один период кварцевого резонатора. Таким образом, AVR способны обеспечивать заданную производительность при более низкой тактовой частоте. Именно эта особенность архитектуры и позволяет AVR-микроконтроллерам иметь наилучшее соотношение энергопотребление/производительность, так как потребление КМОП микросхем, как известно, определяется их рабочей частотой.

EEPROM-блок электрически стираемой памяти AVR предназначен для хранения энергонезависимых данных, которые могут изменяться непосредственно на объекте. Это калибровочные коэффициенты, различные установки, конфигурационные параметры системы. EEPROM-память имеет меньшую по сравнению с FLASH емкость (до 4 Кбайт), но при этом допускает возможность побайтной перезаписи ячеек, которая может происходить как под управлением внешнего процессора, так и под управлением собственно AVR-микроконтроллера во время его работы по программе.

Программирование энергонезависимых блоков памяти AVR может осуществляться как параллельно, так и последовательно через SPI-интерфейс (Serial Peripheral Interface).

Управление и обмен данными с EEPROM-памятью и со всеми периферийными узлами осуществляется при помощи регистров ввода/вывода, которые имеются в каждом периферийном узле.

3.3. Программная модель AVR-микроконтроллеров

На рис. 3.2 изображена программная модель AVR-микроконтроллеров, которая представляет собой диаграмму программно доступных ресурсов AVR. Главным блоком на этой диаграмме является регистровый файл из 32 оперативных регистров (R0–R31), непосредственно доступных ALU (рис. 3.3).

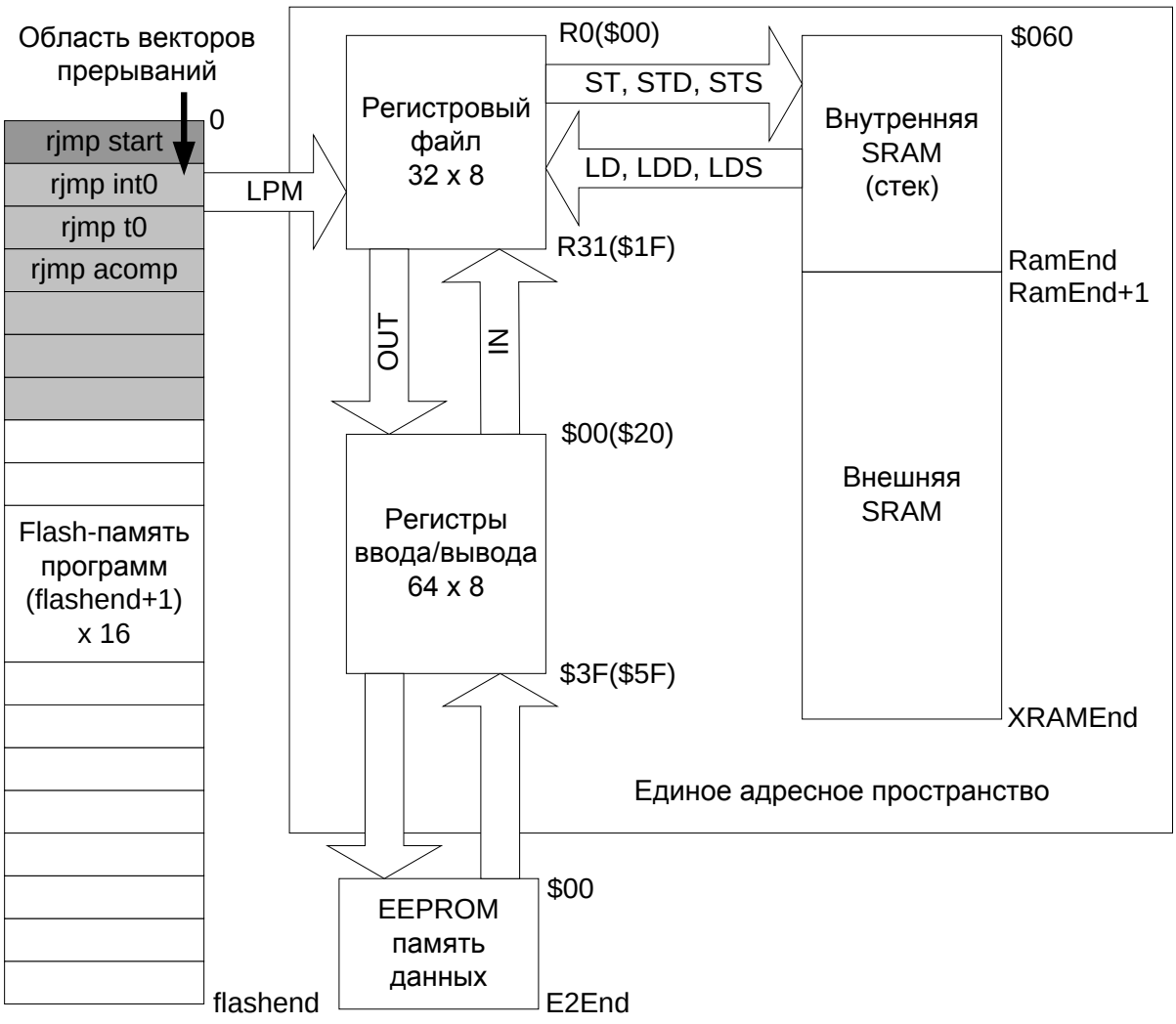


Рис. 3.2. Программная модель AVR-микроконтроллеров

Старшие регистры объединены парами и образуют три 16-разрядных регистра, предназначенных для косвенной адресации ячеек памяти (AVR без SRAM имеют только один 16-битный регистр Z).

Все арифметические и логические операции, а также часть операций работы с битами выполняются в ALU только над содержимым оперативных регистров. Следует обратить внимание на то, что команды, которые в качестве второго операнда имеют константу (SUBI, SBCI, ANDI, ORI, SBR, CBR), могут использовать в качестве первого операнда только регистры из второй половины регистрового файла (R16–R31). Команды 16-разрядного сложения с константой ADIW и вычитания константы SBIW в качестве первого операнда используют только регистры R24, R26, R28, R30.

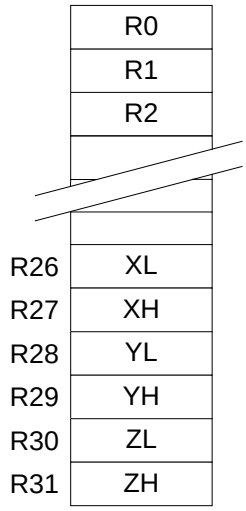


Рис. 3.3. Регистровый файл

Во время выполнения арифметических и логических операций или операций работы с битами ALU формирует те или иные признаки результата операции, т. е. устанавливает или сбрасывает биты в регистре состояния SREG (Status Register) (рис. 3.4).

I	T	H	S	V	N	Z	C
7	6	5	4	3	2	1	0

Рис. 3.4. Регистр состояния SREG (Status Register)

Бит C (carry) устанавливается, если во время выполнения операции был перенос из старшего разряда результата.

Бит Z (zero) устанавливается, если результат операции равен 0.

Бит N устанавливается, если MSB (Most Significant Bit – старший бит) результата равен 1 (правильно показывает знак результата, если не было переполнения разрядной сетки знакового числа).

Бит V устанавливается, если во время выполнения операции было переполнение разрядной сетки знакового результата.

Бит S = N + V (правильно показывает знак результата и при переполнении разрядной сетки знакового числа).

Бит H устанавливается, если во время выполнения операции был перенос из 3-го разряда результата.

Бит T (Trase) – бит трассировки, разрешение пошагового выполнения программ.

Бит I (Interupt) установливается для разрешения программных и аппаратных прерываний.

Признаки результата операции затем могут быть использованы в программе для выполнения дальнейших арифметико-логических операций или команд условных переходов.

Для хранения оперативных данных кроме регистрового файла можно использовать внутренние и внешние (если они имеются) блоки SRAM (см. рис. 3.2).

Работа с внешней SRAM может быть программно разрешена/запрещена установкой/сбросом бита SRE в регистре ввода/вывода MCUSR.

Операции обмена с внутренней оперативной памятью AVR-микроконтроллер выполняет за два машинных цикла. Доступ к внешней SRAM требует одного дополнительного цикла на каждый байт по сравнению

с внутренней памятью. Кроме того, установкой бита SRW в регистре ввода/вывода MCUSR можно программно увеличить время обмена с внешней SRAM еще на один дополнительный машинный цикл ожидания.

Выполнять арифметико-логические операции и операции сдвига непосредственно над содержимым ячеек памяти нельзя. Нельзя также записать константу или очистить содержимое ячейки памяти. Система команд AVR позволяет лишь выполнять операции обмена данными между ячейками SRAM и оперативными регистрами. Достоинством системы команд можно считать разнообразные режимы адресации ячеек памяти. Кроме прямой адресации имеются следующие режимы: косвенная, косвенная с постинкрементом, косвенная с предекрементом и косвенная со смещением.

Поскольку внутренняя и внешняя SRAM входят в единое адресное пространство (вместе с оперативными регистрами и регистрами ввода/вывода), то для доступа к ячейкам внутренней и внешней памяти используются одни и те же команды.

В ячейках оперативной памяти организуется системный стек, который используется автоматически для хранения адресов возврата при выполнении подпрограмм, а также может использоваться программистом для временного хранения содержимого оперативных регистров (команды PUSH и POP). (Микроконтроллеры, не имеющие SRAM, содержат трехуровневый аппаратный стек.)

Следует иметь в виду, что если стек располагается во внешней SRAM, то вызовы подпрограмм и возвраты из них требуют двух дополнительных циклов, если бит SRW не установлен, и четырех, если установлен.

Размер стека, организуемого в оперативной памяти, ограничен лишь размерами этой памяти. Если микроконтроллер содержит на кристалле 128 байт внутренней SRAM и не имеет возможности подключения внешней SRAM, то в качестве указателя вершины стека используется регистр ввода/вывода SPL. Если есть возможность подключения внешней памяти или внутренняя память имеет размеры 256 байт и больше, то указатель стека состоит из двух регистров ввода/вывода SPL и SPH.

При занесении числа в стек автоматически выполняются следующие действия:

1. Число записывается в ячейку памяти по адресу, хранящемуся в указателе стека (SPH:SPL) ← число.
2. Содержимое указателя стека уменьшается на единицу. SPH:SPL = SPH:SPL – 1.

Обратные действия выполняются при извлечении числа из стека:

1. Содержимое указателя увеличивается на единицу. SPH:SPL = SPH:SPL + 1.
2. Число извлекается из ячейки памяти с адресом, хранящимся в указателе стека (SPH:SPL) → число.

Таким образом, стек растет от старших адресов к младшим, поэтому учитывая, что начальное значение указателя стека после сброса равно нулю, необходимо в инициализирующей части программы установить указатель

стека, если предполагается использование хотя бы одной подпрограммы.

Регистры ввода/вывода, также изображенные на рис. 3.2, представляют собой набор регистров управления процессорного ядра и регистров управления и данных аппаратных узлов AVR-микроконтроллера. Регистрами ввода/вывода являются упоминавшиеся регистры SREG, MCUSR и указатель стека SPH:SPL, а также регистры, управляющие системой прерывания микроконтроллера, режимами подключения EEPROM памяти, сторожевым таймером, портами ввода/вывода и другими периферийными узлами. Изучение данных регистров удобно выполнять одновременно с изучением конкретного периферийного узла.

Все регистры ввода/вывода могут считываться и записываться через оперативные регистры при помощи команд IN, OUT (см. группу команд передачи данных). Регистры ввода/вывода, имеющие адреса в диапазоне $00–$1F (знак $ указывает на шестнадцатеричную систему счисления), обладают возможностью побитовой адресации. Непосредственная установка и сброс отдельных разрядов этих регистров выполняется командами SBI CBI (см. группу команд работы с битами). Для признаков результата операции, которые являются битами регистра ввода/вывода SREG, имеется целый набор команд установки и сброса. Команды условных переходов в качестве своих операндов могут иметь как биты-признаки результата операции, так и отдельные разряды побитно адресуемых регистров ввода/вывода.

Регистровый файл, блок регистров ввода/вывода и оперативная память, как показано на рис. 3.3, образуют единое адресное пространство, что дает возможность при программировании обращаться к 32 оперативным регистрам и к регистрам ввода/вывода как к ячейкам памяти, используя команды доступа к SRAM (в том числе и с косвенной адресацией).

На рис. 3.3 показано распределение адресов в едином адресном пространстве. Младшие 32 адреса ($0–$1F) соответствуют оперативным регистрам. Следующие 64 адреса ($20–$5F) зарезервированы для регистров ввода/вывода. Внутренняя SRAM у всех AVR начинается с адреса $60.

Таким образом, регистры ввода/вывода имеют двойную нумерацию. Если используются команды IN, OUT, SBI, CBI, SBIC, SBIS, то следует применять нумерацию регистров ввода/вывода, начинающуюся с нуля (назовем ее основной). Если же к регистрам ввода/вывода доступ осуществляется как к ячейкам памяти, то необходимо использовать нумерацию единого адресного пространства оперативной памяти данных AVR. Очевидно, что адрес в едином адресном пространстве памяти данных получается путем прибавления числа $20 к основному адресу регистра ввода/вывода.

Следует отметить, что регистры ввода/вывода не полностью используют отведенные для них 64 адреса. Неиспользуемые адреса зарезервированы для будущих применений, дополнительных ячеек памяти по этим адресам не существует.

Следует также иметь в виду, что у разных типов AVR одни и те же регистры ввода/вывода могут иметь различные адреса. Для того чтобы обеспечить переносимость программного обеспечения с одного типа кристалла на другой, следует использовать в программе стандартные, принятые в ориги-

нальной фирменной документации символические имена регистров ввода/вывода, а соответствие этих имен реальным адресам задавать, подключая в начале своей программы (при помощи директивы ассемблера .INCLUDE) файл определения адресов регистров ввода/вывода.

Файлы определения адресов регистров ввода/вывода имеют расширение .inc. Они уже созданы разработчиками фирмы ATMEL и свободно распространяются. В этих файлах задается соответствие символических имен основным адресам регистров ввода/вывода. Если для обращения к регистру ввода/вывода используются команды обмена с SRAM, то к символическому имени необходимо прибавить число $20.

Кроме оперативной памяти программно доступными ресурсами микроконтроллера являются энергонезависимые, электрически программируемые FLASH- и EEPROM-блоки памяти, которые имеют отдельные адресные пространства.

Так как все команды AVR представляют собой 16-разрядные слова, FLASH-память организована как последовательность 16-разрядных ячеек и имеет емкость от 512 слов до 64 K слов в зависимости от типа кристалла.

Во FLASH-память кроме программы могут быть записаны постоянные данные, которые не изменяются во время функционирования микропроцессорной системы. Это различные константы, таблицы знакогенераторов, таблицы линеаризации датчиков и т. п. Данные из FLASH-памяти могут быть программным образом считаны в регистровый файл при помощи команд LPM, ELPM (см. группу команд передачи данных).

Младшие адреса памяти программ имеют специальное назначение. Адрес $0000 является адресом, с которого начинает выполняться программа после сброса процессора. Начиная со следующего адреса $0001, ячейки памяти программ образуют область векторов прерывания. В этой области для каждого возможного источника прерывания отведен свой адрес, по которому (в случае использования данного прерывания) размещают команду относительного перехода RJMP на подпрограмму обработки прерывания (рис. 3.3). Следует помнить, что адреса векторов прерывания одних и тех же аппаратных узлов для разных типов AVR могут иметь разное значение. Поэтому для обеспечения переносимости программного обеспечения удобно, так же как и в случае с регистрами ввода/вывода, использовать символические имена адресов векторов прерывания, которые определены в соответствующем inc-файле.

EEPROM-блок электрически стираемой памяти данных AVR предназначен для хранения энергонезависимых данных, которые могут изменяться непосредственно на объекте. Это калибровочные коэффициенты, различные установки, конфигурационные параметры системы и т. п. EEPROM-память данных может быть программным путем как считана, так и записана. Однако специальных команд обращения к EEPROM-памяти нет. Чтение и запись ячеек EEPROM выполняется через регистры ввода/вывода EEAR (регистр адреса), EEDR (регистр данных) и EECR (регистр управления).

Периферийные устройства AVR

В состав периферийных устройств AVR-контроллеров, в зависимости от типа кристалла, могут входить следующие периферийные узлы:

таймер/счетчик разрядностью 8 бит;

сторожевой таймер;

n-разрядный таймер/счетчик с возможностью организации функций ШИМ и захвата/сравнения;

аналого-цифровой преобразователь;

аналоговый компаратор;

скоростной последовательный интерфейс SPI;

асинхронный дуплексный последовательный порт UART;

контроллер прерываний;

внутренний тактовый генератор;

сторожевой (WATCHDOG) таймер;

порты ввода/вывода и т. д.

3.4. Исполнительные модули AVR

Таймер/счетчик событий – внутренний таймер, предназначенный для запуска программы обработки прерывания при определенных условиях счета, в том числе и внешних событий.

Сторожевой таймер предназначен для защиты микроконтроллера от сбоев в процессе работы. При срабатывании сторожевого таймера происходит внутренний перезапуск работы микроконтроллера.

Порты ввода/вывода AVR имеют от 5 до 32 независимых линий ввода/вывода, причем каждый разряд любого порта может быть запрограммирован на ввод или на вывод.

Аналого-цифровой преобразователь – это 10-разрядный АЦП с устройством выборки/хранения и входным аналоговым мультиплексором.

Аналоговый компаратор предназначен для сравнения непрерывно изменяющихся сигналов. Входные аналоговые сигналы компаратора $U_{вх}$ – анализируемый сигнал и $U_{оп}$ – опорный сигнал сравнения, а выходной $U_{вых}$ – дискретный или логический сигнал, содержащий 1 бит информации *n таймер/счетчик* – дополнительные таймера/счетчики. Кроме того, может содержать широтно-импульсный модулятор (ШИМ), предназначенный для формирования сигналов заданной длительности при определенных условиях.

Скоростной последовательный интерфейс SPI – последовательный синхронный интерфейс ввода/вывода, используется для передачи данных по протоколу SPI.

Асинхронный, дуплексный последовательный порт UART – последовательный порт ввода/вывода информации.

Контроллер прерываний предназначен для организации векторной системы прерываний контроллера. Различают как внутренние, так и внешние источники прерываний.

Внутренний тактовый генератор – это внутренняя RC-цепочка, являющаяся схемой тактирования контроллера. В последних разработках внутренняя тактовая частота контроллера может задаваться программно, т. е. происходит настройка параметров этой цепочки. Кроме того, можно подключать и внешний источник тактовых сигналов.

Рассмотрение периферийных узлов AVR-микроконтроллеров проведем на примере микроконтроллеров серии Classic AT90S4434/8535.

3.5. Порты ввода/вывода

Устройства AT90S4434/8535 имеют в своем составе четыре восьмиразрядных квазидвунаправленных порта ввода/вывода. Термин «квази» означает, что каждая линия порта может быть настроена как на ввод, так и на вывод. Настройка линий порта осуществляется записью управляющих слов в регистр управления портом (DDRx). При записи в соответствующий бит нуля данная линия порта настраивается на ввод. При установке соответствующего бита в регистре управления в единицу линия порта настраивается на вывод.

Данные, предназначенные для передачи из порта или принятые в порт от внешнего источника, хранятся в регистре данных порта (PORTx). Кроме того, выводы портов используются как альтернативные входы/выходы дополнительных устройств микроконтроллера (ADC, UART и т. д.). При разрешении работы соответствующего устройства, установке битов разрешения работы в регистре управления данным устройством линии порта автоматически перенастраиваются и становятся линиями обмена с данным устройством.

3.6. Таймеры/счетчики

Устройства AT90S4434/8535 оснащены тремя таймерами/счетчиками общего назначения – двумя 8-разрядными и одним 16-разрядным. Таймер/счетчик 2 дополнительно может тактироваться асинхронно от внешнего генератора. Этот генератор оптимизирован под использование кварцевого кристалла на частоту 32.768 кГц, что позволяет использовать таймер/счетчик 2 как часы реального времени (RealTimeClock-RTC).

Таймеры/счетчики 0 и 1 используют общий 10-разрядный предварительный делитель опорной частоты. Таймер/счетчик 2 оснащен своим собст-

венным предварительным делителем. Эти таймеры/счетчики можно использовать как таймеры с встроенной временной базой или как счетчик, переключаемый по состоянию на внешнем выводе.

3.6.1. Предварительные делители частоты таймеров/счетчиков

Предварительный делитель таймеров/счетчиков 0 и 1 содержит четыре ступени деления: СК/8, СК/64, СК/256 и СК/1024, где СК – входной тактовый сигнал. В качестве источников тактовых сигналов могут быть использованы сигналы от внешних источников, тактовый сигнал СК и нулевой тактовый сигнал (stop).

