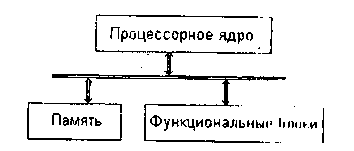
**1. История**

В 1642 Паскаль изобрел механическую счетную машину(+,-). 30 лет спустя Лейбниц сделал машину , которая могла умножать, делить , плюсовать и вычитать. В начале 19в Беббидж сконструировал разностную машину, которая могла складывать и вычитать. Потом сделал аналитическую машину, ее можно было программировать. Программист на этой машине – Ада Августа Ловлейс. В 30 годы 20 в в Гарварде Айкент сделал машину Беббиджа на электрических реле. 1943 – Моушли создал электронный компьютер ENIAC (он имел 20 регистров , 1 регистр – 10-разрядное десятичное число). Джон фон Нейман: «Машина должна состоять: АЛУ, память, устройство ввода-вывода, устройство управления ». 1953 IBM построила первый компьютер – IBM 701. 1956 – изобретен транзистор. 1961 – фирма DEC выпустила комп. PDP-1(120 тыс. долларов). PDP-8 был 12 битным и имел шинную архитектуру(16 тыс. дол.). 1964 – выпуск IBM 360. В начале 80-х изобретена БИС. 1981 – INTEL 8088, на этом проце сделан IBM PC.

**4.** **АРХИТЕКТУРА МИКРОКОНТРОЛЛЕРА**

Контроллером в технике регулирования считается управляющее устройство, осуществляющее регулирующие или контролирующие функции в системе. Контроллер, реализованный на одном кристалле, называется микроконтроллером. Современный микроконтроллер является большой цифровой интегральной схемой, объединяющей миллионы, выполненных по микронным технологиям, транзисторов.



Типовая структура микроконтроллера изображена на рис. 1.1. Микроконтроллер состоит из трех, связанных системными шинами, элементов: процессорного ядра, памяти и набора программируемых функциональных блоков различного назначения.

Рис. 1.1. Структура микроконтроллера

Процессорное ядро (MCU - Microprocessor Com Unit) является основой микроконтроллера. Оно выполняет все вычислительные операции и одновременно, управляет работой всех остальных элементов схемы. По системным шинам процессорное ядро обменивается данными с памятью и всеми функциональными (ннж.ши Разрядность процессорного ядра определяет разрядность микроконтроллера Наиболее распространены в настоящее время 8-битные (8-разрядные) микроконфолперы Вместе с тем, широкое применение в простых задачах находят и 4-битные издания, а в сложных высокопроизводительных системах 16- и 32-битные

В памяти (Memory) хранится программа работы микрокош рол лора, исходные данные и все промежуточные результаты вычислений. Память состоит из множества многоразрядных ячеек, каждая из которых имеет свой адрес По этому адресу процессорное ядро находит конкретную ячейку памяти в процессе обмена. Память микроконтроллера обычно разделена на две части: память данных (Data Memory) и память программ (Program Memory).

Функциональные блоки различных типов обеспечивают взаимодействие микроконтроллера с внешним миром. Эти блоки могут выполнять самые различные функции: ввод и вывод информации, подсчет внешних событий и интервалов времени, передача внешних запросов на процессорное ядро, аналого-цифровые и цифроаналоговые преобразования сигналов, сравнение различных величин, контроль за напряжением питания и др. Для процессорного ядра любой функциональный блок представляется в виде одного или нескольких регистров. Каждый регистр имеет свой оригинальный адрес, по которому процессорное ядро находит его в процессе работы.

Программа работы микроконтроллера хранится в памяти в виде последовательности команд (инструкций). 8 процессе работы процессорное ядро последовательно извлекает из памяти инструкции, расшифровывает и выполняет их. В зависимости от инструкции в ядре выполняются различные арифметические и логические операции, пересыпки данных. При необходимости, в процессе выполнения инструкции, процессорное ядро обращается за данными к ячейкам памяти и функциональным блокам, либо пересыпает в них результаты вычислений. Множество инструкций, которые понимает процессорное ядро, образует систему команд микроконтроллера.

Практически все ведущие производители разрабатывают целые семейства микроконтроллеров с так называемой модульной структурой. При этом процессорное ядро для всего семейства неизменно, а память и состав функциональных блоков у каждого микроконтроллера различны. Процессорное ядро всегда имеет свою оригинальную схему и, обязательно, оригинальное имя. Например, микроконтроллеры фирмы Motorola построены на базе ядра НС05 и НС08, фирма Intel создала ядро MCS-51 и мг.8-96, контроллеры фирмы Microchip строятся на базе ядра PIC12, PIC16, PIC17. 1'К'Ш фирма Л/me/ усиленно развивает семейство микроконтроллеров с ядром AVR

Процессорное ядро на основе известных схемотехнических решений, технологий проектирования и производства цифровых схем реализует определенную архитектуру системы. Для микропроцессорной системы понятие «архитектура» включает в себя множество её структурных особенностей, основными из которых считаются: организация памяти и система команд. В настоящее время известны четыре общих архитектурных принципа в той или другой мере, реализуемые в любом процессорном ядре.