8-разрядный таймер/счетчик 0 (T/C0)

Для T/C0 источником тактирования могут быть избраны СК, масштабированный СК или СК от внешнего вывода. Таймер/счетчик может быть остановлен, как показано в описании регистров управления таймерами/счетчиками TCCR0 (рис. 3.5).

Флаг переполнения T/C0 находится в регистре флагов прерывания таймеров (TIFR). Установки управляющих сигналов хранятся в регистрах управления таймерами/счетчиками TCCR0. Установка разрешения/запрещения прерываний производится в регистре масок прерываний таймеров/счетчиков TIMSK (Timer/Counter Interrupt Mask Register).

3.6.2. Регистр управления таймером/счетчиком 0-TCCR0

На рис. 3.5 изображен регистр управления таймером/счетчиком 0.
В табл. 3.4 приведены варианты настройки таймера/счетчика 0.

$33(53)

биты	7	6	5	4	3	2	1	0
	-	-	-	-	-	CS02	CS01	CS00
Доступ	Чт	Чт	Чт	Чт	Чт	Чт/Зап	Чт/Зап	Чт/Зап
Нач. состояние	0	0	0	0	0	0	0	0

Рис. 3.5. Регистр управления таймером/счетчиком 0

Биты 7...3 – зарезервированные биты в 4434/8535 всегда читаются как 0.
Биты 2, 1, 0 – определяют источник тактирования T/C0.

Таблица 3.4

Настройка таймера/счетчика 0

Cs02	cs01	cs00	Описание
0	0	0	Стоп
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	Внешний вход T0, падающий фронт
1	1	1	Внешний вход T0, нарастающий фронт

Условие СТОП обеспечивает функцию включения/выключения таймера. Режимы деления СК масштабируют непосредственно импульсы тактового генератора. Если используется режим тактирования от внешнего входа, соответствующая установка должна выполняться в соответствующем регистре управления направлением данных (DDRx) (сброс в нуль настраивает пин на ввод).

3.6.3. Регистр данных таймера/счетчика 0-TCNT0

На рис. 3.6 изображен регистр данных таймера/счетчика 0.

$32(52)

биты	7	6	5	4	3	2	1	0	
	MSB	-	-	-	-	-	-	-	LSB
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	
Нач. состояние	0	0	0	0	0	0	0	0	

Рис. 3.6. Регистр данных таймера/счетчика 0

T/C0 реализован как счетчик, считающий вверх, с доступом по чтению и записи. Если в таймер/счетчик записано некоторое значение и выбран источник тактового сигнала, то он продолжит счет с записанного значения с тактовой частотой счетчика.

16-разрядный таймер/счетчик 1

16-разрядный таймер/счетчик 1 может получать тактовый сигнал от CK, CK после предварительного делителя и от внешнего вывода. В регистрах управления TCCR1A и TCCR1B находятся различные флаги, указывающие на переполнение, совпадение при сравнении и случаи захвата событий. В регистре масок прерываний TIMSK устанавливаются разрешения/запрещения прерываний таймера/счетчика 1. При внешнем тактировании таймера/счет-

чика 1 внешний сигнал синхронизируется частотой тактового генератора CPU.

Таймер/счетчик 1 поддерживает две функции сравнения выхода, используя регистр сравнения выходов A и B – OCR1A и OCR1B в качестве источников данных, сравниваемых с содержимым таймера/счетчика 1. Функции сравнения выхода включают очистку счетчика по совпадению сравнения и воздействие на выводы сравнения выхода при обоих совпадениях сравнения. Таймер/счетчик 1 может быть использован в качестве 8-, 9- или 10-разрядного широтно-импульсного модулятора. В этом режиме счетчик и регистры OCR1A/OCR1B работают как сдвоенный самостоятельный ШИМ со сцентрированными импульсами, без формирования ложных импульсов.

Функция захвата входа таймера/счетчика 1 обеспечивает захват содержимого таймера/счетчика 1 в регистр захвата входа (ICR1), запускаемый внешним событием на выводе входа захвата (ICP). Реальные установки захвата события определяются регистром управления таймером/счетчиком 1 TCCR1B (Timer/Counter1 Control Register). Кроме того, для переключения входа захвата может быть использован аналоговый компаратор.

Если разрешена функция подавления шума, действительные условия переключения события захвата тестируются четырьмя выборками, прежде чем захват будет активирован. Тестирование сигнала на входном выводе производится с частотой XTAL.

3.6.4. Управляющий регистр A таймера/счетчика 1-TCCR1A

На рис. 3.7 изображен управляющий регистр A таймера/счетчика 1.

$2F(4F)								
биты	7	6	5	4	3	2	1	0
	COM1A1	COM1A0	COM1B1	COM1B0	-	-	PWM11	PWM10
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт	Чт	Чт/Зп	Чт/Зп
Нач.состояние	0	0	0	0	0	0	0	0

Рис. 3.7. Управляющий регистр A таймера/счетчика 1

Биты 7, 6 – COM1A1, COM1A0; – режим 1 A сравнения выхода.

Управляющие биты COM1A1 и COM1A0 определяют характер сигнала выхода, следующего за совпадением при сравнении таймера/счетчика 1. Сигнал выхода поступает на вывод OC1A (Output Compare A). Поскольку это альтернативная функция порта I/O, то соответствующий бит управления на-

правлением должен быть установлен в 1 (вывод работает на выход). Конфигурирование управления представлено в табл. 3.5.

Биты 5, 4 – COM1B1, COM1B0; – режим 1 B сравнения выхода.

Управляющие биты COM1B1 и COM1B0 определяют характер сигнала выхода, следующего за совпадением при сравнении таймера/счетчика 1. Сигнал выхода поступает на вывод OC1B (Output CompareB). Поскольку это альтернативная функция порта I/O, то соответствующий бит управления направлением должен быть установлен в 1 (вывод работает на выход). Конфигурирование управления представлено в табл. 3.5.

Таблица 3.5

Выбор режима сравнения 1

COM1X1	COM1X0	**Описание**
0	0	Таймер/счетчик 1 отключен от вывода выхода OC1X
0	1	Переключение выходной линии OC1X
1	0	Очистка выходной линии OC1X (в 0)
1	1	Установка выходной линии OC1X (в 1)

Примечание. X = A или B.

Таблица 3.6

Настройка режима ШИМ

PWM1	***PWM1***	Описание
0	0	Работа таймера/счетчика 1 в режиме ШИМ запрещена
0	1	Работа таймера/счетчика 1 в 8-разрядном режиме ШИМ
1	0	Работа таймера/счетчика 1 в 9-разрядном режиме ШИМ
1	1	Работа таймера/счетчика 1 в 10-разрядном режиме ШИМ

В режиме ШИМ функции этих битов различаются. Подробное описание приведено в табл. 3.6.

Биты 3..2 – зарезервированные биты.

Эти биты в микроконтроллерах AT90S4434/8535 зарезервированы и при считывании всегда будут 0.

Биты 1..0 – PWM11, PWM10: – биты выбора режима ШИМ.

Данные биты определяют установку режима ШИМ, как это показано в табл. 3.6.

3.6.5. Управляющий регистр B таймера/счетчика 1 – TCCR1B

На рис. 3.8 изображен управляющий регистр B таймера/счетчика 1.

$2E(4E)

биты	7	6	5	4	3	2	1	0
	ICNC1	ICES1	-	-	CTC1	CS12	CS11	CS10
Доступ	Чт/Зп	Чт/Зп	Чт	Чт	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Нач.состояние	0	0	0	0	0	0	0	0

Рис. 3.8. Управляющий регистр B таймера/счетчика 1

Бит 7 – ICNC1: – установка режима подавления шума на входе захвата 1.

При сброшенном в состояние 0 бите ICNC1 функция подавления шума входного триггера захвата запрещена. Вход захвата переключается по первому нарастающему/падающему фронту, поступившему на вывод входа/захвата (ICP).

При установленном в состояние 1 бите ICNC1 выполняются четыре последовательных опроса состояния вывода (ICP) и все четыре выборки должны иметь одинаковый (высокий/низкий), определяемый битом ICES1, уровень. Частота опроса соответствует частоте XTAL

Бит 6 – ICES1: – выбор фронта срабатывания на входе захвата 1.

При сброшенном в состояние 0 бите ICES1 содержимое таймера/счетчика 1 по падающему фронту на выводе входа захвата (ICP) пересылается в регистр захвата входа ICR1. При установленном в 1 бите ICES1 содержимое таймера/счетчика 1 пересылается в регистр захвата входа ICR1 по нарастающему фронту на выводе входа захвата (ICP).

Биты 5, 4 – зарезервированные биты.

Эти биты в микроконтроллерах AT90S4434/8535 зарезервированы и при считывании всегда будут 0.

Бит 3 – CTC1: – очистка таймера/счетчика 1 по совпадению.

При установленном в состояние 1 бите CTC1 таймер/счетчик 1 сбрасывается в состояние $0000 в течение тактового цикла, следующего за совпадением при сравнении. Если бит CTC1 очищен, таймер/счетчик 1 продолжает отсчет и не реагирует на совпадение при сравнении. Поскольку совпадение при сравнении детектируется в течение тактового цикла CPU, следующего за совпадением, то поведение функции будет отличаться при установке коэффициента предварительного деления таймера/счетчика большем 1. При коэффициенте предварительного деления 1 и установленном в регистре сравнения состоянии C таймер будет считать в соответствии с установкой CTC1:

... | C-2 | C-1 | C | 0 | 1 |...

При установленном коэффициенте предварительного деления 8 таймер будет считать подобно:

... | C-2, C-2, C-2, C-2, C-2, C-2, C-2, C-2 | C-1, C-1, C-1, C-1, C-1, C-1, C-1, C-1

| C, 0, 0, 0, 0, 0, 0, 0 |1,1,1,1,1,1,1,1|...

В ШИМ режиме состояние бита CTC1 значения не имеет.

Биты 2,1,0 – CS12, CS11, CS10: – выбор источника тактовой частоты.

Установкой состояния данных битов производится выбор источника тактового сигнала (в том числе коэффициента предварительного деления). Stop выполняет функцию разрешения/запрещения таймера/счетчика 1. При использовании внешнего тактирования необходимо выполнить соответствующие установки в регистре управления направлением (очистка переводит вывод в режим входа). Режимы настройки битов приведены в табл. 3.7.

Таблица 3.7

Выбор источника тактовой частоты

cs12	cs11	cs10	Описание
0	0	0	Стоп
0	0	1	CK
0	1	0	CK/8
0	1	1	CK/64
1	0	0	CK/256
1	0	1	CK/1024
1	1	0	Внешний вход T1, падающий фронт
1	1	1	Внешний вход T1, нарастающий фронт

16-разрядный регистр данных T/C1 – TCNT1H и TCNT1L

На рис. 3.9 изображен 16-разрядный регистр данных таймера/счетчика 1. 16-разрядный регистр содержит текущее значение 16-разрядного таймера/счетчика 1. Для того чтобы CPU могло считывать/записывать и старший и младший байты этого регистра одновременно, обращение реализовано посредством 8-разрядного регистра временного хранения (TEMP). Этот регистр используется также при обращении к OCR1A, OCR1B и ICR1.

$2D(4D) TCNT1H
$2C(4C) TCNT1L

биты	15	14	13	12	11	10	9	8
	MSB							
								LSB
биты	7	6	5	4	3	2	1	0
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Нач.состояние	0	0	0	0	0	0	0	0

Рис. 3.9. 16-разрядный регистр данных таймера/счетчика 1

Если основная программа и подпрограммы обработки прерываний используют обращение к регистрам посредством TEMP, то прерывания должны быть запрещены на время обращения из основной программы.

Таймер/счетчик 1 выполнен в виде вверх считающего или реверсивного счетчика (в ШИМ режиме) с возможностью чтения/записи. Если в таймер/счетчик 1 занесено некоторое значение и выбран источник тактового сигнала, то таймер/счетчик 1 продолжит отсчет через один тактовый цикл после установки в нем записанного значения.

16-разрядные регистры сравнения выхода с доступом по чтению и записи OCR1AH и OCR1AL

Регистры сравнения выхода таймера/счетчика 1 хранят данные, постоянно сравниваемые с состоянием таймера/счетчика 1. Действие, запускаемое совпадением при сравнении, определяется содержимым регистра управления и состояния таймера/счетчика 1. Совпадение при сравнении может произойти, только если таймер/счетчик 1 досчитает до значения содержимого OCR. Если в TCNT1 и OCR1A или OCR1B программно будут записаны одинаковые значения, то совпадение при сравнении сгенерировано не будет.

Совпадение при сравнении устанавливает флаг прерывания по совпадению в тактовом цикле CPU, следующем за самим совпадением. Запись в PORTD5 и PORTD4 устанавливает соответственно значения OC1A и OC1B.

Поскольку регистры сравнения выхода OCR1A и OCR1B являются 16-разрядными, то для обеспечения одновременного занесения старшего и младшего байтов данных в регистры OCR1A/B используется регистр временного хранения TEMP. Когда CPU записывает старший байт, то данные временно сохраняются в регистре TEMP. Когда же CPU записывает младший байт в OCR1AL или OCR1BL, то одновременно содержимое регистра OCR1BH переписывается в OCR1AH или OCR1BH. Следовательно, при 16-разрядных операциях старшие байты регистров OCR1A/B должны записываться первыми.

Кроме того, регистр TEMP используется при обращении к TCNT1 и ICR1. Если основная программа и подпрограммы обработки прерываний используют обращение к регистрам посредством TEMP, то прерывания должны быть запрещены на время обращения из основной программы.

16-разрядный регистр захвата входа с доступом по чтению ICR1

При обнаружении на выводе захвата входа (ICP) нарастающего или падающего фронта сигнала (определяемого установкой ICES1) текущее состояние таймера/счетчика 1 пересылается в регистр захвата входа ICR1. Одновременно устанавливается в состояние 1 флаг захвата входа ICF1.

Поскольку регистр захвата входа является 16-разрядным, то для обеспечения одновременного чтения старшего и младшего байтов данных реги-

стра ICR1 используется регистр временного хранения TEMP. При считывании CPU данных младшего байта содержимое ICR1L пересылается в CPU, а содержимое старшего байта ICR1H размещается в регистре TEMP, чтение старшего байта заключается в переносе в CPU содержимого регистра временного хранения TEMP. Следовательно, при чтении всего 16-разрядного регистра операцию чтения необходимо начинать с младшего байта ICR1L. Регистр TEMP используется также при обращении к TCNT1, OCR1A и OCR1B. Если основная программа и подпрограммы обработки прерываний, используют обращение к регистрам посредством TEMP, то прерывания должны быть запрещены на время обращения из основной программы.

3.6.6. Таймер/счетчик 1 в режиме ШИМ

При установленном режиме ШИМ таймер/счетчик 1 и регистры сравнения выхода A и B (OCR1A и OCR1B) образуют сдвоенный 8-, 9- или 10-разрядный автономный генератор ШИМ с правильным чередованием фаз, отсутствием ложных выбросов и выходами на выводы PD5(OC1A) и PD4(OC1B). Таймер/счетчик 1 работает как реверсивный счетчик, считающий от $0000 до TOP (табл. 3.8), где направление счета меняется и отсчет ведется до нуля, после чего цикл повторяется. Когда состояние счетчика совпадет с содержимым 10 младших битов OCR1A или OCR1B, выводы PD5(OC1A)/PD4(OC1B) устанавливаются или очищаются, в соответствии с установками битов COM1A1/COM1A0 или COM1B1/COM1B0 в регистре управления таймером/счетчиком 1 TCCR1A. Более подробно см. в табл. 3.9.

Таблица 3.8

Значения TOP и частота ШИМ

Разрешение ШИМ	**Значения TOP**	**Частота**
8-разрядное	$00FF (255)	f TC1 /510
9-разрядное	$01FF (511)	f TC1 /1022
10-разрядное	$03FF (1023)	f TC1 /2046

Примечание. Если регистр сравнения содержит значение TOP и предварительный делитель частоты не используется (CS12..CS10 = 001), на выходе ШИМ не появятся импульсы, потому что значения при счете вверх и вниз достигаются одновременно. Когда предварительный делитель частоты используется (CS12..CS10 001 или 000), на выходе ШИМ появляется активность, если счетчик достигает значения TOP. При счете вниз достижение совпадения не интерпретируется, пока счетчик не достигнет значения TOP, получая один период импульса ШИМ.

Таблица 3.9

Выбор режима сравнения1 в режиме ШИМ

COM1X1	COM1X0	Выходной сигнал на OCX1
0	0	Не подключен
0	1	Не подключен
1	0	Очищается по совпадению при счете вверх. Устанавливается по совпадению при счете вниз (неинвертированный ШИМ)
1	1	Очищается по совпадению при счете вниз. Устанавливается по совпадению при счете вверх (инвертированный ШИМ)

Примечание. X = A или B.

Отметим, что в режиме ШИМ младшие 10 разрядов OCR1A/OCR1B при записи пересылаются в ячейки временного хранения. Они фиксируются по достижении таймером/счетчиком 1 значения TOP. Таким способом обеспечивается защита от появления уширенных импульсов ШИМ (ложных выбросов – glitches) при несинхронной записи OCR1A/OCR1B.

В режиме ШИМ флаг переполнения таймера 1 (TOV1) устанавливается при смене направления счета по достижении значения $0000. Прерывание по переполнению таймера 1 работает так же, как и в обычном режиме таймера/счетчика, т. е. оно выполняется, когда TOV1 установлен и разрешены прерывания по переполнению таймера 1 и глобальные прерывания. Это относится и к флагам сравнения выхода таймера 1, и прерываниям.

3.6.7. Сторожевой таймер (WDT)

Сторожевой таймер тактируется отдельным встроенным генератором. Установкой коэффициента предварительного деления интервал сброса сторожевого таймера может быть настроен, как показано в табл. 3.10. Команда WDR (Watchdog Reset) сбрасывает сторожевой таймер. Можно установить восемь периодов длительности тактового сигнала. Если период сброса завершается (в течение этого периода не поступил сигнал сброса сторожевого таймера), то микроконтроллер AT90S4434/8535 сбрасывается и его работа продолжается по вектору сброса. Чтобы предохранить непреднамеренное выключение сторожевого таймера, отключение сторожевого таймера должно сопровождаться специальной последовательностью выключения.

Регистр управления сторожевым таймером – WDTCR

На рис. 3.10 изображен регистр управления сторожевым таймером.

$21(41)

биты	7	6	5	4	3	2	1	0
	-	-	-	WDTOE	WDE	WDP2	WDP1	WDP0
Доступ	Чт	Чт	Чт	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.10. Регистр управления сторожевым таймером

Биты 7...5 – зарезервированные биты в 4434/8535 всегда читаются как 0.

Бит 4 – WDTOE: – разрешение отключения сторожевого таймера. Данный бит должен быть установлен в состояние 1 при очистке бита WDE. В ином случае сторожевой таймер не будет запрещен. Установленный бит аппаратно очищается после четырех тактовых циклов.

Бит 3 – WDE: – разрешение сторожевого таймера.

Если бит WDE установлен в состояние 1 (сторожевой таймер разрешен) и если бит WDE очищен, то функционирование сторожевого таймера запрещено. Бит WDE может быть очищен, если установлен бит WDTOE. Для запрещения разрешенного сторожевого таймера необходимо выполнить следующую процедуру:

1. В одной операции записать логическую 1 в WDTOE и WDE. Логическая 1 должна быть записана в WDE, даже если этот бит был установлен перед началом операции запрета сторожевого таймера.
2. За время последующих четырех тактовых циклов записать логический 0 в WDE. Сторожевой таймер будет запрещен.

Биты 2..0 – WDP2, WDP1, WDP0: – биты установки коэффициента предварительного деления сторожевого таймера.

Состояния битов WDP2, WDP1 и WDP0 определяют коэффициент предварительного деления тактовой частоты разрешенного сторожевого таймера. Коэффициенты и соответствующие им промежутки времени представлены в табл. 3.10.

Таблица 3.10

Коэффициент предварительного деления сторожевого таймера

WDP2	WDP1	WDP0	Количество циклов генератора WDT	Задержка V_{CC} = 3.0V	Задержка V_{CC} = 5.0V
0	0	0	16K циклов	47 мс	15 мс
0	0	1	32K циклов	94 мс	30 мс
0	1	0	64K циклов	0.19 с	60 мс
0	1	1	128K циклов	0.38 с	0.12 с
1	0	0	*256K циклов*	0.75 с	0.24 с
1	0	1	512K циклов	1.5 с	0.49 с
1	1	0	1024K циклов	3.0 с	0.97 с
1	1	1	2048K циклов	6.0 с	1.9 с

Примечание. Частота тактового генератора сторожевого таймера зависит от напряжения питания. Команда WDR (Watchdog Reset) должна всегда быть выполнена перед разрешением сторожевого таймера. Это гарантирует, что задержка будет соответствовать настройкам предварительного делителя частоты сторожевого таймера. Если не произвести сброс перед разрешением сторожевого таймера, то счет может начаться не с нуля. Чтобы избежать непреднамеренного сброса, сторожевой таймер должен быть запрещен или сброшен перед изменением настроек предварительного делителя сторожевого таймера.

3.7. Последовательный периферийный интерфейс (SPI)

Последовательный периферийный интерфейс (SPI) обеспечивает высокоскоростной синхронный обмен данными между микроконтроллерами AT90S4434/8535 и периферийными устройствами или между несколькими AVR-устройствами.

Основные характеристики SPI интерфейса:

полнодуплексный 3-проводной синхронный обмен данными;

режим работы ведущий или ведомый;

обмен данными с передаваемыми первыми старшим или младшим битами;

четыре программируемые скорости обмена данными;

флаг ошибки при записи;

активация из Idle-режима.

Соединения между ведущим и ведомым CPU, использующими SPI-интерфейс, показаны на рис. 3.11. Вывод PB7(SCK) является выходом тактового сигнала ведущего микроконтроллера и входом тактового сигнала ведомого.

По записи ведущим CPU-данных в SPI-регистр начинает работать тактовый генератор SPI и записанные данные сдвигаются через вывод выхода PB5(MOSI) ведущего микроконтроллера на вывод входа PB5 (MOSI) ведомого микроконтроллера. После сдвига одного байта тактовый генератор SPI останавливается, устанавливая флаг окончания передачи (SPIF). Если в регистре SPCR будет установлен бит разрешения прерывания SPI (SPIE), то произойдет запрос прерывания. Вход выбора ведомого PB4(SS) для выбора индивидуального SPI-устройства в качестве ведомого устанавливается на низкий уровень.

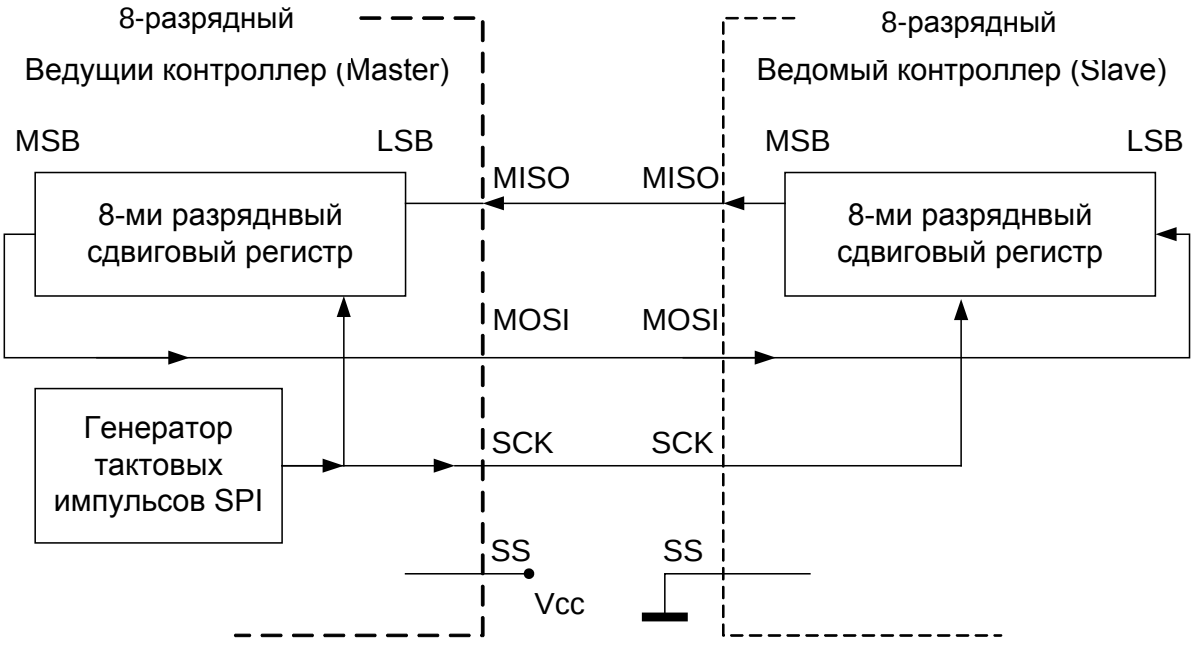


Рис. 3.11. SPI-интерфейс (подключение устройств)

Два сдвиговых регистра ведущего и ведомого микроконтроллеров можно рассматривать как один разнесенный 16-разрядный циклический сдвиговый регистр (рис. 3.11). При сдвиге данных из ведущего микроконтроллера в ведомый одновременно происходит сдвиг данных из ведомого микроконтроллера в ведущий, т. е. в течение одного цикла сдвига происходит обмен данными между ведущим и ведомым микроконтроллерами.

В системе организовано одиночное буферирование передающей стороны и двойное буферирование на приемной стороне. Это означает, что передаваемые символы не могут быть записаны в регистр данных SPI, прежде чем будет полностью завершен цикл сдвига.

Тем не менее, когда передаются данные, переданный байт должен быть считан из регистра данных SPI ранее, чем следующий байт будет полностью передан. В противном случае первый байт потерян.

3.7.1. Функционирование входа SS

При работе SPI ведущим (бит MSTR регистра SPCR установлен) пользователь имеет возможность установить направление работы вывода SS. Если вывод SS сконфигурирован как выход, то вывод является выводом общего назначения и он не активируется системой SPI. Если же вывод SS сконфигурирован как вход, то для обеспечения работы ведущего SPI он должен удерживаться на высоком уровне. Если в режиме ведущего вывод SS является входом и внешней периферийной схемой на него подан низкий уровень, то

SPI воспримет его как обращение другого ведущего SPI к себе как к ведомому. Чтобы избежать конфликтной ситуации на шине, система SPI выполняет следующие действия:

1. Бит MSTR в регистре SPCR очищается и SPI-система становится ведомой. Результатом этого является то, что MOSI- и SCK-выводы становятся входами.

2. Устанавливается флаг SPIF регистра SPSR и, если разрешено прерывание SPI, начнется выполнение подпрограммы обработки прерывания.

Таким образом, когда управляемый прерыванием передающий SPI используется в ведущем режиме и существует вероятность подачи на вывод SS управляющего сигнала низкого уровня, прерывание должно всегда проверять, установлен ли еще бит MSTR. Если же бит MSTR был очищен выбором режима ведомого, то он должен быть установлен пользователем.

Если же SPI работает в режиме ведомого, то вывод SS постоянно работает на вход. Если на вывод SS подан низкий уровень, то SPI активируется и MISO, если это определено пользователем, становится выходом. Все остальные выводы являются входами. Если вывод SS удерживается на высоком уровне, то все выводы являются входами, SPI пассивен, что означает, что он не будет получать входящих данных.

Отметим, что SPI-логика будет сброшена при приведении пина SS в высокий уровень. Если пин SS приведен в высокий уровень в течение передачи, SPI останавливает передачу и прием данных немедленно и полученные данные и посланные должны считаться потерянными.

3.7.2. Регистр управления SPI – SPCR

На рис. 3.12 изображен регистр управления SPI.

$0D(2D)
биты

	7	6	5	4	3	2	1	0
	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	*SPR0*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.12. Регистр управления SPI

Бит 7 – SPIE: – разрешение прерывания SPI.

Установка бита SPIE в состояние 1 означает, что прерывания будут выполняться, если установлен бит SPIF регистра SPSR и разрешено глобальное прерывание.

Бит 6 – SPE: – разрешение SPI.

Установка бита SPE в состояние 1 разрешает SPI-операции.

Бит 5 – DORD: – порядок данных.

При установленном в состояние 1 бите DORD передача слова данных начинается с LSB. При очищенном бите DORD первым передается MSB слова данных.

Бит 4 – MSTR: – выбор режима ведущий/ведомый.

При установленном в состояние 1 бите MSTR SPI работает в ведущем режиме и при очищенном бите – в ведомом режиме. Если SS сконфигурирован как вход и на него подан низкий уровень при установленном MSTR, то MSTR будет сброшен и будет установлен бит SPIF в регистре SPSR. Чтобы вновь разрешить ведущий режим SPI, пользователь должен установить MSTR.

Бит 3 – CPOL: – полярность тактового сигнала.

SCK в режиме ожидания находится на высоком уровне при установленном в состояние 1 бите CPOL и на низком уровне при сброшенном бите CPOL.

Бит 2 – CPHA : – фаза тактового сигнала.

Биты 1,0 – SPR1, SPR0: – выбор частоты тактового сигнала, биты 1 и 0.

Таблица 3.11

Состояния битов
и устанавливаемый коэффициент деления частоты

SPR1	**SPR0**	Частота SCK
0	0	f_{CL} / 4
0	1	f_{CL} / 16
1	0	f_{CL} / 64
1	1	f_{CL} / 128

Эти два бита управляют частотой тактового сигнала прибора, работающего в ведущем режиме. В ведомом режиме состояния битов влияния
не оказывают. Состояния битов и устанавливаемый коэффициент деления частоты показаны в табл. 3.11.

3.7.3. Регистр статуса SPI – SPCR

На рис. 3.13 изображен регистр статуса SPI.

$0E(2E)
биты

	7	6	5	4	3	2	1	0
	SPIF	WCOL	-	-	-	-	-	-
Доступ	Чт	Чт	Чт	Чт	Чт	Чт	Чт	Чт
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.13. Регистр статуса SPI

Бит 7 – SPIF: – флаг прерывания по SPI.

По завершении обмена последовательными данными бит SPIF устанавливается в состояние 1 и, если бит SPIE в регистре SPCR установлен и разрешено глобальное прерывание, генерируется сигнал прерывания. Если SS вход управляется низким уровнем, когда SPI находится в режиме ведущего, это установит флаг SPIF. Бит SPIF очищается аппаратно при выполнении подпрограммы обработки соответствующего вектора прерывания. Бит SPIF может быть очищен также при первом считывании состояния регистра статуса SPI с установленным битом SPIF, с последующим обращением к регистру данных SPI (SPDR).

Бит 6 – WCOL: – флаг ошибки при записи.

Бит WCOL устанавливается в состояние 1, если в процессе передачи данных выполнялась запись в регистр данных (SPDR). Чтение содержимого регистра данных, как и запись в него, выполненные во время пересылки данных, могут привести к неверному результату. Бит WCOL (и бит SPIF) аппаратно очищаются (сбрасываются в состояние 0) при первом считывании регистра статуса SPI с установленным WCOL, с последующим обращением к регистру данных SPI (SPDR).

Биты 5…0 – Res: – зарезервированные биты.

Эти биты в микроконтроллерах AT90S4434/8535 зарезервированы.

Интерфейс SPI на AT90S4434/8535 также используется для программной памяти и EEPROM-загрузки или загрузки.

3.7.4. Регистр данных SPI – SPDR

На рис. 3.14 изображен регистр данных SPI.

$0F(2F)

биты	7	6	5	4	3	2	1	0
	MSB	-	-	-	-	-	-	*LSB*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.14. Регистр данных SPI

Регистр данных SPI представляет собой регистр с возможностью чтения/записи и предназначен для пересылки данных между регистровым файлом и сдвиговым регистром SPI. Запись в регистр SPDR инициирует передачу данных, считывание регистра приводит к чтению сдвигового регистра приема.

3.8. Универсальный асинхронный приемопередатчик (UART)

Микроконтроллеры AT90S4434/8535 оснащены полнодуплексными универсальными приемопередатчиками (UART). Их основные возможности следующие:

генератор, обеспечивающий множество скоростей передачи информации в бодах;

высокая скорость передачи при низкой частоте XTAL;

8-разрядный или 9-разрядный форматы данных;

фильтрация шума;

обнаружение переполнения;

обнаружение ошибок формирования кадров;

детектирование бита ложного старта;

три отдельных прерывания: по завершении передачи (TX Complete), по пустому регистру передаваемых данных (TX Data Register Empty) и по завершении приема (RX Complete);

буферизованная передача и прием.

Передача данных инициируется записью передаваемых данных в регистр данных I/O UART (UDR). Данные пересылаются из UDR в сдвиговый регистр передачи в следующих случаях:

- Новый символ записан в UDR после того, как был выведен из регистра стоповый бит предшествовавшего символа. Сдвиговый регистр загружается немедленно.

- Новый символ записан в UDR прежде, чем был выведен стоповый бит предшествовавшего символа. Сдвиговый регистр загружается после выхода стопового бита передаваемого символа, находившегося в сдвиговом регистре.

Если из 10(11)-разрядного сдвигового регистра передачи выведена вся информация (сдвиговый регистр передачи пуст), данные из UDR пересылаются в сдвиговый регистр. В это время устанавливается бит UDRE (UART Data Register Empty) регистра статуса UART (USR). При установленном в состояние 1 бите UDRE последний готов принять следующий символ. В то самое время, когда данные пересылаются из UDR в 10(11)-разрядный сдвиговый регистр, бит 0 сдвигового регистра сбрасывается в состояние 0 (состояние 0 – стартовый бит), а бит 9 или 10 устанавливается в состояние 1 (состояние 1 – стоповый бит). Если в регистре управления UART (UCR) установлен бит CHR9 (т. е. выбран режим 9-разрядного слова данных), то бит TXB8 регистра UCR пересылается в бит 9 сдвигового регистра передачи.

Сразу после пересылки данных в сдвиговый регистр тактом бодгенератора стартовый бит сдвигается на вывод TXD. За ним следует LSB-бит данных. Когда будет выдан стоповый бит, сдвиговый регистр загружается новой порцией данных, если она была записана в UDR во время передачи. В процессе загрузки бит UDRE находится в установленном состоянии. Если же новые данные не будут загружены в UDR до выдачи стопового бита, флаг UDRE остается установленным. В этом случае, после того как стоповый бит будет присутствовать на выводе TXD в течение одного такта, в регистре статуса UART (USR) устанавливается флаг завершения передачи TXC (TX Complete Flag).

Установленный в состояние 1 бит TXEN регистра UCR разрешает передачу UART. При очищенном бите TXEN (сброшенном в состояние 0) вывод PD1 может быть использован в качестве вывода I/O общего назначения. При установленном бите TXEN передатчик UART подключается к PD1 и использует его в качестве выхода, независимо от установки бита 1 в DDRD.

Прием данных

Логика приема данных производит выборку состояний вывода RXD с частотой, в 16 раз большей, чем частота бодов. При нахождении линии в пассивном состоянии одиночная выборка нулевого логического уровня будет интерпретироваться как падающий фронт стартового бита и будет запущена последовательность детектирования стартового бита. Считается, что первая выборка обнаружила первый нулевой логический уровень вероятного стартового бита. На выборках 8, 9 и 10 приемник вновь тестирует вывод RXD на изменение логических состояний. Если две или более из этих трех выборок обнаружат логические 1, то данный вероятный стартовый бит отвергается как шумовой всплеск и приемник начнет выявлять и анализировать следующие переходы из 1 в 0. Если же был обнаружен действительный стартовый бит, то начинает производиться выборка следующих за стартовым битом

информационных битов. Эти биты также тестируются на выборках 8, 9 и 10. Логическое состояние бита принимается по двум и более (из трех) одинаковым состояниям выборок. Все биты вводятся в сдвиговый регистр приемника с тем значением, которое было определено тестированием выборок.

При поступлении стопового бита необходимо, чтобы не менее двух выборок из трех подтвердили прием стопового бита (показали высокий уровень). Если же две или более выборок покажут состояния 0, то в регистре статуса UART (USR) устанавливается бит ошибки кадра FE (Framing Error). Для обнаружения ошибки кадра перед чтением регистра UDR необходимо проверять состояние бита FE.

Вне зависимости от того, принят правильный стоповый бит или нет, данные пересылаются в регистр UDR и устанавливается флаг RXC в регистре статуса UART (USR). Регистр UDR фактически является двумя физически отдельными регистрами, один из которых служит для передачи данных, а другой – для приема. При считывании UDR обращение ведется к регистру приема данных, при записи – к регистру передачи. Если выбран режим обмена 9-разрядными словами данных (установлен бит CHR9 регистра UCR), при пересылке данных в UDR бит RXB8 регистра UCR загружается в бит 9 сдвигового регистра передачи.

Если после получения символа к регистру UDR не было обращения, начиная с последнего приема, в регистре UCR устанавливается флаг переполнения (OR). Это означает, что последний байт данных, пересылаемый в сдвиговый регистр, не может быть передан в UDR и потерян. Бит OR буферирован и доступен тогда, когда в UDR читается байт достоверных данных. Для обнаружения переполнения необходимо всегда проверять флаг OR после считывания содержимого регистра UDR.

При очищенном (сброшенном в логическое состояние 0) бите RXEN регистра UCR приемник запрещен. Это означает, что вывод PD0 может использоваться в качестве вывода I/O общего назначения. При установленном бите RXEN приемник UART подключается к выводу PD0, который работает как вывод входа, вне зависимости от установки бита 0 в DDRD.

При установке UART вывода PD0 на работу в качестве входа бит PORTD0 может использоваться для управления нагрузочным резистором вывода.

Когда бит CHR9 в регистре UCR установлен, принимаемые и передаваемые символы – 9-разрядные, плюс стартовый и стоповый биты. Девятый бит данных, который должен передаваться, – бит TXB8 в регистре UCR. Этот бит должен быть установлен в требуемое значение перед началом передачи, которая произойдет по записи в регистр UDR. Девятый бит принимается RXB8 в регистре UCR.

3.8.1. Управление UART

Регистр данных UART – UDR

На рис. 3.15 изображен регистр управления UART.

$0C(2C)

биты

	7	6	5	4	3	2	1	0
	MSB	-	-	-	-	-	-	*LSB*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.15. Регистр управления UART

В действительности регистр UDR является двумя физически разделенными регистрами – регистром передачи данных и регистром приема данных, использующими одни и те же адреса ввода/вывода. При записи в регистр запись производится в регистр передачи данных UART, при чтении происходит чтение содержимого регистра приема данных UART.

Регистр статуса UART – USR

На рис. 3.16 изображен регистр статуса UART.

$0B(2B)

биты

	7	6	5	4	3	2	1	0
	RXC	TXC	UDRE	FE	OR	-	-	-
Доступ	Чт	Чт/Зп	Чт	Чт	Чт	Чт	Чт	Чт
Н. С.	0	0	1	0	0	0	0	0

Рис. 3.16. Регистр статуса UART

Регистр USR обеспечивает только чтение информации о состоянии UART.

Бит 7 – RXC: – прием завершен.

Данный бит устанавливается в состояние 1 при пересылке принятого символа из сдвигового регистра приема в UDR. Бит устанавливается вне зависимости от отсутствия или наличия ошибок приема кадра. При установленном в UCR бите RXCIE и установленном бите RXC выполняется прерывание по завершении приема UART. Бит RXC очищается при считывании UDR. При приеме данных, инициированном прерыванием, подпрограмма обработки прерывания по завершении приема UART должна считать UDR, чтобы очистить RXC, иначе по окончании подпрограммы обработки прерывания произойдет новое прерывание.

Бит 6 – TXC: – передача завершена.

Данный бит устанавливается в состояние 1, когда весь символ (включая стоповый бит) выведен из сдвигового регистра передачи и в UDR не записа-

ны новые данные. Этот флаг используется при полудуплексном связном интерфейсе, когда оборудование передачи должно установить режим приема и освободить коммуникационную шину сразу после завершения передачи.

При установленном в регистре UCR бите TXCIE установка TXC приведет к выполнению прерывания по завершении передачи UART. Флаг TXC очищается аппаратно при выполнении обработки соответствующего вектора прерывания. Очистить бит TXC можно записью в бит логической 1.

Бит 5 – UDRE: – регистр данных пуст.

Данный бит устанавливается в состояние 1, когда символ, записанный в UDR, пересылается в сдвиговый регистр передачи. Установка этого бита означает, что передатчик готов к получению нового символа для передачи.

Когда бит UDRIE в UCR установлен, до тех пор пока установлен UDRE, выполняется прерывание по завершении передачи UART. Бит UDRE очищается при записи в UDR. При приеме данных, инициированном прерыванием, подпрограмма обработки прерывания по пустому регистру данных UART должна считать UDR, чтобы очистить UDRE, иначе по окончании подпрограммы прерывания произойдет новое прерывание. Во время сброса бит UDRE устанавливается в состояние 1, чтобы показать готовность передатчика.

Бит 4 – FE: – ошибка кадра.

Данный бит устанавливается в состояние 1 при обнаружении условий ошибочного приема кадра, т. е. когда стоповый бит входящего символа в состоянии 0. Бит FE очищается при приеме стопового бита с логическим уровнем 1.

Бит 3 – OR: – переполнение данных.