По организации памяти различаются:

° Неймановская архитектура - характеризуется общим пространством памяти для хранения данных и программы. При этом разрядность памяти зафиксирована (как правило, равна одному байту). Такую архитектуру имеют, например, микроконтроллеры НС05 и НС08 фирмы Motorola, в которых общий массив 8-битных ячеек памяти включает в себя как память программ, так и память данных [5].

0 Гарвардская архитектура - отличается разделением памяти программ и памяти данных. При этом разрядность памяти программ и памяти данных, а также шины доступа к ним, различны. В частности, все микроконтроллеры PIC12, PIC16 фирмы Microchip имеют 8-битную память данных, а разрядность памяти программ у них различна: PIC12 имеют 12 битную память программ, а РЮШ - 14 битную [3]. По системе команд различаются:

0 CISC-архитектура (Complicated Instruction Set Computer) - архитектура с развитой системой команд. Система команд процессорного ядра имеет инструкции разного формата: однобайтовые, двухбайтовые, трехбайтовые. Различные инструкции при этом имеют и существенно разное время исполнения.

0 RISC-архитектура (Reduced Instruction Set Computer) - архитектура с сокращенным набором команд. Одна инструкция, как правило, занимает только одну ячейку памяти, и все инструкции имеют равное время исполнения.

Микроконтроллеры с RISC-архитектурой имеют сравнительно более высокую производительность при той же тактовой частоте сигнала синхронизации и в настоящее иремя более распространены.

Разные производители в своих изделиях используют зачастую различные архитектурные принципы. Поэтому приведенное выше деление довольно условно

Например, AVR-микроконтроллеры фирмы Atmel, по мнению ее создателей (Alf Bogen и Vergard Wollan), имеют улучшенную RISC (enhanced RISC) архитектуру. В соответствии с принципами RISC - архитектуры практически все команды микроконтроллера (исключая те, у которых одним из операндов является 16-разрядный адрес) занимают только в одну ячейку памяти программ. Но сделать это разработчикам удалось за счет одновременного использования принципов Гарвардской архитектуры и расширения ячейки памяти программ до 16 разрядов. Поэтому в системе команд AVR-микроконтроллеров целых 130 различных команд, что значительно больше, чем у большинства современных RISC - архитектур. Для сравнения, контроллеры фирмы Microchip с ядром PIC12. PIC16, PIC17 имеют всего 33 команды [3].

**5.** **Архитектура микропроцессора КР580ВМ80А**

Микросхема КР580ВМ80А — функционально законченный однокристальный параллельный 8-разрядный микропроцессор с фиксированной системой команд, применяется в качестве центрального процессора в устройствах обработки данных и управления. Микропроцессор имеет раздельные 16-разрядный канал адреса и 8-разрядный канал данных. Канал адреса обеспечивает прямую адресацию внешней памяти объемом до 65536 байт, 256 устройств ввода и 256 устройств вывода.

Восьмиразрядное арифметико-логическое устройство микропроцессора обеспечивает выполнение арифметических и логических операций над двоичными данными, представленными в дополнительном коде, а также обработку двоично-десятичных упакованных чисел.

В состав блока регистров входят: 16-разрядный регистр адреса команды (IP), 16-разрядный регистр указателя стека (SP), 16-разрядный регистр временного хранения (WZ), 16-разрядная схема инкремента-декремента и шесть 8-разрядных регистров общего назначения (B, С, D, Е, Н, L), которые могут использоваться и как три 16-разрядных регистра (ВС, DE, HL).

Микропроцессор выполняет команды по машинным циклам. Число циклов, необходимое для выполнения команды, зависит от ее типа и может быть от одного до пяти. Машинные циклы выполняются по машинным тактам. Число тактов в цикле определяется кодом выполняемой команды и может быть от трех до пяти. Длительность такта равна периоду тактовой частоты и при частоте 2,0 МГц составляет 500 нс.

**6.** **Определение микроконтроллера**.

Контроллером в технике регулирования считается управляющее устройство, осуществляющее регулирование или контролирующие функции в системе. Контроллер, реализованный на одном кристалле, называется микроконтроллером. Современный микроконтроллер является большой цифровой интегральной схемой, объединяющей миллионы, выполненных по микронным технологиям, транзисторов.

Типовая структура МК изображена на рис. МК состоит из трех, связанных системными шинами, элементов: процессорного ядра, памяти и набора программируемых функциональных блоков различного назначения.

Процессорное ядро является основой МК.

Процессорное ядро

Память

Функциональные блоки

Оно выполняет все вычислительные операции и, одновременно, управляет работой всех остальных элементов схемы.

По системным шинам ПрЯд обменивается данными с памятью и всеми функциональными блоками. Разрядность ПЯ определяет разрядность МК (4,8,16,32-битные).

В памяти хранится программа работы МК, исходные данные и все промежуточные результаты вычислений. Память состоит из множества многоразрядных ячеек, каждая из которых имеет свой адрес. Память МК обычно разделена на две части: память данных и память программ.