Бит OR устанавливается в состояние 1 при обнаружении условий переполнения, т. е. когда символ, уже находящийся в регистре UDR, не считан перед пересылкой нового символа из сдвигового регистра приема. Бит OR буферирован, что означает, что он будет оставаться установленным, пока не будут считаны правильные данные из UDR. Бит OR очищается (сбрасывается в 0), когда данные приняты и пересланы в UDR.

Биты 2…0 – Res: – зарезервированные биты.

Эти биты в микроконтроллерах AT90S4434/8535 зарезервированы и при считывании всегда покажут состояние 0.

3.8.3. Регистр управления UART – UCR

На рис. 3.17 изображен регистр управления UART.

$0A(2A)

биты	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	*TXB8*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/З	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.17. Регистр управления UART

Бит 7 – RXCIE: – разрешение прерывания по завершении приема.

При установленном в состояние 1 бите RXCIE и установленном разрешении глобального прерывания установка бита RXC в регистре USR приведет к выполнению прерывания по завершении приема.

Бит 6 – TXCIE: – разрешение прерывания по завершению передачи.

При установленном в состояние 1 бите TXCIE и установленном разрешении глобального прерывания установка бита TXC в регистре USR приведет к выполнению прерывания по завершении передачи.

Бит 5 – UDRIE: – разрешение прерывания по пустому регистру данных.

При установленном в состояние 1 бите UDRIE и установленном разрешении глобального прерывания установка бита UDRE в регистре USR приведет к выполнению прерывания по пустому регистру данных UART.

Бит 4 – RXEN: – разрешение приемника.

Установленный в состояние 1 бит RXEN разрешает приемник UART. Если приемник запрещен, то флаги статуса TXC, DOR и FE установить невозможно. Если эти флаги установлены, то очистка бита RXEN не приведет к очистке этих флагов.

Бит 3 – TXEN: – разрешение передатчика.

Установленный в состояние 1 бит TXEN разрешает передатчик UART. При запрещении передатчика во время передачи символа передатчик не будет заблокирован, прежде чем будут полностью переданы символ в сдвиговом регистре и любой находящийся в UDR следующий символ.

Бит 2 – CHR9: – режим 9-разрядных символов.

При установленном в состояние 1 бите CHR9 передаются и принимаются 9-разрядные символы плюс стартовый и стоповый биты. Девятые биты читаются и записываются с использованием битов RXB8 и TXB8 (соответственно) регистра UCR. Девятый бит данных может использоваться как дополнительный стоповый бит или бит контроля четности.

Бит 1 – RXB8: – прием 8-разрядных данных.

При установленном в состояние 1 бите CHR9 бит RXB8 является девятым битом данных принятого символа.

Бит 0 – TXB8: – передача 8-разрядных данных.

При установленном в состояние 1 бите CHR9 бит TXB8 является девятым битом данных передаваемого символа.

3.8.4. Бод-генератор (Baud Rate Generator)

Бод-генератор представляет собой делитель, генерирующий импульсы передачи с частотой, определяемой выражением

$$BAUD = \frac{f_{CK}}{16(UBRR+1)},$$

где BAUD – частота в бодах;
f_{CK} – частота кварцевого генератора;
UBRR – содержимое регистра UBRR (Baud Rate register – 0–255).

При использовании стандартных кварцевых кристаллов наиболее часто используемые скорости передачи в бодах могут быть получены установками UBRR, представленными в табл. 3.12. При установках UBRR, указанных в таблице, реальные скорости в бодах будут иметь отличия менее 2 % от стандартных скоростей. Однако использовать при ошибке более 1 % не рекомендуется. Высокие значения ошибки уменьшают сопротивление шуму.

3.8.5. Регистр бод-генератора UART – UBRR

На рис. 3.18 изображен регистр бод-генератора UART.

$09(29)

биты	7	6	5	4	3	2	1	0
	MSB	-	-	-	-	-	-	*LSB*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.18. Регистр бод-генератора UART

Таблица 3.12

Установки UBRR
при различных стандартных частотах кварцевых кристаллов

Ско-рость (бод)	1 МГц	Ошиб-ка, %	1,8432 МГц	Ошиб-ка, %	**2 МГц**	Ошиб-ка, %	2,4576 МГц	Ошиб-ка, %
1	2	3	4	5	6	7	8	9
2400	UBRR = 25	0,2	UBRR = 47	0,0	UBRR = 51	0,2	UBRR = 63	0,0
4800	UBRR = 12	0,2	UBRR = 23	0,0	UBRR = 25	0,2	UBRR = 31	0,0
9600	UBRR = 6	7,5	UBRR = 11	0,0	**UBRR = 12**	0,2	UBRR = 15	0,0
14400	UBRR = 3	7,8	UBRR = 7	0,0	UBRR = 8	3,7	UBRR = 10	3,1
19200	UBRR = 2	7,8	UBRR = 5	0,0	UBRR = 6	7,5	UBRR = 7	0,0
28800	UBRR = 1	7,8	UBRR = 3	0,0	UBRR = 3	7,8	UBRR = 4	6,3
38400	UBRR = 1	22,9	UBRR = 2	0,0	UBRR = 2	7,8	UBRR = 3	0,0
57600	UBRR = 0	7,8	UBRR = 1	0,0	UBRR = 1	7,8	UBRR = 2	12,5
76800	UBRR = 0	22,9	UBRR = 1	33,3	UBRR = 1	22,9	UBRR = 1	0,0
115200	UBRR = 0	84,3	UBRR = 0	0,0	UBRR = 0	7,8	UBRR = 0	25,0
2400	UBRR = 84	0,4	UBRR = 95	0,0	UBRR = 103	0,2	UBRR = 119	0,0
4800	UBRR = 42	0,8	UBRR = 47	0,0	UBRR = 51	0,2	UBRR = 59	0,0
9600	UBRR = 20	1,6	UBRR = 23	0,0	UBRR = 25	0,2	UBRR = 29	0,0
14400	UBRR = 13	1,6	UBRR = 15	0,0	UBRR = 16	2,1	UBRR = 19	0,0
19200	UBRR = 10	3,1	UBRR = 11	0,0	UBRR = 12	0,2	UBRR = 14	0,0
28800	UBRR = 6	1,6	UBRR = 7	0,0	UBRR = 8	3,7	UBRR = 9	0,0
38400	UBRR = 4	6,3	UBRR = 5	0,0	UBRR = 6	7,5	UBRR = 7	6,7
57600	UBRR = 3	12,5	UBRR = 3	0,0	UBRR = 3	7,8	UBRR = 4	0,0
76800	UBRR = 2	12,5	UBRR = 2	0,0	UBRR = 2	7,8	UBRR = 3	6,7
115200	UBRR = 1	12,5	UBRR = 1	0,0	UBRR = 1	7,8	UBRR = 2	20,0
2400	UBRR = 191	0,0	UBRR = 207	0,2	UBRR = 239	0,0	UBRR = 287	–
4800	UBRR = 95	0,0	UBRR = 103	0,2	UBRR = 119	0,0	UBRR = 143	0,0
9600	UBRR = 47	0,0	UBRR = 51	0,2	UBRR = 59	0,0	UBRR = 71	0,0
14400	UBRR = 31	0,0	UBRR = 34	0,8	UBRR = 39	0,0	UBRR = 47	0,0
19200	UBRR = 23	0,0	UBRR = 25	0,2	UBRR = 29	0,0	UBRR = 35	0,0
28800	UBRR = 15	0,0	UBRR = 16	2,1	UBRR = 19	0,0	UBRR = 23	0,0
38400	UBRR = 11	0,0	UBRR = 12	0,2	UBRR = 14	0,0	UBRR = 17	0,0
57600	UBRR = 7	0,0	UBRR = 8	3,7	UBRR = 9	0,0	UBRR = 11	0,0
76800	UBRR = 5	0,0	UBRR = 6	7,5	UBRR = 7	6,7	UBRR = 8	0,0
115200	UBRR = 3	0,0	UBRR = 3	7, 8	UBRR = 4	0,0	UBRR = 5	0,0

Регистр UBRR является 8-разрядным регистром с возможностью чтения/записи, определяющим скорость UART в соответствии с установками табл. 3.12.

3.9. Аналоговый компаратор (AC)

Аналоговый компаратор сравнивает уровни на положительном выводе PB2 (AIN0) и отрицательном выводе PB3 (AIN1). При напряжении на положительном выводе PB2 (AIN0), большем, чем напряжение на отрицательном выводе PB3 (AIN1), выход аналогового компаратора ACO устанавливается в состояние 1. Выход компаратора может быть использован для управления входом захвата таймера/счетчика 1. Кроме того, компаратор может формировать свой запрос прерывания. Можно задать формирование запроса на прерывание по нарастающему или падающему фронту или по переключению.

Регистр статуса и управления аналогового компаратора – ACSR

На рис. 3.19 изображен регистр статуса и управления аналогового компаратора.

$08(28)

биты	7	6	5	4	3	2	1	0
	ACD	-	ACO	ACI	ACIE	ACIC	ACIS1	*ACIS0*
Доступ	Чт/Зп	Чт	Чт	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	N/A	0	0	0	0	0

Рис. 3.19. Регистр статуса и управления аналогового компаратора

Бит 7 – ACD: – запрет аналогового компаратора.

При установленном в состояние 1 бите ACD аналоговый компаратор запрещен. Для выключения аналогового компаратора установку данного бита можно производить в любое время. При изменении состояния бита ACD необходимо запрещать прерывание по аналоговому компаратору очисткой бита ACIE в регистре ACSR. В противном случае при изменении состояния бита ACD может произойти прерывание.

Бит 6 – Res: – зарезервированный бит.

Этот бит в микроконтроллерах AT90S4434/8535 зарезервирован и при считывании всегда покажет состояние 0.

Бит 5 – ACO: – выход аналогового компаратора.

Бит ACO связан непосредственно с выходом компаратора.

Бит 4 – ACI: – флаг прерывания по аналоговому компаратору.

Данный бит устанавливается в состояние 1 в случае формирования компаратором прерывания, определяемого ACI1 и ACI0. Подпрограмма обработки прерывания по аналоговому компаратору будет выполняться при установленном бите ACIE и установленном бите глобального прерывания в регистре SREG. Бит ACI очищается аппаратно при выполнении соответствующего вектора обработки прерывания. Бит ACI можно очистить также записью во флаг логической 1.

Бит 3 – ACIE: – разрешение прерывания по аналоговому компаратору.

При установленном бите ACIE и установленном бите глобального прерывания регистра SREG активируется прерывание по аналоговому компаратору. При сброшенном бите ACIE прерывание запрещено.

Бит 2 – ACIC: – разрешение входа захвата аналогового компаратора.

Установленный в состояние 1 бит ACIC разрешает срабатывание функции захвата входа таймера/счетчика 1 по переключению аналогового компаратора. В этом случае выход аналогового компаратора подсоединяется непосредственно к входной цепи логики захвата входа, что обеспечивает использование функций подавления шума и выбора вида срабатывания прерывания по захвату входа таймера/счетчика 1. При очищенном бите ACIC соединения нет. Для запуска прерывания по захвату входа таймера/счетчика 1 бит TICIE1 в регистре масок.

Биты 1,0 – ACIS1, ACIS0: – выбор режима прерывания по аналоговому компаратору.

Эти биты определяют характер события компаратора, при котором запускается прерывание по аналоговому компаратору. Варианты установок показаны в табл. 3.13.

Таблица 3.13

Варианты установок прерывания аналогового компаратора

ACIS1	ACIS0	Режим прерывания
0	0	Прерывание по переключению выхода компаратора
0	1	Зарезервировано
1	0	Прерывание по падающему фронту на выходе компаратора
1	1	Прерывание по нарастающему фронту на выходе компаратора

Примечание. При изменении состояния битов ACIS1/ACIS0 прерывание по аналоговому компаратору должно быть запрещено очисткой бита разрешения прерывания в регистре ACSR. В противном случае при изменении состояния битов может произойти прерывание. Использование команд SBI или CBI над битами, кроме ACI, в этом регистре приведет к записи «1» обратно в ACI, что очистит флаг.

3.10. Аналого-цифровой преобразователь (ADC)

Основные характеристики:

разрешение 10 разрядов;
нелинейность 0.5;
абсолютная точность ±2;
время преобразования 65–260 мс;
до 15 kSPS в максимальном разрешении;

8 мультиплексируемых каналов входа;
режимы циклического и однократного преобразования;
прерывание по завершении преобразования;
устройство подавления шумов Sleep режима.

Микроконтроллеры AT90S4434/8535 оснащены 10-разрядным ADC последовательного приближения. ADC подсоединен к 8-канальному аналоговому мультиплексору, позволяющему использовать любой вывод порта A в качестве входа ADC. ADC содержит усилитель выборки/хранения, удерживающий напряжение входа ADC во время преобразования на неизменном уровне.

Для питания ADC используются два отдельных вывода: AVCC и AGND. Вывод AGND должен быть подсоединен к GND и напряжение AVCC не должно отличаться от напряжения VCC более чем на 0,3 В.

Внешнее напряжение сравнения должно быть приложено к выводу AREF. Напряжение должно быть в диапазоне 2 В – AVCC.

3.10.1. Функционирование аналого-цифрового преобразователя

ADC преобразовывает аналоговое входное напряжение в 10-битовую цифровую величину методом последовательного приближения.

Минимальная величина представляет AGND, максимальная величина – напряжение на AREF-пине минус один LSB.

Аналоговый входной канал выбирается записью в биты MUX в ADMUX, тем самым подключается один из восьми входов аналогового мультиплексора ADC7..0 к входу ADC.

Аналого-цифровой преобразователь может работать в двух режимах: режиме однократного преобразования и режиме циклического преобразования. В режиме однократного преобразования каждое преобразование инициируется пользователем. В режиме циклического преобразования ADC осуществляет выборку и обновление содержимого регистра данных ADC непрерывно. Выбор режима производится битом ADFR регистра ADCSR.

Работа ADC разрешается установкой в состояние 1 бита ADEN в регистре ADCSR. Входные канальные выбора не вступят в работу, пока АДЕН установлен. ADC не поглощает мощность, когда АДЕН очищается, таким образом, рекомендуется выключать ADC прежде, чем ввести режим sleep.

Преобразование начинается с установки в состояние 1 бита начала преобразования ADSC. Этот бит находится в состоянии 1 в течение всего цикла преобразования и сбрасывается по завершении преобразования, аппаратно. Если в процессе выполнения преобразования выполняется смена канала данных, то ADC вначале закончит текущее преобразование и лишь потом выполнит переход к другому каналу.

Поскольку ADC формирует 10-разрядный результат, то по завершении преобразования результирующие данные размещаются в двух регистрах дан-

ных ADCH и ADCL. Для обеспечения соответствия результирующих данных считываемому уровню используется специальная логика защиты. Этот механизм работает следующим образом: при считывании данных первым должен быть считан регистр ADCL. Как только ADCL считан, обращение ADC к регистрам данных блокируется. Таким образом, если после считывания состояния ADCL, но до считывания ADCH, будет завершено следующее преобразование, ни один из регистров не будет обновлен и записанный ранее результат не будет искажен. Обращение ADC к регистрам ADCH и ADCL разрешается по завершении считывания содержимого регистра ADCH.

ADC имеет свое собственное прерывание, которое может быть активировано по завершении преобразования. Когда обращение ADC к регистрам запрещено, в процессе считывания регистров ADCL и ADCH прерывание будет активироваться, даже если результат будет потерян.

В режиме однократного преобразования по завершении процесса оцифровки происходит сброс флага ADSC. Вновь установленный программным способом флаг ADSC вызовет новое однократное преобразование.

В режиме циклического преобразования новое будет стартовать немедленно после завершения предыдущего преобразования, пока флаг ADSC остается установленным. Использование режима циклического преобразования и частоты оцифровки 200 kHz дает минимальное время преобразования с максимальным разрешением 65 мкс.

3.10.2. Регистр выбора мультиплексора ADC – ADMUX

На рис. 3.20 изображен регистр выбора канала мультиплексора.

$07(27)

биты	7	6	5	4	3	2	1	0
	-	-	-	-	-	MUX2	MUX1	*MUX0*
Доступ	Чт	Чт	Чт	Чт	Чт	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.20. Регистр выбора канала мультиплексора

Биты 7…3 – Res: – зарезервированные биты.

Эти биты в микроконтроллерах AT90S4434/8535 зарезервированы и при считывании всегда покажут состояние 0.

Биты 2…0 – MUX2...MUX0: – биты выбора аналогового канала.

Содержимое этих битов определяет, какой из восьми аналоговых каналов (0–7) будет подключен к ADC (табл. 3.14).

Таблица 3.14

Определение входа АЦП

MUX2.0	**Вход**
000	ADC0
001	ADC1
010	ADC2
011	*ADC3*
100	ADC4
101	ADC5
110	ADC6
111	ADC7

3.10.3. Регистр управления и состояния ADC – ADCSR

На рис. 3.21 изображен регистр управления и состояния ADC.

$06(26)

биты	7	6	5	4	3	2	1	0
	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	*ADPS0*
Доступ	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп	Чт/Зп
Н. С.	0	0	0	0	0	0	0	0

Рис. 3.21. Регистр управления и состояния ADC

Бит 7 – ADEN: – разрешение ADC.

Установка данного бита в состояние 1 разрешает ADC. Очистка бита запрещает ADC. Запрещение ADC в процессе преобразования прекращает преобразование.

Бит 6 – ADSC: – запуск преобразования ADC.

В режиме однократного преобразования для запуска каждого цикла преобразования необходимо устанавливать бит ADSC в состояние 1. В циклическом режиме бит ADSC устанавливается в состояние 1 только при запуске первого цикла преобразования. Каждый раз после первой установки бита ADSC, выполненной после разрешения ADC или одновременно с разрешением ADC, будет выполняться пустое преобразование, предшествующее активируемому преобразованию. Это пустое преобразование активирует ADC. ADSC будет сохранять состояние 1 в течение всего цикла преобразования и сбрасывается по завершении преобразования. При выполнении пустого преобразования, предшествующего активируемому, бит ADSC остается установленным до завершения активируемого преобразования. Запись 0 в этот бит эффекта не оказывает.

Бит 5 – ADFR: – установка циклического режима работы ADC.

При установленном в состояние 1 бите ADFR ADC будет работать в циклическом режиме. В этом режиме ADC производит выборки и обраще-

ния к регистрам непрерывно (одно за другим). Очистка бита приводит к прекращению циклического режима.

Бит 4 – ADIF: – флаг прерывания ADC.

Данный бит устанавливается в состояние 1 после завершения преобразования и обновления регистров данных. Прерывание по завершении преобразования ADC выполняется, если в состояние 1 установлены бит ADIE и I – бит регистра SREG. Бит ADIF сбрасывается аппаратно при выполнении подпрограммы обработки соответствующего вектора прерывания. Кроме того, бит ADIF может быть очищен записью во флаг логической 1.

Бит 3 – ADIE: – разрешение прерывания ADC.

При установленных в состояние 1 бите ADIE и I – бите регистра SREG активируется прерывание по завершении преобразования ADC.

Биты 2…0 – ADPS2...ADPS0: – выбор коэффициента предварительного деления.

Таблица 3.15

Выбор коэффициента предварительного деления

ADPS2	ADPS1	ADPS0	Коэффициент деления
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Данные биты определяют коэффициент деления частоты XTAL для получения необходимой тактовой частоты ADC (табл. 3.15).

3.10.4. Сканирование аналоговых каналов

Поскольку смена аналоговых каналов происходит после завершения цикла преобразования в циклическом режиме, смена каналов (сканирование каналов) может происходить без прерывания преобразователя. Обычно для выполнения смены канала выполняется прерывание по завершении преобразования. Однако необходимо принять к сведению следующее соображение: прерывание активируется сразу по готовности результата к считыванию. В циклическом режиме следующее преобразование начинается через один тактовый цикл ADC после активации прерывания. Если содержимое ADMUX будет изменено в течение этого одного тактового цикла, то новые установки будут задействованы при начале нового преобразования. Если же

изменение состояния ADMUX произойдет позднее этого тактового цикла, то при активированном преобразовании будут использоваться предшествовавшие установки.

3.11. AVR®-Ассемблер

Компилятор транслирует исходные коды с языка ассемблера в объектный код. Полученный объектный код можно использовать в симуляторе ATMEL AVR Studio либо в эмуляторе ATMEL AVR In-Circuit Emulator. Компилятор также генерирует код, который может быть непосредственно запрограммирован в микроконтроллеры AVR. Компилятор генерирует код, который не требует линковки. Компилятор работает под Microsoft Windows 3.11, Microsoft Windows95 и Microsoft Windows NT. Кроме этого существует консольная версия для MS-DOS.

Набор инструкций семейства микроконтроллеров AVR описан ниже кратко. Для более полной информации обращайтесь к полному описанию инструкций и документации по конкретному микроконтроллеру.