Функциональные блоки различных типов обеспечивают взаимодействие МК с внешним миром. Эти блоки могут выполнять самые различные функции: ввод и вывод информации, подсчет внешних событий и интервалов времени, передача внешних запросов на ПЯ, анолого-цифровые и цифроанологовые преобразования сигналов. Для ПЯ любой функциональный блок представляется в виде одного или нескольких регистров. Каждый регистр имеет свой оригинальный адрес, по которому ПЯ находит его процессе работы.

Программа работы МК хранится в памяти в виде последовательности команд. В процессе работы ПЯ последовательно извлекает из памяти инструкции, расшифровывает и выполняет их. В зависимости от инструкций в ядре выполняются различные арифметические и логические операции, пересылки данных. При необходимости, в процессе выполнения инструкций, ПЯ обращается за данными к ячейкам памяти и функциональным блокам, либо пересылает в них результаты вычислений. Множество инструкций, которые понимает ПЯ, образуют систему команд МК.

В настоящее время известны четыре общих архитектурных принципа в той или иной мере, реализуемых в любом ПЯ.

По организации памяти различаются:

- Неймановская архитектура – характеризуется общим пространством памяти для хранения данных и программы. При этом разрядность памяти зафиксирована (как правило 1 байт)

- Гарвардская архитектура – отличается разделением памяти программ и памяти данных. При этом разрядность памяти программ и памяти данных, а также шины доступа к ним, различны.

По системе команд различаются:

- CISC архитектура – архитектура с развитой системой команд. Система команд ПЯ имеет инструкции разного формата: однобайтовые, двухбайтовые, трехбайтовые. Различные имеют разное время исполнения.

- RISC архитектура – архитектура с сокращенным набором команд. Одна инструкция как правило занимает одну ячейку памяти, и все инструкции имеют равное время исполнения.

7. Процессорное ядро.

Каждый производитель МК для серии выпускаемых им изделий разрабатывает и патентует свое оригинальное процессорное ядро. Однако в большинстве из них присутствуют одни и те же элементы: регистр инструкций, программный счетчик, арифметико-логическое устройство, регистры общего назначения, регистр состояния, регистры ввода/вывода, стек.

Программный счетчик: регистр предназначенный для хранения адреса ячейки памяти программ, в которой находится выполняемая в данный момент инструкция. Разрядность программного счетчика определяется количеством ячеек в памяти программы. При выполнении команды содержимое ПС изменяется. В простейшем случае оно увеличивается на 1. Но некоторые команды способны сами записывать данные в ПС. В этом случае новое содержимое ПС и определяется данными, заложенными в выполняемой инструкции.

Регистр инструкций: регистр, предназначенный для хранения считанной из памяти программ инструкции. Считанная из памяти программ инструкция декодируется дешифратором команд и исполняется микропрограммным автоматом ядра. Разрядность регистра инструкций определяется разрядностью памяти программ.

Арифметико-логическое устройство: логическая схема, непосредственно осуществляющая преобразование одной или двух переменных в соответствии с инструкцией занесенной в регистр команд. Стандартное АЛУ способно выполнять простейшие арифметические или логические операции над одной или двумя переменными (сложение, вычитание, инкремент, декремент, NOT, AND, OR, exclusive OR). Некоторые производители интегрируют на кристалл также встроенный умножитель двух переменных.

Регистры общего значения: Регистры общего значения предназначены для временного хранения данных в процессе вычислений. Разрядность регистров определяет разрядность вычислений и разрядность самого МК. Количество регистров может быть произвольным. Обычно в этих регистрах хранится информация, обрабатываемая АЛУ и полученный в нем результат вычислений. На некоторые из регистров могут быть возложены дополнительные функции. В большинстве архитектур один из регистров отличается от других большими возможностями. Он обычно называется аккумулятор или рабочий регистр. В этом регистре может храниться одна из переменных, обрабатываемых в АЛУ, и туда же помещается результат операции.

Регистры ввода/вывода: предназначены для управления функциональными блоками МК, энергозависимой памятью данных и программ. В различных операциях регистры могут участвовать целиком или отдельными битами. Отдельный бит регистра именуется флагом. Обращение к регистрам в различных архитектурах организуется различным образом. Обычно обращение к ним осуществляется как к элементам процессорного ядра по присвоенным в архитектуре именам и адресам, в ряде случаев к ним обращаются с помощью специальных команд ввода/вывода.

Регистр состояния: предназначен для хранения отдельных признаков результата, полученного при выполнении различных арифметических и логических операций в АЛУ. Регистр обычно рассматривается состоящим из отдельных бит (флагов), каждый из которых несет в себе определенную информацию о каком либо одном признаке результата. Типовыми флагами регистра состояния являются:

- флаг переноса: устанавливается при возникновении переноса из старшего разряда результата.

- флаг переполнения: устанавливается при переполнении разрядной сетки.

- флаг отрицательного результата: устанавливается, когда результат является отрицательным числом.

- флаг нулевого результата: устанавливается, когда результат операции равен нулю.

- флаг полупереноса: устанавливается при возникновении переноса из младшей тетрады 8-битного числа в старшую (из третьего разряда в четвертый).

Стек: память данных, организованная по принципу: последний зашел - первый вышел. Такая память предназначается, обычно, для оперативного сохранения содержимого отдельных регистров при переходах к подпрограммам. Одним из таких регистров является программный счетчик. Извлечение из стека содержимого регистров производится в порядке, обратном порядку записи. Запись в стек и извлечение из стека не требует знания адреса ячеек памяти, в которые записываются данные. Стек может быть организован либо в специально созданных в ядре ячейках памяти, либо в области SRAM. В последнем случае в ядре предусматривается специальный регистр – указатель стека. Указатель стека хранит адрес последней записанной ячейки памяти в области стека. Количество ячеек памяти, которые используются в стеке для хранения данных, именуется глубиной стека. Глубина стека в различных архитектурах ядра может быть различна: от 2-х ячеек до размера SRAM.

**классификация команд**

Множество инструкций микроконтроллера образует систему команд. Система команд, как правило, не меняется для всех микроконтроллеров одного семейства. В полном списке инструкций при описании системы обычно выделяются отдельные группы, родственных по применению инструкций, например, арифметические и логические команды, команды пересылки и др. Это деление довольно условно и полностью определяется политикой разработчика.

При описании системы команд приводится полная информация о каждой инструкции: способах адресации, использовании флагов регистра состояния, времени исполнения.

В качестве примера рассмотрим систему команд микроконтроллера А Ттеда 163

Система команд микроконтроллера содержит 130 команд, условно разделенных на четыре группы:

команды пересылки данных (data transfer instructions), 0 арифметические и логические команды (arithmetic and logic instructions), ° команды работы с битами (bit and bit-test instructions), ° команды ветвления (branch instructions).

При описании системы команд использованы следующие обозначения:

Rd - регистр-приемник результата (0 < d < 31),

Rd\* - регистр-приемник результата с номером более 16 (16 < d < 31),

Rr-регистр-источник (0 < г < 31),

Rdl: регистры R24, R26, R28, R30 (для инструкций ADIW и SBIW),

Р- адрес регистра ввода/вывода, Р\*- адрес побитно адресуемого регистра ввода/вывода ($00-$1F)

К - символьная или численная константа (8 бит)

к - адресная константа

b - номер бита в регистре (3 бита)

0 s - номер бита в регистре статуса (3 бита)

0 X, Y, Z - регистры косвенной адресации (X=R27:R26, Y=R29:R28; Z=R31 :R30 Команды пересылки

Команды пересылки осуществляют перемещение данных между ячейками памяти и регистрами процессорного ядра. Один из операндов, участвующих в инструкции, является источником данных, второй - приемником. При пересылке из источника в приемник копия данных всегда остается в источнике. Таким образом, все команды пересылки практически осуществляют копирование данных.

Одним из операндов в любой команде пересылки обычно является регистр общего назначения процессорного ядра. Вторым может быть любой регистр или ячейка памяти.

Инструкции, как правило, не влияют на флаги в регистре состояния процессорного ядра.

В системе команд микроконтроллеров AVR предусмотрено 34 инструкции, осуществляющие пересылку данных. В инструкциях используются все, за исключением битовых, способы адресации данных; одним из операндов в любой инструкции является регистр общего назначения. Инструкция Idi (toad immediate), использующая непосредственную адресацию, может быть использована только в старшей половине файла регистров общего назначения (R16....R31), остальные команды работают с любым из регистров файла. Инструкцию spm (store program memory), осуществляющую запись когда в память программ, можно использовать только в Boot Program Section памяти программ;

Время выполнения инструкций, работающих с регистрами, равно 1 такту. Инструкции, обращающиеся к ячейкам памяти данных, выполняются за 2 такта, а 'вращающиеся к ячейкам памяти программ - за 3 такта.

**Арифметические и логические команды**

В группу арифметических команд входит команды выполняющие сложение, вычитание, декремент и инкремент данных, логическое умножение логическое сложение, операцию ИСКЛЮЧАЮЩЕЕ ИЛИ, инверсию переменной. Обычно к этои группе относят также инструкции сравнения данных. В микроконтроллере Aтmegа 163реализованы также функции арифметического умножения целых чисел и дробных чисел, без знака и со знако3м

Все инструкции этой группы, как правило, приводят к изменению состояния флагов регистра состояния в соответствии с резулыа1ами, выполняемой операции.

В микроконтроллерах с архитектурой AVR предусмотрено довольно много (31) инструкций, выполняющих арифметические и логические преобразования данных. Инструкции используют исключительно прямую регистровую или непосредственную адресацию данных. Операнды хранятся в регистрах общего назначения, в один из них (регистр-приемник) всегда направляется и результат вычислений.

В результате выполнения инструкций изменяются флаги регистра SREG (Status Register), а флаг переноса С (Сапу), кроме того, непосредственно участвует в выполнении ряда операций.