Компилятор работает с исходными файлами, содержащими инструкции, метки и директивы. Инструкции и директивы, как правило, имеют один или несколько операндов.

Примечание. Строка кода не должна превышать 120 символов.

Любая строка может начинаться с метки, которая является набором символов, заканчивающимся двоеточием. Метки используются для указания места, в которое передаётся управление при переходах, а также для задания имён переменных. Входная строка может иметь одну из четырёх форм:

[метка:] директива [операнды] [Комментарий]
[метка:] инструкция [операнды] [Комментарий]
Комментарий
Пустая строка

Комментарий имеет следующую форму:

; [Текст]

Позиции в квадратных скобках необязательны. Текст после точки с запятой (;) и до конца строки игнорируется компилятором. Метки, инструкции и директивы более детально описываются ниже.

Примеры:

```
label: .EQU var1=100     ; Устанавливает var1 равным 100 (Это директива)
       .EQU var2=200     ; Устанавливает var2 равным 200
test:  rjmp test         ; Бесконечный цикл (Это инструкция)
                         ; Строка с одним только комментарием
```

```
; Ещё одна строка с комментарием
```

Компилятор не требует, чтобы метки, директивы, комментарии или инструкции находились в определённой колонке строки.

Инструкции процессоров AVR

В табл. 3.16, табл. 3.17, табл. 3.18, табл. 3.19 приведен набор команд процессоров AVR, более детальное описание их можно найти в AVR Data Book.

Таблица 3.16

Арифметические и логические инструкции

Мнемоника	Операнды	Описание	Операция	Флаги	Циклы
ADD	Rd,Rr	Суммирование без переноса	Rd = Rd + Rr	Z,C,N,V,H,S	1
ADC	Rd,Rr	Суммирование с переносом	Rd = Rd + Rr + C	Z,C,N,V,H,S	1
SUB	Rd,Rr	Вычитание без переноса	Rd = Rd - Rr	Z,C,N,V,H,S	1
SUBI	Rd,K8	Вычитание константы	Rd = Rd - K8	Z,C,N,V,H,S	1
SBC	Rd,Rr	Вычитание с переносом	Rd = Rd - Rr - C	Z,C,N,V,H,S	1
SBCI	Rd,K8	Вычитание константы с переносом	Rd = Rd - K8 - C	Z,C,N,V,H,S	1
AND	Rd,Rr	Логическое И	Rd = Rd · Rr	Z,N,V,S	1
ANDI	Rd,K8	Логическое И с константой	Rd = Rd · K8	Z,N,V,S	1
OR	Rd,Rr	Логическое ИЛИ	Rd = Rd V Rr	Z,N,V,S	1
ORI	Rd,K8	Логическое ИЛИ с константой	Rd = Rd V K8	Z,N,V,S	1
EOR	Rd,Rr	Логическое исключающее ИЛИ	Rd = Rd EOR Rr	Z,N,V,S	1
COM	Rd	Побитная инверсия	Rd = $FF - Rd	Z,C,N,V,S	1
NEG	Rd	Изменение знака (доп. код)	Rd = $00 - Rd	Z,C,N,V,H,S	1
SBR	Rd,K8	Установить бит (биты) в регистре	Rd = Rd V K8	Z,C,N,V,S	1
CBR	Rd,K8	Сбросить бит (биты) в регистре	Rd = Rd · ($FF - K8)	Z,C,N,V,S	1
INC	Rd	Инкрементировать значение регистра	Rd = Rd + 1	Z,N,V,S	1
DEC	Rd	Декрементировать значение регистра	Rd = Rd -1	Z,N,V,S	1
TST	Rd	Проверка на нуль либо отрицательность	Rd = Rd · Rd	Z,C,N,V,S	1
CLR	Rd	Очистить регистр	Rd = 0	Z,C,N,V,S	1
SER	Rd	Установить регистр	Rd = $FF	None	1
ADIW	Rdl,K6	Сложить константу и слово	Rdh:Rdl = Rdh:Rdl + K6	Z,C,N,V,S	2
SBIW	Rdl,K6	Вычесть константу из слова	Rdh:Rdl = Rdh:Rdl - K 6	Z,C,N,V,S	2
MUL	Rd,Rr	Умножение чисел без знака	R1:R0 = Rd * Rr	Z,C	2
MULS	Rd,Rr	Умножение чисел со знаком	R1:R0 = Rd * Rr	Z,C	2
MULSU	Rd,Rr	Умножение числа со знаком с числом без знака	R1:R0 = Rd * Rr	Z,C	2
FMUL	Rd,Rr	Умножение дробных чисел без знака	R1:R0 = (Rd * Rr) << 1	Z,C	2
FMULS	Rd,Rr	Умножение дробных чисел со знаком	R1:R0 = (Rd *Rr) << 1	Z,C	2
FMUL-SU	Rd,Rr	Умножение дробного числа со знаком с числом без знака	R1:R0 = (Rd * Rr) << 1	Z,C	2

Таблица 3.17

Инструкции ветвления

Мнемоника	Операнды	Описание	Операция	Флаги	Циклы
1	2	3	4	5	6
RJMP	k	Относительный переход	PC = PC + k +1	None	2
IJMP	Нет	Косвенный переход на (Z)	PC = Z	None	2
EIJMP	Нет	Расширенный косвенный переход на (Z)	STACK = PC+1, PC(15:0) = Z, PC(21:16) = EIND	None	2
JMP	k	Переход	PC = k	None	3
RCALL	k	Относительный вызов подпрограммы	STACK = PC+1, PC = PC + k + 1	None	3/4*
ICALL	Нет	Косвенный вызов (Z)	STACK = PC+1, PC = Z	None	3/4*
EICALL	Нет	Расширенный косвенный вызов (Z)	STACK = PC+1, PC(15:0) = Z, PC(21:16) =EIND	None	4*
CALL	k	Вызов подпрограммы	STACK = PC+2, PC = k	None	4/5*
RET	Нет	Возврат из подпрограммы	PC = STACK	None	4/5*
RETI	Нет	Возврат из прерывания	PC = STACK	I	4/5*
CPSE	Rd,Rr	Сравнить, пропустить, если равны	if (Rd ==Rr) PC = PC 2 or 3	None	1/2/3
CP	Rd,Rr	Сравнить	Rd -Rr	Z,C,N,V,H,S	1
CPC	Rd,Rr	Сравнить с переносом	Rd - Rr - C	Z,C,N,V,H,S	1
CPI	Rd,K8	Сравнить с константой	Rd - K	Z,C,N,V,H,S	1
SBRC	Rr,b	Пропустить, если бит в регистре очищен	if(Rr(b)==0) PC = PC + 2 or 3	None	1/2/3
SBRS	Rr,b	Пропустить, если бит в регистре установлен	if(Rr(b)==1) PC = PC + 2 or 3	None	1/2/3
SBIC	P,b	Пропустить, если бит в порту очищен	if(I/O(P,b)==0) PC = PC + 2 or 3	None	1/2/3
SBIS	P,b	Пропустить, если бит в порту установлен	if(I/O(P,b)==1) PC = PC + 2 or 3	None	1/2/3
BRBC	s,k	Перейти, если флаг в SREG очищен	if(SREG(s)==0) PC = PC + k + 1	None	1/2
BRBS	s,k	Перейти, если флаг в SREG установлен	if(SREG(s)==1) PC = PC + k + 1	None	1/2

Окончание табл. 3.17

1	2	3	4	5	6
BREQ	k	Перейти, если равно	if(Z==1) PC = PC + k + 1	None	1/2
BRNE	k	Перейти, если не равно	if(Z==0) PC = PC + k + 1	None	1/2
BRCS	k	Перейти, если перенос установлен	if(C==1) PC = PC + k + 1	None	1/2
BRCC	k	Перейти, если перенос очищен	if(C==0) PC = PC + k + 1	None	1/2
BRSH	k	Перейти, если равно или больше	if(C==0) PC = PC + k + 1	None	1/2
BRLO	k	Перейти, если меньше	if(C==1) PC = PC + k + 1	None	1/2
BRMI	k	Перейти, если минус	if(N==1) PC = PC + k + 1	None	1/2
BRPL	k	Перейти, если плюс	if(N==0) PC = PC + k + 1	None	1/2
BRGE	k	Перейти, если больше или равно (со знаком)	if(S==0) PC = PC + k + 1	None	1/2
BRLT	k	Перейти, если меньше (со знаком)	if(S==1) PC = PC + k + 1	None	1/2
BRHS	k	Перейти, если флаг внутреннего переноса установлен	if(H==1) PC = PC + k + 1	None	1/2
BRHC	k	Перейти, если флаг внутреннего переноса очищен	if(H==0) PC = PC + k + 1	None	1/2
BRTS	k	Перейти, если флаг T установлен	if(T==1) PC = PC + k + 1	None	1/2
BRTC	k	Перейти, если флаг T очищен	if(T==0) PC = PC + k + 1	None	1/2
BRVS	k	Перейти, если флаг переполнения установлен	if(V==1) PC = PC + k + 1	None	1/2
BRVC	k	Перейти, если флаг переполнения очищен	if(V==0) PC = PC + k + 1	None	1/2
BRIE	k	Перейти, если прерывания разрешены	if(I==1) PC = PC + k + 1	None	1/2
BRID	k	Перейти, если прерывания запрещены	if(I==0) PC = PC + k + 1	None	1/2

Примечание. Для операций доступа к данным количество циклов указано при условии доступа к внутренней памяти данных и некорректно при работе с внешним ОЗУ. Для инструкций CALL, ICALL, EICALL, RCALL, RET и RETI необходимо добавить три цикла плюс по два цикла для каждого ожидания в контроллерах с PC меньшим 16 бит (128 KB памяти программ). Для устройств с памятью программ свыше 128 KB добавьте пять циклов плюс по три цикла на каждое ожидание.

Таблица 3.18

Инструкции передачи данных

Мнемо-ника	Операн-ды	Описание	Операция	Флаги	Циклы
1	2	3	4	5	6
MOV	Rd,Rr	Скопировать регистр	Rd = Rr	None	1
MOVW	Rd,Rr	Скопировать пару регистров	Rd+1:Rd = Rr+1:Rr, r,d even	None	1
LDI	Rd,K8	Загрузить константу	Rd = K	None	1
LDS	Rd,k	Прямая загрузка	Rd = (k)	None	2*
LD	Rd,X	Косвенная загрузка	Rd = (X)	None	2*
LD	Rd,X+	Косвенная загрузка с пост-инкрементом	Rd = (X), X=X+1	None	2*
LD	Rd,-X	Косвенная загрузка с пре-декрементом	X=X-1, Rd = (X)	None	2*
LD	Rd,Y	Косвенная загрузка	Rd = (Y)	None	2*
LD	Rd,Y+	Косвенная загрузка с пост-инкрементом	Rd = (Y), Y=Y+1	None	2*
LD	Rd,-Y	Косвенная загрузка с пре-декрементом	Y=Y-1, Rd = (Y)	None	2*
LDD	Rd,Y+q	Косвенная загрузка с замеще-нием	Rd = (Y+q)	None	2*
LD	Rd,Z	Косвенная загрузка	Rd = (Z)	None	2*
LD	Rd,Z+	Косвенная загрузка с пост-инкрементом	Rd = (Z), Z=Z+1	None	2*
LD	Rd,-Z	Косвенная загрузка с пре-декрементом	Z=Z-1, Rd = (Z)	None	2*
LDD	Rd,Z+q	Косвенная загрузка с замеще-нием	Rd = (Z+q)	None	2*
STS	k,Rr	Прямое сохранение	(k) = Rr	None	2*
ST	X,Rr	Косвенное сохранение	(X) = Rr	None	2*
ST	X+,Rr	Косвенное сохранение с пост-инкрементом	(X) = Rr, X=X+1	None	2*
ST	-X,Rr	Косвенное сохранение с пре-декрементом	X=X-1, (X)=Rr	None	2*
ST	Y,Rr	Косвенное сохранение	(Y) = Rr	None	2*

Окончание табл. 3.18

1	2	3	4	5	6
ST	Y+,Rr	Косвенное сохранение с постинкрементом	(Y) = Rr, Y=Y+1	None	2
ST	-Y,Rr	Косвенное сохранение с предекрементом	Y=Y-1, (Y) = Rr	None	2
ST	Y+q,Rr	Косвенное сохранение с замещением	(Y+q) = Rr	None	2
ST	Z,Rr	Косвенное сохранение	(Z) = Rr	None	2
ST	Z+,Rr	Косвенное сохранение с постинкрементом	(Z) = Rr, Z=Z+1	None	2
ST	-Z,Rr	Косвенное сохранение с предекрементом	Z=Z-1, (Z) = Rr	None	2
ST	Z+q,Rr	Косвенное сохранение с замещением	(Z+q) = Rr	None	2
LPM	Нет	Загрузка из программной памяти	R0 = (Z)	None	3
LPM	Rd,Z	Загрузка из программной памяти	Rd = (Z)	None	3
LPM	Rd,Z+	Загрузка из программной памяти с постинкрементом	Rd = (Z), Z=Z+1	None	3
ELPM	Нет	Расширенная загрузка из программной памяти	R0 = (RAMPZ:Z)	None	3
ELPM	Rd,Z	Расширенная загрузка из программной памяти	Rd = (RAMPZ:Z)	None	3
ELPM	Rd,Z+	Расширенная загрузка из программной памяти с постинкрементом	Rd = (RAMPZ:Z), Z = Z+1	None	3
SPM	Нет	Сохранение в программной памяти	(Z) = R1:R0	None	-
ESPM	Нет	Расширенное сохранение в программной памяти	(RAMPZ:Z) = R1:R0	None	-
IN	Rd,P	Чтение порта	Rd = P	None	1
OUT	P,Rr	Запись в порт	P = Rr	None	1
PUSH	Rr	Занесение регистра в стек	STACK = Rr	None	2
POP	Rd	Извлечение регистра из стека	Rd = STACK	None	2

Примечание. Для операций доступа к данным количество циклов указано при условии доступа к внутренней памяти данных и некорректно при работе с внешним ОЗУ. Для инструкций LD, ST, LDD, STD, LDS, STS, PUSH и POP необходимо добавить один цикл плюс по одному циклу для каждого ожидания.

Таблица 3.19

Инструкции работы с битами

Мнемо-ника	Операн-ды	Описание	Операция	Флаги	Циклы
1	2	3	4	5	6
LSL	Rd	Логический сдвиг влево	Rd(n+1)=Rd(n), Rd(0)=0, C=Rd(7)	Z,C,N,V,H,S	1
LSR	Rd	Логический сдвиг вправо	Rd(n)=Rd(n+1), Rd(7)=0, C=Rd(0)	Z,C,N,V,S	1
ROL	Rd	Циклический сдвиг влево через C	Rd(0)=C, Rd(n+1)=Rd(n), C=Rd(7)	Z,C,N,V,H,S	1
ROR	Rd	Циклический сдвиг вправо через C	Rd(7)=C, Rd(n)=Rd(n+1), C=Rd(0)	Z,C,N,V,S	1
ASR	Rd	Арифметический сдвиг вправо	Rd(n)=Rd(n+1), n=0,...,6	Z,C,N,V,S	1
SWAP	Rd	Перестановка тетрад	Rd(3..0) = Rd(7..4), Rd(7..4)= Rd(3..0)	None	1
BSET	s	Установка флага	SREG(s) = 1	SREG(s)	1
BCLR	s	Очистка флага	SREG(s) = 0	SREG(s)	1
SBI	P,b	Установить бит в порту	I/O(P,b) = 1	None	2
CBI	P,b	Очистить бит в порту	I/O(P,b) = 0	None	2
BST	Rr,b	Сохранить бит из регистра в T	T = Rr(b)	T	1
BLD	Rd,b	Загрузить бит из T в регистр	Rd(b) = T	None	1
SEC	Нет	Установить флаг переноса	C =1	C	1
CLC	Нет	Очистить флаг переноса	C = 0	C	1
SEN	Нет	Установить флаг отрицательного числа	N = 1	N	1
CLN	Нет	Очистить флаг отрицательного числа	N = 0	N	1
SEZ	Нет	Установить флаг нуля	Z = 1	Z	1
CLZ	Нет	Очистить флаг нуля	Z = 0	Z	1
SEI	Нет	Установить флаг прерываний	I = 1	I	1
CLI	Нет	Очистить флаг прерываний	I = 0	I	1
SES	Нет	Установить флаг числа со знаком	S = 1	S	1

Окончание табл. 3.19

1	2	3	4	5	6
CLN	Нет	Очистить флаг числа со знаком	S = 0	S	1
SEV	Нет	Установить флаг переполнения	V = 1	V	1
CLV	Нет	Очистить флаг переполнения	V = 0	V	1
SET	Нет	Установить флаг T	T = 1	T	1
CLT	Нет	Очистить флаг T	T = 0	T	1
SEH	Нет	Установить флаг внутреннего переноса	H = 1	H	1
CLH	Нет	Очистить флаг внутреннего переноса	H = 0	H	1
NOP	Нет	Нет операции	Нет	None	1
SLEEP	Нет	Спать (уменьшить энергопотребление)	Смотрите описание инструкции	None	1
WDR	Нет	Сброс сторожевого таймера	Смотрите описание инструкции	None	1

Примечание. Ассемблер не различает регистр символов.

Операнды могут быть следующих видов:
Rd: Результирующий (и исходный) регистр в регистровом файле.
Rr: Исходный регистр в регистровом файле.
b: Константа (3 бита), может быть константное выражение.
s: Константа (3 бита), может быть константное выражение.
P: Константа (5–6 бит), может быть константное выражение.
K6; Константа (6 бит), может быть константное выражение.
K8: Константа (8 бит), может быть константное выражение.
k: Константа (размер зависит от инструкции), может быть константное выражение.
q: Константа (6 бит), может быть константное выражение.
Rdl: R24, R26, R28, R30. Для инструкций ADIW и SBIW.
X,Y,Z: Регистры косвенной адресации (X=R27:R26, Y=R29:R28, Z=R31:R30).

Директивы ассемблера

Компилятор поддерживает ряд директив, которые не транслируются непосредственно в код. Вместо этого они используются для указания положения в программной памяти, определения макросов, инициализации памяти и т. д. Список директив приведен в табл. 3.20.

Таблица 3.20

Директивы AVR ассемблера

Директива	Описание
BYTE	Зарезервировать байты в ОЗУ
CSEG	Программный сегмент
DB	Определить байты во флэш или EEPROM
DEF	Назначить регистру символическое имя
DEVICE	Определить устройство для которого компилируется программа
DSEG	Сегмент данных
DW	Определить слова во флэш или EEPROM
ENDM, ENDMACRO	Конец макроса
EQU	Установить постоянное выражение
ESEG	Сегмент EEPROM
EXIT	Выйти из файла
INCLUDE	Вложить другой файл
LIST	Включить генерацию листинга
LISTMAC	Включить разворачивание макросов в листинге
MACRO	Начало макроса
NOLIST	Выключить генерацию листинга
ORG	Установить положение в сегменте
SET	Установить переменный символический эквивалент выражения

Примечание. Все директивы предваряются точкой.

BYTE – зарезервировать байты в ОЗУ.

Директива BYTE резервирует байты в ОЗУ. Если Вы хотите иметь возможность ссылаться на выделенную область памяти, то директива BYTE должна быть предварена меткой. Директива принимает один обязательный параметр, который указывает количество выделяемых байт. Эта директива может использоваться только в сегменте данных (см. директивы CSEG и DSEG). Выделенные байты не инициализируются.

Синтаксис:
МЕТКА: .BYTE выражение
Пример:

```
        .DSEG
var1:   .BYTE 1           ; резервирует 1 байт для var1
table:  .BYTE tab_size    ; резервирует tab_size байт
        .CSEG
        ldi r30,low(var1) ; Загружает младший байт регистра Z
        ldi r31,high(var1); Загружает старший байт регистра Z
        ld r1,Z           ; Загружает VAR1 в регистр 1
```

CSEG – программный сегмент.

Директива CSEG определяет начало программного сегмента. Исходный файл может состоять из нескольких программных сегментов, которые объединяются в один программный сегмент при компиляции. Программный сегмент является сегментом по умолчанию. Программные сегменты имеют свои собственные счётчики положения, которые считают не побайтно, а пословно. Директива ORG может быть использована для размещения кода и констант в необходимом месте сегмента. Директива CSEG не имеет параметров.

Синтаксис:

CSEG

Пример:

```
  .DSEG      ; Начало сегмента данных
vartab: .BYTE 4   ; Резервирует 4 байта в ОЗУ
        .CSEG      ; Начало кодового сегмента
const: .DW 2       ; Разместить константу 0x0002 в памяти программ
        mov r1,r0  ; Выполнить действия
```

DB – определить байты во флэш или EEPROM.