Арифметические команды сложения и вычитания выполняют сложение и вычитание одно- и двухбайтных операндов. Команды adc (add with сапу two registers) и sbc (subtract with carry two registers) используют при вычислениях флаг переноса С.

Инструкции логического умножения (and) и логического сложения (or) , ИСКЛЮЧАЮЩЕЕ ИЛИ (вот) преобразуют только однобайтные данные.

Инструкция дополнения до единицы com (one's complement), фактически выполняет операцию инверсии, а инструкция дополнения до двух neg (two's complement) - меняет знак числа

Инструкции установки (set) позволяют установить как отдельные, так и все биты выбранного регистра в единичное состояние, а команды очистки (dear) - в нулевое.

Инструкции инкремента регистра inc (increment) и декремента регистра dec (decrement) используют прямую адресацию одного выбранного регистра.

Тест на нуль или минус tst (test for zero or minus) фактически не меняет содержимого регистра, но устанавливает соответствующие флаги при равенстве операнда нулю или при его отрицательном значении.

Команды сравнения (compare) также не меняют содержиМ°го РегистР°в' а оценивают разность операндов и устанавливают соответствующие Флаги в регистре состояния.

Шесть инструкций умножения (multiply выполняют умножение |-1ель|Х и Дробных операндов с учетом и без учета знака. 16-битный результат /1^ножения всегда описывается в регистры общего назначения RO:R1.

Большинство арифметических и логических команд выполня!0^ за°Дин такт-Исключение составляют только команды с непосредственной адресац^.' Ра2?таюи^ ° двухбайтными словами: adiw (add immediate to word) и sbiw (subtract im^196 m vvo/")- и команды умножения, выполняющиеся за два такта.

**Битовые команды**

Битовые команды (Вit And Bit-Test Instructions) позволяют обращаться непосредственно к отдельным битам регистров процессорного ядра и выполнять с выбранными битами простейшие операции пересылки, установки и сброса. Операндами команд могут быть как биты регистров общего назначения, так и биты регистров ввода/вывода. Битовые команды могут влиять на отдельные флаги регистра признаков.

Битовые команды микроконтроллером семейства AVR работают с отдельными битами регистров общего назначения и регистров ввоода/вывода Биты регистров ввода/вывода могут быть установлены или сброшены. Биты регистров общего назначения могут быть сдвинуты в соседние ячейки, как влево (в сторону старших разрядов), так и вправо (в сторону младших разрядов). В операциях сдвига, кроме разрядов регистров, может участвовать и бит переноса. Специальные команды предусмотрены дли установки и сброса отдельных флагов регистра состояния.

Команда swap меняет местами тетрады (полубайты) регистров общего назначения.

**10. Микроконтроллер Atmega и его внешние подключения.**

Контроллером в технике регулирования считается управляющее устройство, осуществляющее регулирующие и контролирующие функции в системе. Как правило это вычилительная машина. С развитием технологий различные компоненты этой машины, такие как процессорное ядро, различные виды памяти, АЦП, таймеры и др. становились миниатюрнее и стало возможным включать их в одну интегральную микросхему или их малый набор. Микроконтроллер – контр., реализованный на одном кристалле.

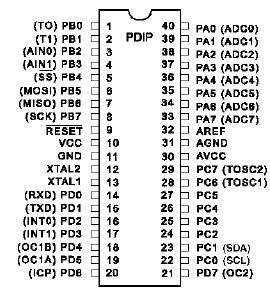
AVR микроконтроллеры фирмы Atmel имеют улучшенную RISС архитектуру.

Микроконтроллер состоит из трех элементов: процессорного ядра, памяти и набора функциональных блоков различного назначения, связанных системными шинами.

Процессорное ядро является основой микроконтроллера. Оно выполняет все арифметические операции, управляет работой всех остальных элементов системы.

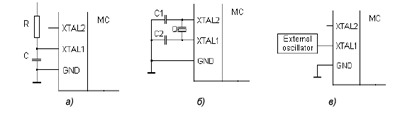
В памяти хранится программа работы микроконтроллера, исходные данные и все промежуточные результаты вычислений.

Функциональные блоки обеспечивают связь микроконтроллера с внешним миром. Для процессорного ядра любой функ. блок представляется в виде одного или нескольких регистров. Каждый регистр имеет свой оригинальный адрес.



Внешние подключения условно можно разделить на выводы питания и информационные выводы. Информационные выводы включают в себя 4 порта ввода-вывода, выводы подключения кварцевого резонатора, вывод сброса. Контакты портов ввода-вывода могут быть использованы в качестве параллельных портов либо другими функциональными блоками, контакты которых объединены с контактами портов. Все выводы портов имеют индивидуальные подтягивающие резисторы. Для подключения этих резисторов с регистре SFIOR предусмотрен бит PUD.

Для повышения стабильности используется внешний тактовый генератор. Схема подключении приведена на рисунке.



Емкость С=13,,43пФ. Для выбора синхронизации используются биты CRSEL0-3.