Директива DB резервирует необходимое количество байт в памяти программ или в EEPROM. Если Вы хотите иметь возможность ссылаться на выделенную область памяти, то директива DB должна быть предварена меткой. Директива DB должна иметь хотя бы один параметр. Данная директива может быть размещена только в сегменте программ (CSEG) или в сегменте EEPROM (ESEG).

Параметры, передаваемые директиве, – это последовательность выражений, разделённых запятыми. Каждое выражение должно быть или числом в диапазоне (-128..255), или в результате вычисления должно давать результат в этом же диапазоне, в противном случае число усекается до байта, причём без выдачи предупреждений.

Если директива получает более одного параметра и текущим является программный сегмент, то параметры упаковываются в слова (первый параметр – младший байт), и если число параметров нечётно, то последнее выражение будет усечено до байта и записано как слово со старшим байтом, равным нулю, даже если далее идет ещё одна директива DB.

Синтаксис:

МЕТКА: .DB список_выражений

Пример:

```
        .CSEG
consts: .DB 0, 255, 0b01010101, -128, 0xaa
        .ESEG
const2: .DB 1,2,3
```

DEF – назначить регистру символическое имя.

Директива DEF позволяет ссылаться на регистр через некоторое сим-

волическое имя. Назначенное имя может использоваться во всей нижеследующей части программы для обращений к данному регистру. Регистр может иметь несколько различных имен. Символическое имя может быть переназначено позднее в программе.

Синтаксис:
.DEF Символическое_имя = Регистр
Пример:

```
.DEF temp=R16
.DEF ior=R0
.CSEG
ldi     temp,0xf0    ; Загрузить 0xf0 в регистр temp (R16)
in      ior,0x3f     ; Прочитать SREG в регистр ior (R0)
eor     temp,ior     ; Регистры temp и ior складываются по исключающему или
```

DEVICE – определить контроллер, для которого компилируется программа.

Директива DEVICE позволяет указать, для какого контроллера компилируется программа. При использовании данной директивы компилятор выдаст предупреждение, если будет найдена инструкция, которую не поддерживает данный микроконтроллер. Также будет выдано предупреждение, если программный сегмент либо сегмент EEPROM превысят размер, допускаемый устройством. Если же директива не используется, то все инструкции считаются допустимыми, и отсутствуют ограничения на размер сегментов.

Синтаксис:
.DEVICE AT90S1200 |AT90S2313 | AT90S2323 | AT90S2333 | AT90S2343 | AT90S4414 | AT90S4433 | AT90S4434 | AT90S8515 | AT90S8534 | AT90S8535 | ATtiny11 | ATtiny12 | ATtiny22 | ATmega603 | ATmega103
Пример:

```
.DEVICE AT90S1200 ; Используется AT90S1200
.CSEG
push r30      ; Эта инструкция вызовет предупреждение
              ; поскольку AT90S1200 её не имеет
```

DSEG – сегмент данных.

Директива DSEG определяет начало сегмента данных. Исходный файл может состоять из нескольких сегментов данных, которые объединяются в один сегмент при компиляции. Сегмент данных обычно состоит только из директив BYTE и меток. Сегменты данных имеют свои собственные побайтные счётчики положения. Директива ORG может быть использована для размещения переменных в необходимом месте ОЗУ. Директива не имеет параметров.

Синтаксис:
.DSEG
Пример:

```
.DSEG               ; Начало сегмента данных
var1:  .BYTE 1      ; зарезервировать 1 байт для var1
table: .BYTE tab_size  ; зарезервировать tab_size байт.
.CSEG
ldi r30,low(var1)   ; Загрузить младший байт регистра Z
ldi r31,high(var1)  ; Загрузить старший байт регистра Z
ld r1,Z             ; Загрузить var1 в регистр r1
```

DW – определить слова во флэш или EEPROM.

Директива DW резервирует необходимое количество слов в памяти программ или в EEPROM. Если Вы хотите иметь возможность ссылаться на выделенную область памяти, то директива DW должна быть предварена меткой. Директива DW должна иметь хотя бы один параметр. Данная директива может быть размещена только в сегменте программ (CSEG) или в сегменте EEPROM (ESEG). Параметры, передаваемые директиве, – это последовательность выражений, разделённых запятыми. Каждое выражение должно быть или числом в диапазоне (-32768..65535), или в результате вычисления должно давать результат в этом же диапазоне, в противном случае число усекается до слова, причем без выдачи предупреждений.

Синтаксис:
МЕТКА: .DW expressionlist
Пример:

```
.CSEG
varlist: .DW 0, 0xffff, 0b1001110001010101, -32768, 65535
.ESEG
eevarlst: .DW 0,0xffff,10
```

ENDMACRO – конец макроса.

Директива определяет конец макроопределения и не принимает никаких параметров. Для информации по определению макросов см. директиву MACRO.

Синтаксис:
.ENDMACRO
Пример:

```
.MACRO SUBI16          ; Начало определения макроса
subi r16,low(@0)  ; Вычесть младший байт первого параметра
sbci r17,high(@0) ; Вычесть старший байт первого параметра
.ENDMACRO
```

EQU – установить постоянное выражение.

Директива EQU присваивает метке значение. Эта метка может позднее

использоваться в выражениях. Метка, которой присвоено значение данной директивой, не может быть переназначена и её значение не может быть изменено.

Синтаксис:
.EQU метка = выражение
Пример:

```
.EQU io_offset = 0x23
.EQU porta = io_offset + 2
.CSEG        ; Начало сегмента данных
clr r2       ; Очистить регистр r2
out porta,r2 ; Записать в порт A
```

ESEG – сегмент EEPROM.

Директива ESEG определяет начало сегмента EEPROM. Исходный файл может состоять из нескольких сегментов EEPROM, которые объединяются в один сегмент при компиляции. Сегмент EEPROM обычно состоит только из директив DB, DW и меток. Сегменты EEPROM имеют свои собственные побайтные счётчики положения. Директива ORG может быть использована для размещения переменных в необходимом месте EEPROM. Директива не имеет параметров.

Синтаксис:
.ESEG
Пример:

```
        .DSEG              ; Начало сегмента данных
var1:  .BYTE 1             ; зарезервировать 1 байт для var1
table: .BYTE tab_size      ; зарезервировать tab_size байт.
.ESEG
eevar1: .DW 0xffff         ; проинициализировать 1 слово в EEPROM
```

EXIT – выйти из файла.

Встретив директиву EXIT, компилятор прекращает компиляцию данного файла. Если директива использована во вложенном файле (см. директиву INCLUDE), то компиляция продолжается со строки, следующей после директивы INCLUDE. Если же файл не является вложенным, то компиляция прекращается.

Синтаксис:
.EXIT
Пример:

```
.EXIT        ; Выйти из данного файла
```

INCLUDE – вложить другой файл.

Встретив директиву INCLUDE, компилятор открывает указанный в ней файл, компилирует его, пока файл не закончится или не встретится директи-

ва EXIT, после этого продолжает компиляцию начального файла со строки, следующей за директивой INCLUDE. Вложенный файл может также содержать директивы INCLUDE.

Синтаксис:
.INCLUDE "имя_файла"
Пример:

```
; файл iodefs.asm:
.EQU sreg = 0x3f          ; Регистр статуса
.EQU sphigh = 0x3e        ; Старший байт указателя стека
.EQU splow = 0x3d         ; Младший байт указателя стека
; файл incdemo.asm
.INCLUDE iodefs.asm       ; Вложить определения портов
in r0,sreg                ; Прочитать регистр статуса
```

LIST – включить генерацию листинга.

Директива LIST указывает компилятору на необходимость создания листинга. Листинг представляет собой комбинацию ассемблерного кода, адресов и кодов операций. По умолчанию генерация листинга включена, однако данная директива используется совместно с директивой NOLIST для получения листингов отдельных частей исходных файлов.

Синтаксис:
.LIST
Пример:

```
.NOLIST                   ; Отключить генерацию листинга
.INCLUDE "macro.inc"      ; Вложенные файлы не будут
.INCLUDE "const.def"      ; отображены в листинге
.LIST                     ; Включить генерацию листинга
```

LISTMAC – включить разворачивание макросов в листинге.

После директивы LISTMAC компилятор будет показывать в листинге содержимое макроса. По умолчанию в листинге показывается только вызов макроса и передаваемые параметры.

Синтаксис:
.LISTMAC

Пример:

```
.MACRO MACX ; Определение макроса
add r0,@0           ; Тело макроса
eor r1,@1
.ENDMACRO     ; Конец макроопределения
.LISTMAC      ; Включить разворачивание макросов
MACX r2,r1    ; Вызов макроса (в листинге будет показано тело макроса)
```

MACRO – начало макроса.

С директивы MACRO начинается определение макроса. В качестве параметра директиве передаётся имя макроса. При встрече имени макроса позднее в тексте программы компилятор заменяет это имя на тело макроса. Макрос может иметь до 10 параметров, к которым в его теле обращаются через @0-@9. При вызове параметры перечисляются через запятые. Определение макроса заканчивается директивой ENDMACRO. По умолчанию в листинг включается только вызов макроса, для разворачивания макроса необходимо использовать директиву LISTMAC. Макрос в листинге показывается знаком +.

Синтаксис:

```
.MACRO макроимя
```

Пример:

```
.MACRO SUBI16                ; Начало макроопределения
subi @1,low(@0)          ; Вычесть младший байт параметра 0 из параметра 1
sbci @2,high(@0)             ; Вычесть старший байт параметра 0 из параметра 2
.ENDMACRO                ; Конец макроопределения
.CSEG                    ; Начало программного сегмента
SUBI16 0x1234,r16,r17  ; Вычесть 0x1234 из r17:r16
```

NOLIST – выключить генерацию листинга.

Директива NOLIST указывает компилятору на необходимость прекращения генерации листинга. Листинг представляет собой комбинацию ассемблерного кода, адресов и кодов операций. По умолчанию генерация листинга включена, однако может быть отключена данной директивой. Кроме того, данная директива может быть использована совместно с директивой LIST для получения листингов отдельных частей исходных файлов.

Синтаксис:

```
.NOLIST
```

Пример:

```
.NOLIST                  ; Отключить генерацию листинга
.INCLUDE "macro.inc"  ; Вложенные файлы не будут
.INCLUDE "const.def"   ; отображены в листинге
.LIST                    ; Включить генерацию листинга
```

ORG – установить положение в сегменте.

Директива ORG устанавливает счётчик положения равным заданной величине, которая передаётся как параметр. Для сегмента данных она устанавливает счётчик положения в SRAM (ОЗУ), для сегмента программ это программный счётчик, а для сегмента EEPROM это положение в EEPROM. Если директиве предшествует метка (в той же строке), то она размещается по

адресу, указанному в параметре директивы. Перед началом компиляции программный счётчик и счётчик EEPROM равны нулю, а счётчик ОЗУ равен 32 (поскольку адреса 0–31 заняты регистрами). Обратите внимание на то, что для ОЗУ и EEPROM используются побайтные счётчики, а для программного сегмента – пословный.

Синтаксис:
.ORG выражение
Пример:

```
            .DSEG       ; Начало сегмента данных
            .ORG 0x37   ; Установить адрес SRAM равным 0x37
variable:   .BYTE 1     ; Зарезервировать байт по адресу 0x37H
            .CSEG
            .ORG 0x10   ; Установить программный счётчик равным 0x10
            mov r0,r1   ; Данная команда будет размещена по адресу 0x10
```

SET – установить переменный символический эквивалент выражения.

Директива SET присваивает имени некоторое значение. Это имя позднее может быть использовано в выражениях. Причем в отличие от директивы EQU значение имени может быть изменено другой директивой SET.

Синтаксис:
.SET имя = выражение
Пример:

```
.SET io_offset = 0x23
.SET porta = io_offset + 2
.CSEG       ; Начало кодового сегмента
clr r2      ; Очистить регистр 2
out porta,r2 ; Записать в порт A
```

Выражения и операнды

Компилятор позволяет использовать в программе выражения, которые могут состоять операндов, операторов и функций. Все выражения являются 32-битными.

Могут быть использованы следующие операнды:
метки, определённые пользователем (дают значение своего положения);
переменные, определённые директивой SET;
константы, определённые директивой EQU;
числа, заданные в формате:
десятичном (принят по умолчанию): 10, 255;
шестнадцатеричном (два варианта записи): 0x0a, $0a, 0xff, $ff;
двоичном: 0b00001010, 0b11111111;
восьмеричном (начинаются с нуля): 010, 077.
PC – текущее значение программного счётчика (Programm Counter).

Операторы

Компилятор поддерживает ряд операторов, которые перечислены в табл. 3.21 (чем выше положение в таблице, тем выше приоритет оператора). Выражения могут заключаться в круглые скобки, такие выражения вычисляются перед выражениями за скобками.

Таблица 3.21

Операторы

Приоритет	Символ	Описание
1	2	3
14	!	Логическое отрицание
14	~	Побитное отрицание
14	-	Минус
13	*	Умножение
13	/	Деление
12	+	Суммирование
12	-	Вычитание
11	<<	Сдвиг влево
11	>>	Сдвиг вправо
10	<	Меньше чем
10	<=	Меньше или равно
10	>	Больше чем
10	>=	Больше или равно
9	==	Равно
9	!=	Не равно
8	&	Побитное И
7	^	Побитное исключающее ИЛИ
6	|	Побитное ИЛИ
5	&&	Логическое И
4	||	Логическое ИЛИ

Функции

Применимы следующие функции:
LOW(выражение) возвращает младший байт выражения;
HIGH(выражение) возвращает второй байт выражения;
BYTE2(выражение) то же, что и функция HIGH;
BYTE3(выражение) возвращает третий байт выражения;
BYTE4(выражение) возвращает четвёртый байт выражения;

LWRD(выражение) возвращает биты 0–15 выражения;
HWRD(выражение) возвращает биты 16–31 выражения;
PAGE(выражение) возвращает биты 16–21 выражения;
EXP2(выражение) возвращает 2 в степени (выражение);
LOG2(выражение) возвращает целую часть log2(выражение).

Пример простейшей программы на ассемблере для процессора i8080:

```
MVI  B, 1Eh        1. Поместить в регистр B число 1Eh
MVI  A, 00h        2. Поместить в регистр A число 00h
INR  B             3. Увеличить содержимое B на 1.
ADD  B             4. Сложить A и B.
DCR  A             5. Уменьшить содержимое A на 1.
```

Примеры программ

```
; Тестовая программа. Использование светодиодов и кнопок
; Подключите PORTB к LEDS и PORTD к SWITCHES. LEDS.
.include "8515def.inc"
.def Temp =r16          ; Регист временного хранения данных
.def Delay =r17         ; Переменная делителя 1
.def Delay2 =r18        ; Переменная делителя 2
;***** Инициализация
RESET:
        ser     Temp
        out     DDRB,Temp           ; Установить PORTB на вывод
;**** Опрос портов
LOOP:   out     PORTB,temp          ; Переустановить LEDS
        sbis    PIND,0x00           ; Если (Port D, pin0 == 0)
        inc     Temp                ; то счетчик инкрементиров.
        sbis    PIND,0x01           ; Если (Port D, pin1 == 0)
        dec     Temp                ; то счетчик деинкрементиров.
        sbis    PIND,0x02           ; Если (Port D, pin2 == 0)
        ror     Temp                ; то вращать в право
        sbis    PIND,0x03           ; Если (Port D, pin3 == 0)
        rol     Temp                ; то вращать в лево
        sbis    PIND,0x04           ; Если (Port D, pin4 == 0)
        com     Temp                ; то маскировать
        sbis    PIND,0x05           ; Если (Port D, pin5 == 0)
        neg     Temp                ; то инвертировать
        sbis    PIND,0x06           ; Если(Port D, pin6 == 0)
        swap    Temp                ; то поменять тетрады
;**** Задержки для восприятия человеческим глазом.
        DLY:
        dec     Delay
        brne    DLY
```

```
        dec     Delay2
        brne    DLY
        rjmp    LOOP                    ; Повтор цикла

; Программа имитирующая работу "Драйвера" стандартной PC клавиатуры
; Клавиатура подключена к порту D
; пин1 клавиатуры (синхронизация) подключен на вход INT0
; пин2 клавиатуры (данные) подключен к пину 4 порта D
; пин3 клавиатуры (сброс) не используется
; пин4 клавиатуры (земля) подключен к STK
; пин5 клавиатуры (+5В) подключен к STK
; в соответсвии с приходом запроса на прерывание
; с входного пина данных считывается информация и формируется
; принятая посылка.
; из посылки выделяется код нажатой клавиши и выводится на светодиоды
; светодиоды подключены к порту B
; вся посылка-32 бита стартовый байт E0
; затем скан код 1-байт и стоп байт F0
; каждый псевдо "байт" в посылке 11-бит (1-старт, 8-код, 1-четность, 1-стоп
; программа не учитывает последний 33-й стоповый бит

.include "8515def.inc"    ;Подключаемый модуль описания переменных
.def    tmp1_r      =r19  ;Оперативный регистр 1
.def    Count_bit   =r20  ;Счетчик принятых бит в байте
.def    Main_count  =r21  ;Основной счетчик принятых бит в посылке
.dseg                     ; сегмент данных RAM
Store_bytes:    .byte   4 ; здесь хранится принятая посылка 32 бита
.cseg                     ; сегмент команд ROM
.org $0000                ; сегмент команд начинается с таблицы ВП
        rjmp START        ; таблица векторов прерываний вектор 0 старт
        rjmp W_int0       ; вектор внешнего прерывания Int0
        reti              ; прочие вектора не используются
        reti
        reti
        reti
        reti
        reti
        reti
        reti
        reti
        reti
        reti
;-----------настройка основных параметров----------------------------------------
START:
        ldi tmp1_r,LOW(RAMEND)      ;Установка стека
```

```
        out spl,tmp1_r
        ldi tmp1_r,HIGH(RAMEND)
        out sph,tmp1_r
        ldi     r25,$ff         ; настройка портов ввода-вывода
        out     DDRB,r25        ; порт B на вывод к нему подключены
                                ; индикаторы
        ldi     r25,0           ; порт D на ввод на него подается прерывание
        out     DDRD,r25        ; и данные
                                ; настройка процессора
        ldi r16,0b00101010      ; режим sleep - включен, int1 и int0 по
                                ; падающему фронту
        out MCUCR,r16
        ldi r16,0b11000000      ;разрешение int0 int1
        out GIMSK,r16
        clr     r20             ; очистить счетчик принятых бит в байте
        clr     r21             ; очистить счетчик принятых бит в посылке
        ldi     r30,low(Store_bytes)    ; установка буфера
        ldi     r31,high(Store_bytes)   ; для приема посылки
        ldi     r17,0b10000000          ;подготовка маски принятых бит
        sei                             ;разрешение всех прерываний
m1:     rjmp    m1                      ;ожидание прерывания от клавиатуры
; ------------обработка прерывания-----------------------------------------
W_int0: cpi     main_count,32   ; посылка принята полностью ?
        brlo    go_1            ; нет продолжаем принимать
        sbis    PIND,0x04
        nop
        nop
; да вывод принятого скан-кода на индикаторы
        ldi     r30,low(Store_bytes)    ; установка буфера
        ldi     r31,high(Store_bytes)
        inc     ZL      ; первый стартовый байт пропускаем
; ------------выделение кода из посылки-------------------------------------
        ld      r1,z    ; загрузить в регистр R1 принятые данные
        lsl     r1      ; необходимо пропустить первые 4 бита данных
        lsl     r1      ; в них хранится 8 бит стартового байта
        lsl     r1      ; стоп и поритет биты стартового байта
        lsl     r1      ; а также старт бит кода нажатой клавиши итого 4
        inc     ZL      ; загрузить данные содержащие вторую тетраду кода
        ld      r2,z
        lsr     r2      ; подготовить 2-ю тетраду к объединению с первой
        lsr     r2
        lsr     r2
        lsr     r2
        or      r1,r2   ; объеденить тетрады т.е. сфомировать код клавиши
                        ; подготовка кода к выводу на индикаторы
```

```
        clr     r2          ; очистим R2 он используется далее
        clc                 ; очистить бит C регистра флагов он понабится
        rol     r1          ; код в регистре r1 хранится в обратном направлении
        ror     r2          ; поскольку из линии поступает сначала мл. бит
        rol     r1          ; а за тем старший
        ror     r2          ; поэтому необходимо "перевернуть" регистр
        rol     r1          ; для правильного вывода на индикаторы
        ror     r2          ; для этого выдвигаем регистр R1 в регистр R2
        rol     r1          ; через бит C регистра флагов
        ror     r2
        rol     r1
        ror     r2
        rol     r1
        ror     r2
        rol     r1
        ror     r2
        rol     r1
        ror     r2
        out     PORTB,r2            ; "правильный" байт выводим на индикаторы
        clr     r21                 ; последним стоповым байтом
        clr     r20                 ; можно пренебречь
        ldi     r30,low(Store_bytes)    ; установка буфера
        ldi     r31,high(Store_bytes)   ; для приема следующей посылки
        reti
go_1:   cpi     r20,8               ; все биты в байте приняты ?
        brlo    go_2                ; нет продолжаем принимать
        st      Z+,r25              ; да сохраняем принятый байт он в R25
        clr     r20                 ; по адресу в Z  и увеличиваем Z
        clr     r25                 ; очистка промежуточных регистров
        ldi     r17,0b10000000      ; новый бит принимаем подготавливаем
                                    ; маску "скользящая" 1 в регистре R17
        reti
; ------------------установка бита в регистре приемнике---------------------------
go_2:   sbis    PIND,0x04           ; если на пине 4  порта D
        rjmp    to0                 ; 0 - то установить 0 в бите приемника
        rjmp    to1                 ; иначе установить 1
to0:    lsr     r17                 ; 0-на входе данных просто предвигаем маску
        inc     r21                 ; увеличить счетчик принятых бит в посылке
        inc     r20                 ; увеличить счетчик принятых бит в байте
        reti                        ; выход из прерывания бит принят
to1:    or      r25,r17  ;установка 1 в соответствующем бите приемника
        lsr     r17      ;сдвинуть маску
        inc     r21      ;увеличить счетчики
        inc     r20
        reti             ; выход из прерывания бит принят
```

Контрольные вопросы к главе 3

1. Приведите основные исторические сведения и структурные составляющие известных групп микроконтроллеров ATMEL.
2. Какова организация ядра AVR-микроконтроллеров?
3. Рассмотрите AVR-микроконтроллер с точки зрения программиста.
4. Перечислите основные регистры микроконтроллеров AVR.
5. Перечислите основные исполнительные модули микроконтроллеров AVR.
6. Каковы принципы функционирования портов? Почему порт называют квазидвунаправленным?
7. Приведите пример использования таймера для определения длины импульса, а также таймера в режиме тахометра.
8. Что такое широтно-импульсная модуляция.
9. Каков принцип функционирования и какова цель применения сторожевого таймера?
10. Приведите примеры синхронной и асинхронной последовательной связи, раскройте принципы и режимы работы последовательного обмена.
11. Приведите примеры использования микроконтроллеров с АЦП и ЦАП для управления промышленным оборудованием.