**12. Сторожевой таймер**

Сторожевой таймер синхронизирован от отдельного внутреннего генератора на кристалле, работающего с частотой 1МГц (при напряжении питания 5В). Задержка сброса устанавливается с помощью предделителя (Prescaler). Настройка предделителя осуществляется установкой или сбросом битов WDP0…WDP2 регистра управления сторожевым таймером WDTCR. По истечении установленного времени задержки сторожевой таймер подает сигнал сброса на микроконтроллер.

Интервалы задержки сторожевого таймера

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WDP2 | WDP1 | WDP0 | Количество циклов | Задержка сброса |
| 0 | 0 | 0 | 16К | 15ms |
| 0 | 0 | 1 | 32К | 30ms |
| 0 | 1 | 0 | 64К | 60ms |
| 0 | 1 | 1 | 128К | 0,12s |
| 1 | 0 | 0 | 256К | 0,24s |
| 1 | 0 | 1 | 512К | 0,49s |
| 1 | 1 | 0 | 1,024К | 0,97s |
| 1 | 1 | 1 | 2,048К | 1,9s |

Сброс может быть предотвращен инструкцией wdr.

Бит WDE в регистре WDTCR позволяет подключить или отключить сторожевой таймер. При разрешении работы сторожевого таймера его состояние не определено и прежде, чем разрешать его включение, необходимо выполнить инструкцию wdr. В ином случае контроллер может быть сброшен прежде, чем будет выполнена команда wdr, прописанная после разрешения. Для предотвращения случайных ошибок запрет сторожевого таймера должен оформляться специальной процедурой выключения.

**13. Програмно\_управляемый\_обмен**

Порты ввода - предназначены для связи микроконтроллера с различными объектами и могут реализовывать различные алгоритмы обмена данными : асинхронный программный обмен , синхронный обмен , ввод - с сигналами квитирования . Обмен данными между портами и объектами обеспечивается специальными подпрограммами – драйверами , создаваемыми индивидуально для каждого объекта .

АСИНХРОННЫЙ ОБМЕН

В режиме асинхронного программного обмена ввод и вывод данных производится по программе в моменты выполнения инструкций ввода и вывода данных. Предполагается, что объект всегда готов к обмену: при вводе – данные в момент выполнения инструкции in присутствуют на линиях порта, при выводе – данные будут прочитаны с линий порта до следующего вывода. Например , микроконтроллер ATmega 63 осуществляет асинхронный вывод данных при выполнении фрагмента программы:

.equ porta = $B ;

.equ ddra = $A ;

.cseg

ldi r 6,$FF ; $FF в r 6

out ddra, r 6 ; порта А на вывод

out porta, r0 ; данных из регистра r0 в порт A.

Время выполнения команды вывода равно двум периодам тактового сигнала . Тот же микроконтроллер в течении двух тактов введет данные с линий порта при выполнении фрагмента программы :

.equ pina = $9 ;

.cseg

in r0, pina ; ввод данных из порта pina в регистр r0.

По адресу $9 в пространстве ввода - микроконтроллера размещен регистр

pina, с входных линий которого и будут взяты данные во время выполнения инструкции .

# СИМПЛЕКСНЫЙ ОБМЕН

Симплексным считается однонаправленный обмен данными . Такой обмен обычно является синхронным . В этом случае каждое изменение данных на линиях порта сопровождается сигналом синхронизации (). Строб генерируется источником данных и предназначается для задания момента записи данных в регистр приемника . При выводе данных сигнал строба должен сформировать микроконтроллер , используя для этого специальные линии шины управления или отдельные биты портов ввода -. На рис . 9.2 показан вариант соединения микроконтроллера ATmega 63 c посимвольным принтером , имеющим 8-битный вход для приема данных DATA, выход сигнала готовности READY и вход стробирования #STB. Активным на входе стробирования является сигнал низкого уровня . Порт PORTA микроконтроллера и бит PB6 порта PORTB программируются на вывод данных , бит PB2 порта PORTB на ввод . Подпрограмма вывода должна : осуществить проверку готовности принтера ( сигнала готовности принтера READY и его анализ ), при обнаружении сигнала READY=1 вывести данные в порт PORT , подтвердить вывод данных выводом сигнала стробирования #STB=0 для записи данных в принтер . Если принтер не готов к обмену микроконтроллер через заданный интервал времени повторяет операцию.

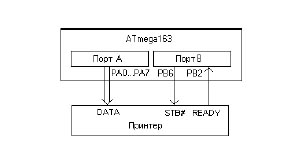


Рис . 9.2. Подключение принтера к микроконтроллеру

Режим синхронного ввода позволяет точно задать момент считывания данных с объекта . В этом режиме сигнал стробирования , подтверждающий готовность данных , поступает с объекта на микроконтроллер . При появлении строба микроконтроллер должен осуществить ввод данных с порта.

ПОЛУДУПЛЕКСНЫЙ ОБМЕН

Полудуплексным считается двунаправленный синхронный обмен , при котором в любой момент времени передача данных может производиться только в одном направлении . Направление передачи данных порта меняется в процессе работы в зависимости от решаемой в текущий момент задачи . На рис . 9.3 изображен вариант соединения микроконтроллера ATmega 63 с объектом для обмена данными в полудуплексном режиме . В схеме на рис . 9.3 передача данных в параллельном формате осуществляется по линиям порта PORTA. Для выдачи и приема четырех сигналов управления обменом ( квитирования ): строб ввода #STB IN (Strobe Input), строб вывода #STB OUT (Strobe Out), подтверждение ввода #ACK IN (Acknowledge Input) и подтверждение вывода #ACK OUT (Acknowledge Out), использованы две линии порта PORTB и входы запросов на прерывания INT0 и INT . При поступлении сигнала #STB IN = 0 контроллер должен выставить сигнал подтверждения #ACK IN = 0 и осуществить запись во входной регистр порта PORTA.. При низком уровне сигнала ASK IN объекту запрещается формировать новый сигнал #STB IN. По окончании записи контроллер снимает сигнал #ACK IN, разрешая повторную передачу данных . Работа микроконтроллера в режиме вывода аналогична . Микроконтроллер выводит данные в порт PORTA, подтверждая вывод сигналом #STB OUT = 0. Объект формирует сигнал подтверждения #ACK OUT = 0, сообщая контроллеру о готовности к приему данных . При снятии сигнала #ACK OUT контроллер должен снять сигнал #STB OUT и может вновь перейти к выводу данных . Одновременная передача данных в двух направлениях при полудуплексном обмене невозможна .

ДУПЛЕКСНЫЙ ОБМЕН

Дуплексным считается двунаправленный синхронный обмен , при котором в любой момент времени возможна передача данных в двух направлениях . В этом случае для передачи данных в каждом направлении выделяется свой однонаправленный порт . На рис . 9.4 изображен вариант соединения микроконтроллера ATmega 1 63 с объектом для обмена данными в дуплексном режиме



Рис . 9.3. Подключение микроконтроллера при полудуплексном обмене

Рис . 9.4. Подключение микроконтроллера при дуплексном обмене В схеме на рис . 9.4 порт PORTA работает на ввод данных , порт PORT – на вывод , а линии PC0 и PC порта PORTC использованы для вывода сигналов квитирования #ACK IN и #ACK OUT. Для ввода сигналов квитирования от объекта #STB IN и #ACK OUT использованы входы запросов на прерывания INT0 и INT . При таком подключении задача ввода данных в микроконтроллер получает больший приоритет чем задача вывода .

**14.Порты ввода-вывода** предназначены для связи микроконтроллера с различными объектами и могут реализовывать различные алгоритмы обмена данными:

**1)Асинхронный программный обмен 2)Синхронный обмен 3)Ввод-вывод с сигналами квитирования.**

Обмен данными между портами и объектами обеспечивается специальными подпрограммами – драйверами, создаваемыми индивидуально для каждого объекта.

**Асинхронный обмен.**

В режиме асинхронного программного обмена ввод и вывод данных производится по программе в моменты выполнения инструкций ввода и вывода данных. Предполагается, что объект всегда готов к обмену: при вводе – данные в момент выполнения инструкции in присутствуют на линиях порта, при выводе – данные будут прочитаны с линий порта до следующего вывода.

Пример программы для осуществления асинхронного вывода данных для Atmega163:

.equ porta=$1B; .equ ddra=$1A; .cseg; ldi r16,$FF;запись $FF в r16

out ddra,r16 ; включение порта А на вывод out porta,r0 ; вывод данных из регистра r0 в порт А.

Ввод данных с линий порта.

.equ pina=$19 .cseg in r0pina ; ввод данных из порта pina в регистр r0.

По адресу $19 в пространстве ввода – вывода микроконтроллера размещен регистр pina, с входных линий которого и будут взяты данные во время выполнения инструкции.

**Симплексный обмен.**

Симплексным считается однонаправленный обмен данными. Такой обмен обычно является синхронным. В этом случае каждое изменение данных на линиях порта сопровождается сигналом синхронизации (стробом). Строб генерируется источником данных и предназначен для задания момента записи данных в регистр приемника.При выводе данных сигнал строба должен сформировать микроконтроллер, используя для этого специальные линии шины управления или отдельные биты портов ввода – вывода.

Вариант соединения Atmega163 с посимвольным принтером, имеющим 8-битный вход для приема данных DATA, выход сигнала готовности READY и вход стробирования #STB. Активным на входе стробирования является сигнал низкого уровня. Порт А микроконтроллера и бит РВ6 порта PORTB программируются на вывод данных, бит PB2 порта PORTB на ввод. Подпрограмма вывода должна: осуществлять проверку готовности принтера, при обнаружении сигнала READY=1 вывести данные в порт PORTА, подтвердить вывод данных выводом сигнала стробирования #STB=0 для записи данных в принтер. Если принтер не готов к обмену микроконтроллер через заданный интервал времени повторяет операцию.