Разработайте функциональную схему системы и программу на языке ассемблера AVR для управления модулем терморегулятора с использованием термопары и цифрового термореле, управляемого по каналу I2C.

ЗАКЛЮЧЕНИЕ

Современные микропроцессорные системы являются универсальным и исключительно эффективным средством при решении самых различных проблем в области сбора и преобразования информации, автоматического и автоматизированного управления, выработки и преобразования энергии.

Сфера применения микропроцессоров постоянно расширяется. Практически каждая достаточно сложная техническая система оснащается электронными и микропроцессорными устройствами управления. Трудно назвать технологический процесс, управление которым осуществлялось бы без использования электроники и микропроцессорной техники.

Менее четверти века насчитывает история развития микропроцессорной техники, однако за это короткое время произошли поистине гигантские изменения, связанные с совершенствованием архитектуры микропроцессоров, расширением их функциональных возможностей, увеличением разрядности, повышением степени интеграции и быстродействия. Действительно, современные микропроцессоры уже имеют 64-разрядную архитектуру и по производительности приближаются к мощным суперЭВМ. Кроме того, с появлением технологических возможностей размещения на одном кристалле совместно с процессорным ядром памяти таймерных секций, периферийных узлов, средств сопряжения с внешней средой и т. п. в микропроцессорной технике выделился самостоятельный класс больших интегральных схем – однокристальных микроконтроллеров, предназначенных для интеллектуализации оборудования различного назначения. Можно считать, что микроконтроллеры являются наиболее массовыми представителями микропроцессорной техники.

Если рассмотреть идеализированную систему управления некоторым объектом, то мы увидим, что электрические сигналы, содержащие информацию о контролируемых величинах, вырабатываются соответствующими датчиками. Эти сигналы фильтруются, усиливаются и преобразовываются в цифровую величину аналого-цифровыми преобразователями. Затем они обрабатываются микропроцессором или микроконтроллером, который может взаимодействовать с ЭВМ. Формируемые микропроцессором сигналы управления преобразуются в аналоговую форму с помощью цифроаналоговых преобразователей, усиливаются и подаются на силовые электронные устройства, управляющие исполнительными устройствами, непосредственно воздействующими на объект.

Тенденции развития современных информационных технологий приводят к постоянному возрастанию сложности таких систем, создаваемых в различных областях науки и техники. Для современных микропроцессорных систем характерны следующие особенности:

- сложность описания (достаточно большое количество выполняемых функций, процессов, элементов данных и сложные взаимосвязи между ними), требующая тщательного моделирования и анализа данных и процессов;
- наличие совокупности тесно взаимодействующих компонентов (подсистем), имеющих свои локальные задачи и цели функционирования;
- отсутствие прямых аналогов, ограничивающее возможность использования каких-либо типовых проектных решений и прикладных систем;
- разобщенность и разнородность отдельных групп разработчиков по уровню квалификации и сложившимся традициям использования тех или иных инструментальных средств;
- существенная временная протяженность проекта, обусловленная,

с одной стороны, ограниченными возможностями коллектива разработчиков, и с другой – масштабами организации-заказчика и различной степенью готовности отдельных ее подразделений к внедрению информационных систем.

Для успешной реализации проекта объект проектирования должен быть прежде всего адекватно описан и построены полные и непротиворечивые функциональные и информационные модели.

Накопленный к настоящему времени опыт проектирования показывает, что это логически сложная, трудоемкая и длительная по времени работа, требующая высокой квалификации участвующих в ней специалистов.

Тем не менее при разработке и проектировании микропроцессорных систем не следует гнаться за неким ускользающим «мировым уровнем», который на самом деле материализуется в конкретных и простых технических решениях.

Опыт разработчика – это прежде всего готовые к эксплуатации и опробованные отдельные программно-аппаратные модули, из которых разработчик создает новые системы, когда этого требует проект.

Надеемся, что изложенный в данной книге материал будет хорошей основой для проверки Ваших идей.

ГЛОССАРИЙ

Автоматизированная система (АС) – комплекс технических, программных, других средств и персонала, предназначенный для автоматизации различных процессов.

Адрес – 1. Символ или группа символов, которые идентифицируют отдельные части памяти, регистр, устройство ввода/вывода, рабочую станцию вычислительной сети или другие источники данных либо место назначения для их передачи. 2. В вычислительных сетях – последовательность битов, идентифицирующих получателя или отправителя передаваемых данных.

Архитектура (информационной системы) – концепция, определяющая модель, структуру, выполняемые функции и взаимосвязь компонентов сложного объекта.

Ввод данных – процесс передачи данных от внешней памяти или от внешнего оборудования во внутреннюю память.

Внешняя память – память, не имеющая постоянной связи с вычислительной машиной, в которой данные записаны в доступной для ЭВМ форме.

Входной поток информации – последовательность документов и данных, поступающих для ввода в систему.

Входной файл – файл, содержащий вводимые данные.

Вычислительная сеть (сеть ЭВМ) – единый комплекс, включающий территориально рассредоточенную систему ЭВМ и их терминалов, объединенных в единую систему средствами связи, коммутационным оборудованием, программным обеспечением и протоколами для решения информационных, управленческих, вычислительных и/или других задач.

Главная (ведущая, центральная) ЭВМ – 1. В многомашинных вычислительных комплексах – ЭВМ, осуществляющая управление другими ЭВМ, организацию работ в системе (вычислительной сети) и производящая основную обработку информации. 2. В телекоммуникационных вычислительных сетях – ЭВМ, обеспечивающая обслуживание сети, передачу сообщений и выполнение программ, связанных с дополнительными функциями или задачами.

Данные – сведения, полученные путем измерения, наблюдения, логических или арифметических операций, представленные в форме, пригодной для постоянного хранения, обработки и передачи.

Декодирование – операция извлечения полезной информации из массива, перегруженного избыточными разрядами записи, включенными в него с целью помехозащиты при хранении.

Диапазон измерений – область значений измеряемой величины, для которой нормированы допускаемые погрешности средства измерений.

Запрос – входное сообщение в автоматизированную систему, содержащее требование на выдачу информации.

Информационная система – организационно упорядоченная совокупность документов (массивов документов) и информационных технологий, в том числе с использованием средств вычислительной техники и связи, реализующих информационные процессы.

ИК-порт (инфракрасный порт, Infrared port) – последовательный беспроводной порт, в котором при передаче данных используются источник света и фотодатчик. Передача информации происходит на небольшое расстояние и в зоне прямой видимости. Скорость передачи до 115 Кб/с. Применяется в ноутбуках, ПК-блокнотах, цифровых камерах, мобильных телефонах и других устройствах.

Канал связи (передачи данных) – часть сети, связывающая между собой каждую пару ее оконечных терминалов и состоящая из технических средств передачи и приема данных, включая линию связи, а также средств программного обеспечения и протоколов.

Квантование – 1. Операция преобразования данных из непрерывной формы в дискретную. 2. Разбиение данных на подгруппы (классы), например, при цифровой обработке изображений.

Коммуникационная система – система, выполняющая вспомогательные функции, связанные с передачей информации между другими системами.

Конвертирование форматов – преобразование данных из одного формата в другой, воспринимаемый иной системой (как правило, при экспорте или импорте данных).

Локальная вычислительная сеть (ЛВС) – группа ЭВМ и периферийное оборудование, объединенные одним или несколькими автономными (не арендуемыми) высокоскоростными каналами передачи цифровых данных (в том числе проводными, волоконно-оптическими, радио- или ИК-диапазона) в пределах одного или нескольких близлежащих зданий.

Макрос (макрокоманда) – 1. В интерактивных системах – команда, вызывающая выполнение последовательности других команд. 2. Выражение программы, вместо которого подставляется текст, заданный макроопределением (например, команда языка ассемблера, транслируемая в несколько машинных команд).

Массив – упорядоченная структура множества документов или данных одного типа.

Массовое запоминающее устройство – 1. Внешнее запоминающее устройство большой емкости. 2. Система резервного хранения типа библиотеки картриджей с магнитными лентами, которая может содержать очень большие объемы записей данных.

Математическое обеспечение автоматизированной системы – совокупность алгоритмов и программ, необходимых для управления системой и решения с ее помощью задач обработки информации вычислительной техникой.

Модем – устройство модуляции и демодуляции, которое предназначено для преобразования дискретных двоичных сигналов ЭВМ или абонентских пультов в сигналы, посылаемые в канал оборудованием связи, и обратного преобразования.

Мэйнфрейм – большая высокопроизводительная ЭВМ с весьма значительным объемом оперативной и внешней памяти, которая выполняет функции сервера в развитых локальных вычислительных сетях (ЛВС) с большим числом периферийных ЭВМ и терминалов.

Обработка данных – последовательность операций, производимых над данными, например операций объединения, проверки, арифметических операций и т. д.

Оперативная память – память, составляющая неотъемлемую часть вычислительной машины и находящаяся под ее непосредственным управлением.

Открытая система – система, которая взаимодействует с другими системами в соответствии с установленными стандартами.

Передача данных – процесс передачи данных по каналу связи от источника к приемнику.

Периферийные устройства – ПУ (внешние устройства – ВУ) – часть технического обеспечения, конструктивно отделенная от основного блока

компьютера; комплекс устройств для внешней обработки данных, обеспечивающий их подготовку, ввод, хранение, управление, защиту, вывод и передачу на расстояние по каналам связи. К ПУ ввода относятся дигитайзеры, сканеры и т. п.

Персональная ЭВМ (ПЭВМ), персональный компьютер (ПК, PC) – универсальная ЭВМ, предназначенная для индивидуального использования.

Погрешность измерения – отклонение результата измерения от истинного значения измеряемой величины.

Погрешность измерения абсолютная – погрешность измерения, выраженная в единицах измеряемой величины.

Погрешность измерения относительная – отношение абсолютной погрешности измерения к истинному значению измеряемой величины.

Погрешность измерения систематическая – составляющая погрешности измерения, остающаяся постоянной или закономерно изменяющаяся при повторных измерениях одной и той же величины.

Погрешность измерения случайная – составляющая погрешности измерения, изменяющаяся случайным образом при повторных измерениях одной и той же величины.

Подсистема – совокупность элементов, объединенных единым процессом функционирования, которые, взаимодействуя, реализуют определенную операцию, необходимую для достижения цели, поставленной перед системой в целом.

Поле данных (ПД, поле) – 1. Область на носителе информации, выделенная для записи данных, элементов данных. 2. Часть записи или заполняемой формы, имеющая функционально самостоятельное значение и обрабатываемая как отдельный элемент данных.

Полная совместимость – техническая, программная и информационная совместимость двух или более ЭВМ без каких-либо ограничений для их пользователей.

Протокол – совокупность правил, регламентирующих формат и процедуры обмена информацией между двумя или несколькими независимыми устройствами или процессами.

Сервер – 1. В локальных вычислительных сетях – специализированная ЭВМ, управляющая использованием разделяемых между терминалами сети

дорогостоящих ресурсов системы, например: внешней памяти, баз данных, средств связи и т. д. 2. ЭВМ, выполняющая определенные функции обслуживания вычислительной сети.

Техническая совместимость – способность одной ЭВМ работать с узлами или устройствами, входящими в состав другой ЭВМ.

Файл – совокупность связанных записей, рассматриваемых как единое целое.

X.25 – протокол передачи данных, используемый в глобальных корпоративных сетях (типа Интранет) с коммутацией пакетов.

Хост-узел – отдельная ЭВМ или их группа, имеющая прямое сетевое соединение с Интернет и предоставляющая пользователям теледоступ к своим информационным ресурсам, программно-техническим средствам и службам.

Цифрование (оцифровка, дигитализация) – 1. Процесс аналого-цифрового преобразования данных, т. е. перевод аналоговых данных в цифровую форму, доступную для существования в цифровой машинной среде или хранения на машиночитаемых средствах с помощью цифрователей (дигитайзеров) различного типа. 2. В геоинформатике, компьютерной графике и картографии – преобразование аналоговых графических и картографических документов (оригиналов) в форму цифровых записей, соответствующих векторным представлениям пространственных объектов.

Чувствительность измерительного прибора – отношение изменения сигнала на выходе измерительного прибора к вызывающему его изменению измеряемой величины.

Электронная карта (ЭК) – 1. Картографическое изображение, визуализированное на дисплее (видеоэкране) компьютера на основе данных цифровых карт или баз данных ГИС в отличие от компьютерных карт, визуализируемых невидеоэкранными средствами графического вывода. 2. Картографическое произведение в электронной (безбумажной) форме, представляющее собой цифровые данные (в т. ч. цифровые карты или слои данных ГИС), как правило, в записях на диске CD-ROM вместе с программными средствами их визуализации, обычно картографическим визуализатором или картографическим браузером, предназначенное для генерации ЭК.

Architecture – способ, с помощью которого структурируется программная система, компьютерная сеть или вычислительная система. Различают закрытую архитектуру (closed architecture), открытую архитектуру (open architecture) и распределённую архитектуру (distributed architecture).

ATA (Advanced Technology Attachment – усовершенствованный интерфейс) – интерфейс для подключения жестких дисков, применяемый в настольных системах и серверах начального уровня. Альтернативное название этого стандарта – IDE (Integrated Drive Electronics – интегрированная электроника привода). Интерфейс существует со времени появления первых IBM PC AT и постоянно совершенствуется. Сейчас используются диски с интерфейсами Ultra ATA/33, Ultra ATA/66, Ultra ATA/100 и Ultra ATA/133, где числовой индекс – максимальная скорость передачи данных по интерфейсу в мегабайтах в секунду. ATA является параллельным интерфейсом (данные передаются по интерфейсному кабелю одновременно через несколько проводников). Это ограничивает длину кабеля и максимальное число подключаемых устройств. Недостатки ATA устранены в стандарте Serial ATA, который пока не очень широко поддерживается производителями контроллеров и жестких дисков.

AGP (Accelerated Graphic Port) – шина для подключения видеоадаптера, имеет 32-разрядную архитектуру, работает на частоте 66 МГц. Последняя модификация AGP 8x (соответствующая спецификации AGP 30) имеет частоту 533 МГц и обладает пропускной способностью 2,1 Гбайт/с. Сейчас AGP фактически является безальтернативным вариантом подключения видеоадаптеров.

BUS (шина) – внутренняя информационная магистраль компьютера, по которой передаются данные, адреса и управляющие сигналы от одной составной части компьютера к другой. В ПК используется процессорная шина, реализованная в виде трех основных магистралей:

- шина данных, служащая для двусторонней пересылки данных между памятью и микропроцессором;
- шина адреса (адресная шина), задающая, какой именно участок памяти будет адресован;
- шина управления, передающая по устройствам управляющие их работой сигналы.

В современных IBM PC и PC-совместимых компьютерах в большинстве случаев используются такие шинные архитектуры:

- шина промышленного стандарта архитектуры (ISA);
- шина микроканальной архитектуры (MCA);
- шина расширенной промышленной стандартной архитектуры (EISA);
- шина интерфейса периферийных компонентов (PCI).

CPU (Central Processing Unit – центральный процессор) – сверхбольшая интегральная микросхема, управляющая работой всей вычислительной системы. Процессор состоит из арифметико-логического устройства (АЛУ), устройства управления и процессорной памяти (процессорных регистров).

Cache memory (кэш-память) – память, необходимая для хранения данных, запрос которых происходит очень часто. Она играет роль буфера (пункта пересылки) между двумя различными устройствами. Обычно изготавливается на микросхемах статической памяти (не требующей регенерации), значительно более быстродействующей, чем память типа DRAM. Объем и быстродействие кэш-памяти являются одними из параметров, определяющих быстродействие вычислительной системы в целом. В современных микропроцессорах для максимального увеличения быстродействия кэш-памяти последняя изготавливается встроенной в собственно кристалл процессора. Это позволяет ей работать на той же тактовой частоте, что и сам процессор.

CISC (Complex Instruction Set Computer) – процессор со сложным набором команд; CISC-архитектура, CISC-процессор – традиционная архитектура процессоров с широким набором различных машинных команд переменной длины и разным временем исполнения в противоположность RISC-процессорам. Процессоры семейств 80x86 и 680x0 относятся к CISC-процессорам, однако часто внутри самих CISC-процессоров используется RISC-архитектура.

Controller (контроллер) – микросхема, плата или блок, осуществляющие управление внутренним или периферийным устройством и обмен данными между ним и компьютером.

CD-ROM Drive (Compact Disk-Read Only Memory) – устройство для считывания компакт-дисков. Информация с таких дисков считывается лазерным лучом, который изменяет свою интенсивность, отражаясь от металлизированной пластины с записанной на ней информацией. Эта пластина запаяна внутрь тонкого прозрачного пластикового покрытия, предохраняющего ее от механических повреждений. Компакт-диски могут вставляться непосредственно в привод CD-ROM либо использовать специальную защитную кассету – Caddy, похожую на защитный чехол дискеты, которая помогает сохранить компакт-диски от внешних повреждений.

CD-changer – устройство, в которое можно одновременно установить несколько компакт-дисков. Текущая работа осуществляется только с одним диском. Его смена происходит автоматически либо по команде компьютера и занимает около 1–5 с. Такое устройство обычно применяют для работы с библиотеками информации, записанными на компакт-диски.

Computer architecture – архитектура компьютера (вычислительной системы), организационная структура компьютера (вычислительной системы), включающая потоки данных, интерфейсы, аппаратное и программное обеспечение. Термин введён корпорацией IBM при создании семейства совместимых ЭВМ System/360.

DMA (Direct Memory Access) – прямой доступ к памяти (ПДП) – метод высокоскоростной пересылки данных между ОЗУ и периферийным устройством (например, жёстким диском). Осуществляется, минуя процессор. Иногда под DMA подразумевается микросхема. Впервые ПДП был применён в компьютере PDP-1 корпорации DEC.

DMAC DMA Controller – контроллер ПДП, специализированный контроллер для прямого доступа к памяти.

DRAM (Dynamic Random Access Memory, Dynamic RAM) – динамическая память, динамическое ОЗУ, ЗУПВ – тип асинхронной динамической оперативной памяти, состоящей из конденсаторов и транзисторов, имеющей, как правило, время доступа около 60 нс. Широко используется в компьютерах в виде 72-контактных SIMM-модулей. Из-за того, что конденсаторы со временем теряют свой заряд, требует обновления (refresh) через каждые несколько миллисекунд, что обычно делается аппаратно, но возможно и программными средствами (отсюда слово «dynamic» в её названии). Во время этой операции память недоступна для процессора. Динамическое ОЗУ просто в производстве, дешевле и в несколько раз более ёмко, чем статическое ОЗУ такой же степени интеграции, но по сравнению с последним в 2–3 раза медленнее.

DVD (Digital Versatile Disk – цифровой универсальный диск) – самый современный стандарт хранения информации на оптическом (лазерном) диске. Отличается от обычного CD-ROM увеличенной почти в 30 раз емкостью для записи информации (на данный момент составляет до 17 Гбайт).

Enhanced ISA (EISA), Enhanced ISA (EISA) expansion bus (шина расширенной промышленной стандартной архитектуры) – 32-разрядный стандарт шины расширения, разработанный с целью создания конкуренции шине расширения Micro Channel Architecture (MCA) для IBM. Пропускная способность такой магистрали 33 Мбайт/с, но этот стандарт уже практически вытеснен с рынка шиной PCI.

EPROM Erasable (Electrically) Programmable Read-Only Memory – стираемое программируемое постоянное запоминающее устройство (СППЗУ, перепрограммируемая память – тип микросхем перепрограммируемой постоянной памяти с ультрафиолетовым стиранием). Для стирания содержимого микросхемы со стеклянного окошка на её корпусе удаляется защитная наклейка, и оно заданное время (до 20 мин) освещается ультрафиолетовым светом. Стирать одну и ту же микросхему можно многократно. Для записи данных в микросхему используется программатор. При соблюдении условий эксплуатации запись в микросхеме может храниться несколько лет.

Eeprom Electrically Erasable Programmable Read-Only Memory – электрически стираемое программируемое постоянное запоминающее устройство (ЭСППЗУ, электрически стираемая память – редко используемый тип полупроводникового ППЗУ, в которой для стирания используются электрические сигналы, подаваемые либо из компьютера, либо с внешнего устройства (программатора) на специальные ножки микросхемы). Допускает от 10 до 100 тыс. циклов перезаписи. ЭСППЗУ было вытеснено флэш-памятью.

Flash memory (флэш-память) – может быть записана и прочитана так же, как и динамическое ОЗУ, но сохраняет свое содержимое без питания и регенерации как EPROM.

FDD (Floppy Disk Drive – накопитель на гибких магнитных дисках) – устройство, предназначенное для чтения и записи информации на гибкие магнитные диски. В настоящее время для ПК применяются диски размером 3,5", на которых размещается до 1,44 Мбайт информации. Конструктивно выполняются в тонком жестком пластиковом корпусе, в котором предусмотрена перемещающаяся металлическая шторка, закрытая в обычном положении и автоматически открывающаяся для доступа магнитных головок к поверхности магнитного слоя дискеты, когда дискета вставляется внутрь дисковода.

FireWire, IEEE 1394 – последовательный высокоскоростной порт для высокоскоростной передачи данных, разработанный фирмой Apple. Аналогично USB FireWire способен обеспечивать энергопитанием подключаемые устройства. На одну сигнальную линию можно подсоединить до 63 устройств. Благодаря высокой скорости передачи данных (до 400 Мбайт/с) шина FireWire в основном используется для передачи цифрового видео в профессиональных и бытовых видео- и телекамерах. Существует также модифицированный гигабайтный вариант IEEE 1394.2, в котором применяется оптоволоконный кабель.

FTP (File Transfer Protocol) – 1. Протокол и стандартная программа, предназначенные для обеспечения передачи и приема файлов между разными компьютерами (сервером и клиентом), работающими в сетях, поддерживающих протокол TCP/IP. 2. Сервисное средство Интернет, обеспечивающее доступ к файлам в файловых массивах.

HTTP, http (HyperText Transfer Protocol) – протокол передачи гипертекста, по которому взаимодействуют клиенты с WWW-серверами.

Harvard architecture (гарвардская архитектура) – архитектура процессора, использующая для повышения производительности раздельные адресные шины для кода и данных (они могут быть считаны одновременно за один машинный такт, что уменьшает число тактов, требуемых для выполнения машинной команды), чем отличается от архитектуры фон Неймана. Недостаток – необходимость большего числа ножек (выводов) у микропроцессора,

поэтому используется главным образом в микроконтроллерах, где один из типов памяти – внутренний.

HDD (Hard Disk Drive – накопитель на жестком диске) – устройство, предназначенное для долговременного хранения информации. В отличие от дискет в жестких дисках используется пакет жестких дисков, помещенных в герметически закрытый корпус, внутри которого располагаются вращающий их двигатель, группа головок записи/чтения и плата контроллера с гнездами для подключения интерфейсных и питающих кабелей. Стандартными интерфейсами жестких дисков считаются ESDI, IDE, SCSI и ряд др. В последнее время наибольшей популярностью пользуются два последних интерфейса.

Interrupt (прерывание) – механизм, позволяющий процессору реагировать на события внешнего мира или особые программные состояния. Можно сказать, что прерывание – асинхронное внешнее событие, требующее обслуживания. Существуют различные классы прерываний: аппаратные, программные, ввода/вывода и от таймера. Различают прерывания маскируемые и немаскируемые, в зависимости от того, может ли быть отложено обслуживание конкретного вида прерываний. Например, нельзя маскировать прерывания по исчезновению питания.

Interrupt controller (контроллер прерываний) – микросхема (например, I8259A, I82489), используемая для управления аппаратными прерываниями (их разрешение/запрещение, маскирование, установка приоритета и т. д.).

ISA (Industry Standard Architecture – шина промышленного стандарта архитектуры) – устаревший вариант шинной архитектуры для ПК типа PC AT с разрядностью 16/24 (16 Мбайт), тактовая частота – 8 МГц при предельной пропускной способности 5,55 Мбайт/с.

Magneto-optical drive (магнитооптический диск) – устройство для хранения данных, в основе которого лежит лазерная и магнитная технология. Поверхностный слой магнитооптического диска имеет способность изменять свои свойства под воздействием магнитного поля головки чтения/записи. Однако для этого его поверхность должна быть предварительно разогрета до высокой температуры, что осуществляется с помощью встроенного в устройство лазера.

Memory Management Unit (MMU) – блок управления памятью, один из блоков современного процессора, обеспечивающий работу с виртуальной памятью. Осуществляет, в частности, трансляцию виртуальных адресов в физические. Все запросы к данным посылаются в MMU, где определяется, находятся они в ОЗУ или необходима «подкачка» страницы с диска. Если данные не в ОЗУ, то генерируется аппаратное прерывание.

Microcontroller (микроконтроллер) – однокристальный микропроцессор, разработанный специально для систем управления технологическими процессами, периферийными, коммуникационными, бытовыми приборами и другими устройствами. Обычно имеет небольшой объём ОЗУ, ППЗУ/ПЗУ и порты ввода/вывода.

MCA (Micro Channel Architecture – шина микроканальной архитектуры) – спецификация 32-разрядной шины расширения, разработанная IBM для компьютеров семейства PS/2. Несовместима ни с одной другой и имеет разрядность 32/32 (базовая – 8/24, остальные – в качестве расширений). Поддерживает функцию Bus Mastering, имеет арбитраж и автоматическую конфигурацию, синхронная (жестко фиксирована длительность цикла обмена), предельная пропускная способность 40 Мбайт/с. В основной секции конструктива предусмотрены линии для передачи звуковых сигналов. Дополнительно рядом с одним из разъемов может устанавливаться разъем видеорасширения (20 контактов). Многими параметрами шина похожа на EISA, так как появление EISA было обусловлено собственностью IBM на архитектуру MCA.

MPU – микропроцессор (центральный процессор), реализованный в виде одной микросхемы. Первый коммерческий микропроцессор был разработан корпорацией Intel в 1972 г.

Nonvolatile memory (энергонезависимая память) – память, содержимое которой не пропадает после выключения питания. Это может быть ПЗУ, ППЗУ, ЭСППЗУ, флэш-память или ОЗУ, подпитываемое от дополнительной аккумуляторной батарейки.

PCI (Peripheral Component Interconnect) – наиболее популярный современный стандарт шинной архитектуры персональных компьютеров. Шина имеет разрядность 32/32 (расширенный вариант 64/64), тактовую частоту до 33 МГц (PCI 21 – до 66 МГц), пропускную способность до 132 Мбайт/с (264 Мбайт/с для 32/32 на частоте 66 МГц и 528 Мбайт/с для 64/64 на частоте 66 МГц), поддерживает функции Bus Mastering и автоконфигурацию (автоопределение настроек). Количество разъемов шины на одном сегменте ограничено четырьмя. Сегментов может быть несколько, они соединяются друг с другом при помощи так называемых мостов (bridge). В настоящее время разрабатывается новая высокоскоростная шина PCI-X, которая должна заменить все последние версии PCI и AGP.

PC Card (PCMCIA) – 16-битная шина, используемая для подключения дополнительных устройств к ноутбукам и другим портативным электронным устройствам, например ПК-блокнотам, цифровым видеокамерам. Разработанная в 1990 г. шина первоначально называлась PCMCIA – Personal Computer Memory Card International Association, затем (1995 г.) была пере-

именована в PC Card. Позволяет подключать к портативным компьютерам различные компактные устройства, такие как дополнительные платы памяти, модемы, сетевые адаптеры, приводы CD-ROM. Также обеспечивается питание этих устройств. В современных ноутбуках присутствуют, как правило, два PC Card слота. Существует модификация данного стандарта под названием CardBus, которая отличается более широкой пропускной способностью.

PROM (Programmable Read-Only Memory) – программируемое постоянное запоминающее устройство (ППЗУ) – вид памяти, в которую запись может быть произведена только один раз с помощью специального устройства – программатора – пережиганием плавких перемычек импульсами высокого напряжения. Используется в различных электронных устройствах для хранения встроенного программного обеспечения.

RAM (Random Access Memory) – оперативная память, оперативное запоминающее устройство (ОЗУ) – полупроводниковая оперативная память для чтения и записи данных. В обычных компьютерах – место, куда программа загружается для исполнения. В отличие от постоянной памяти (ROM) содержимое ячейки ОЗУ можно изменять любое число раз и обращаться к данным в любой последовательности. Из-за того, что скорость выборки данных не зависит от физического расположения ячейки памяти, такие устройства иногда называют ЗУПВ (запоминающее устройство с произвольной выборкой). Разделяется на динамическую и статическую, энергозависимую и энергонезависимую.

ROM (Read-Only Memory) – постоянное запоминающее устройство (ПЗУ) – вид постоянного ЗУ, содержимое которого однократно записывается в микросхемы (обычно в фабричных условиях) и может только читаться. Синоним – nonerasable storage.

Register file (регистровый файл) – совокупность доступных программисту регистров процессора.

RAID (Redundant Array of Inexpensive Disks) – избыточный массив недорогих дисков. Может состоять из жестких дисков, установленных на сервере, а может быть отдельным внешним устройством. RAID-массивы обеспечивают более высокую скорость передачи данных, повышенную отказоустойчивость. Наиболее часто используются следующие конфигурации:

- RAID-0 – чередующиеся тома или наборы томов;
- RAID-1 – зеркальные тома или наборы томов;
- RAID-5 – чередующиеся наборы томов с контролем четности;
- RAID-10 – зеркальные тома с чередованием.

RISC (Reduced Instruction Set Computing, technology) – вычисления с сокращённым набором команд. RISC-архитектура – архитектура процессоров, построенная на основе сокращённого набора команд. Характеризуется наличием команд фиксированной длины, большого количества регистров, операций типа «регистр – регистр», отсутствием косвенной адресации. Концепция RISC разработана Джоном Коком (John Cocke) из IBM Research, название придумано профессором университета в Беркли Дэвидом Паттерсоном (David Patterson). Одним из недостатков RISC-архитектуры считается фиксированная длина команд, требующая для хранения программы большего объёма памяти.

Streamer (стример) – накопитель для хранения информации на магнитных лентах. Наиболее часто используется как средство резервного копирования данных. Кассеты с пленкой, содержащей информацию, можно хранить очень долго. Они значительно дешевле жесткого диска, поэтому стоимость хранения одного и того же объема информации для лент на порядок ниже, чем у жесткого диска.

SCSI (Small Computer System Interface) – спецификация интерфейса для подключения внутренних и внешних устройств. Первая версия была разработана в 1986 г. Контроллер SCSI позволяет соединить устройства таким образом, что они могут обмениваться информацией, минуя центральный процессор. SCSI-интерфейс позволяет подключить одновременно до 15 устройств к одному контроллеру. Распространение получили следующие стандарты: Fast SCSI, Fast Wide SCSI, Ultra SCSI, Wide Ultra SCSI, Ultra 2 SCSI, Wide Ultra 2 SCSI, Ultra 160 SCSI, Ultra 320 SCSI. Последние два имеют 16- и 32-битную шину и пропускную способность 1280 и 2560 Мбайт/с соответственно.

SDRAM synchronous DRAM – синхронное динамическое ОЗУ, синхронная динамическая память. Отличается от обычной наличием специального логического блока и двухбанковой структурой. Все операции записи/чтения синхронизированы с основным тактовым сигналом. Поставляются в виде 168-контактных DIMM-модулей или 144-контактных модулей SO-DIMM.

SIMM Single In-line Memory Module – модуль памяти с однорядным расположением выводов, модуль SIMM – небольшая печатная плата для ПК и периферийных устройств с монтажом на поверхность микросхем памяти. Бывают 30- и 72-контактные модули.

SRAM Static RAM – статическое ОЗУ – более дорогое и более быстрое (с временем доступа до 5 нс) динамическое ОЗУ.

Superscalar architecture (суперскалярная архитектура) – архитектура процессора с несколькими конвейерами, предусматривающая возможность

одновременного выполнения более одной обычной машинной (скалярной) команды, т. е. эти команды запускаются в процессоре на выполнение одновременно и выполняются независимо друг от друга на разных конвейерах.

System architecture (архитектура системы) – представление системы как совокупности её функциональных компонентов, их организации и взаимосвязей (шин, сигналов, протоколов, интерфейсов и т. д.).

TCP/IP (Transmission Control Protocol/Internet Protocol) – основной протокол, предназначенный для работы в сетях Интернет в режиме коммутации каналов.

USB (Universal Serail Bus) – универсальная последовательная шина, способная подключать до 127 устройств через концентраторы с активной инициализацией подключения, не требующей перезагрузки компьютера. Спецификация USB разработана лидерами компьютерной и телекоммуникационной промышленности – Compaq, DEC, IBM, Intel, Microsoft, NEC и Northern Telecom в 1995 г. USB-кабель содержит четыре провода (две витые пары), питание 5 В и общий провод. Через USB можно также подпитывать маломощные устройства. USB 11 обладает пропускной способностью 12 Мбайт/с. Сейчас внедряется следующая версия USB 20, которая полностью совместима с предыдущей и имеет пропускную способность до 480 Мбайт/с.

ZIP-накопитель – устройство, позволяющее производить чтение/запись на специальные носители типа дискет с твердым основанием. Они используются для обмена данными, быстрого резервного копирования или архивирования. Технология ZIP разработана в компании Iomega.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Архитектура и проектирование вычислительных систем. Распределенные вычислительные системы : сб. ст. – Рига : РПИ, 1990. – С. 14–21.
2. Бродин, В. Б. Микроконтроллеры. Архитектура, программирование, интерфейс / В. Б. Бродин, И. И. Шагурин. – М. : ЭКОМ, 1999.
3. Буданов, А. Н. Средства разработки и отладки программного обеспечения промышленных контроллеров на базе 8/16-разрядных микропроцессоров фирмы Motorola / А. Н. Буданов // Инженерная микроэлектроника. 1999. – № 1. – С. 32–34.
4. Долинский, М. С. Внутрисхемные эмуляторы микропроцессоров и микроконтроллеров / М. С. Долинский, И. М. Зисельман, А. О. Федорцов // Автоматика и вычислительная техника. – 1999. – № 1. – С. 62–66.
5. Каган, Б. М. Основы проектирования микропроцессорных устройств автоматики / Б. М. Каган, В. В. Сташин. – М. : Энергоатомиздат, 1987.
6. Казаченко, В. Ф. Микроконтроллеры : руководство по применению 16-разрядных микроконтроллеров INTEL MCS-196/296 во встроенных системах управления / В. Ф. Казаченко. – М. : Изд-во «Эком», 1997. – 200 с.
7. Конов, А. А. Современные видеопроцессоры / А. А. Конов. – М. : ДОДЭКА, 2000.
8. Корнеев, В. В. Современные микропроцессоры / В. В. Корнеев, А. И. Киселев. – М. : НОЛИДЖ, 1998.
9. Куприянов, М. С. Цифровая обработка сигналов. Процессоры, алгоритмы, средства проектирования / М. С. Куприянов, Б. Д. Матюшкин. – СПб. : Политехника, 1998.
10. Лачин, В. И. Электроника : учеб. пособие / В. И. Лачин, Н. С. Савелов. – Ростов н/Д. : Феникс, 2001.
11. Лобанов, В. И. Инженерные методы разработки цифровых устройств / В. И. Лобанов. – М. : НИИРТА, 1977.
12. Микропроцессорные системы и микроЭВМ в измерительной технике : учеб. пособие для вузов / А. Г. Филиппов, А. М. Аужбикович, В. М. Немчинов и др. – М. : Энергоатомиздат, 1995.
13. Микропроцессоры и микропроцессорные комплекты интегральных схем : справочник : в 2 т. / В-Б. Б. Абрайтис, Н. Н. Аверьянов, А. И. Белоус и др. ; под ред. В. А. Шахнова. – М. : Радио и связь, 1988.
14. Науман, Г. Стандартные интерфейсы для измерительной техники / Г. Науман, В. Майлинг, А. Щербина : пер. с нем. – М. : Мир, 1982. – 200 с.
15. Оппенгейм, А. В. Цифровая обработка сигналов / А. В. Оппенгейм, Р. В. Шафер : пер. с англ. ; под ред. С. Я. Шаца. – М. : Связь, 1979.
16. Потемкин, И. С. Функциональные узлы цифровой автоматики / И. С. Потемкин. – М. : Энергоатомиздат, 1986.
17. Предко, М. Руководство по микроконтроллерам / М. Предко : в 2 т. Т. 1, 2. – М. : Постмаркет, 2001.

18. Рабинер, Л. Теория и применение цифровой обработки сигналов / Л. Рабинер, Б. Гоулд : пер. с англ. ; под ред. Ю. Н. Александрова. – М. : Мир, 1973.

19. Ремизевич, Т. Микроконтроллеры семейства HC05 фирмы Motorola / Т. Ремизевич // Chip News. – 1998. – № 11–12. – С. 22–26.

20. Сташин, В. В. Проектирование цифровых устройств на однокристальных микроконтроллерах / В. В. Сташин, А. В. Урусов, О. Ф. Мологонцева. – М. : Энергоатомиздат, 1990.

21. Соучек, Б. Микропроцессоры и микроЭВМ / Б. Соучек : пер. с англ. ; под ред. А. И. Петренко. – М. : Сов. радио, 1979.

22. Угрюмов, Е. П. Цифровая схемотехника / Е. П. Угрюмов. – СПб. : БХВ, Санкт-Петербург, 2000.

23. Ушкар, М. Н. Микропроцессорные устройства в радиоэлектронной аппаратуре / М. Н. Ушкар ; под ред. Б. Ф. Высоцкого. – М. : Радио и связь, 1988.

24. Фиргунов, В. Э. IBM PC для пользователя / В. Э. Фиргунов. – М. : Финансы и статистика, 1994.

25. Микропроцессорные системы и микроЭВМ в измерительной технике : учеб. пособие для вузов / А. Г. Филиппов, А. М. Аужбикович, В. М. Немчинов и др. – М. : Энергоатомиздат, 1995.

26. Шагурин, И. И. Микропроцессоры и микроконтроллеры фирмы Motorola / И. И. Шагурин. – М. : Радио и связь. 1998.

27. Шевкопляс, Б. В. Микропроцессорные структуры. Инженерные решения. Дополнение первое : справочник / Б. В. Шевкопляс. – М. : Радио и связь, 1993.

28. Вейсов, Е. А. Микропроцессоры и микроконтроллеры в вычислительных системах : учеб. пособие / Е. А. Вейсов, О. В. Непомнящий. – Красноярск : ИПЦ КГТУ, 2003. – 503 с.

29. Каган, Б. М. Основы проектирования микропроцессорых устройств автоматики / Б. М. Каган, В. В. Сташин. – М. : Энергоатомиздат, 1987.

30. Микропроцессорные системы и микроЭВМ в измерительной технике : учеб. пособие для вузов / А. Г. Филиппов, А. М. Аужбикович, В. М. Немчинов и др. – М. : Энергоатомиздат, 1995.

31. Новиков, Ю. В. Разработка устройств сопряжения для персональных компьютеров типа IBM PC : практ. пособие / Ю. В. Новиков, О. А. Калашников, С. Э. Гуляев ; под общ. ред. Ю. В. Новикова. – М. : ЭКОМ, 1997.

32. Предко, М. Руководство по микроконтроллерам : в 2 т. Т. 1, 2. – М. : Постмаркет, 2001. – 416 с.