Режим синхронного ввода данных позволяет точно задать момент считывания данных с объекта. В этом режиме сигнал стробирования, подтверждающий готовность данных поступает с объекта в микроконтроллер. При появлении строба микроконтроллер должен осуществить ввод данных с порта.

Полудуплексный обмен.

Полудуплексным считается двунаправленный синхронный обмен, при котором в любой момент времени передача данных может производится только в одном направлении. Направление передачи данных порта меняется в процессе работы в зависимости от решаемой в текущий момент задачи. Пример: передача данных в параллельном формате осуществляется по линиям порта PORTA. Для выдачи и приема четырех сигналов управления обменом (сигналы квитирования): использованы две линии порта PORTB и входы запросов на прерывания INTO и INT1. При поступлении сигнала #STB IN=0 контроллер должен выставить сигнал подтверждения #ACK IN=0 и осуществить запись во входной регистр порта PORTA. При низком уровне сигнала #ACK IN объекта запрещается формировать новый сигнал #STB IN. По окончании записи контроллер снимает сигнал #ACK IN, разрешая повторную передачу данных. Работа микроконтроллера в режиме вывода аналогична. Микроконтроллер выводит данные в порт PORTA, подтверждая вывод сигналом #STB OUT=0. Объект формирует сигнал подтверждения #ACK OUT=0, сообщая контроллеру о готовности к приему данных. При снятии сигнала #ACK OUT контроллер должен снять сигнал #STB OUTи может вновь перейти к выводу данных.



Одновременная передача данных в двух направлениях при полудуплексном обмене невозможна.

**Дуплексный обмен**.

Дуплексным считается двунаправленный синхронный обмен, при котором в любой момент времени возможна передача данных в двух направлениях. В этом случае для передачи данных в каждом направлении выделяется свой однонаправленный порт. Порт PORTA работает на ввод данных, порт PORTB – на вывод, а линии PC0 и PC1 порта PORTC использованы для вывода сигналов квитирования #ACK IN и #STB OUT. Для ввода сигналов квитирования от объекта #STB IN и #ACK OUT использованы входы запросов на прерывания INT0 и INT1. При таком подключении задача ввода данных в микроконтроллер получает больший приоритет, чем задача вывода.

**15.** Одной из разновидностей **программно-управляемого обмена** данными с ВУ является обмен с прерыванием программы, отличающийся от асинхронного программно-управляемого обмена тем, что переход к выполнению команд, физически реализующих обмен данными, осуществляется с помощью специальных аппаратных средств. Команды обмена данными в этом случае выделяют в отдельный программный модуль - подпрограмму обработки прерывания. Задачей обработки прерывания как раз и является приостановка выполнения одной программы (ее еще называют основной программой) и передача управления подпрограмме обработки прерывания. Действия, выполняемые при этом, как правило, те же, что и при обращении к подпрограмме. Только при обращении к подпрограмме они инициируются командой, а при обработке прерывания - управляющим сигналом от ВУ, который называют "Запрос на прерывание" или "Требование прерывания".

Эта важная особенность обмена с прерыванием программы позволяет организовать обмен данными с ВУ в произвольные моменты времени, не зависящие от выполняемой программы. Таким образом, появляется возможность обмена данными с ВУ в реальном масштабе времени, определяемом внешней средой. Обмен с прерыванием программы существенным образом экономит время микропроцессора, затрачиваемое на обмен. Это происходит за счет того, что исчезает необходимость в организации программных циклов ожидания готовности ВУ, на выполнение которых тратится значительное время, особенно при обмене с медленными ВУ.

Прерывание программы по требованию ВУ не должно оказывать на прерванную программу никакого влияния кроме увеличения времени ее выполнения за счет приостановки на время выполнения подпрограммы обработки прерывания. Поскольку для выполнения подпрограммы обработки прерывания используются различные регистры, то информацию, содержащуюся в них в момент прерывания, необходимо сохранить для последующего возврата в прерванную программу.

Формирование сигналов прерываний - запросов ВУ на обслуживание происходит в контроллерах соответствующих ВУ. В простейших случаях в качестве сигнала прерывания может использоваться сигнал "Готовность ВУ", поступающий из контроллера ВУ. Однако такое простое решение обладает существенным недостатком - процессор не имеет возможности управлять прерываниями, т. е. разрешать или запрещать их для отдельных ВУ. В результате организация обмена данными в режиме прерывания с несколькими ВУ существенно усложняется.

Для решения этой проблемы регистр состояния и управления контроллера ВУ (рис. 3.11) дополняют еще одним разрядом - "Разрешение прерывания". Запись 1 или 0 в разряд "Разрешение прерывания" производится программным путем по одной из линий шины данных системного интерфейса. Управляющий сигнал системного интерфейса "Запрос на прерывание" формируется с помощью схемы совпадения только при наличии единиц в разрядах "Готовность ВУ" и "Разрешение прерывания" регистра состояния и управления контроллера.

Аналогичным путем решается проблемам управления прерываниями в микроконтроллера, в целом. Для этого в регистре состояния выделяется разряд, содержимое которого определяет, разрешены или запрещены прерывания от внешних устройств. Значение этого разряда может устанавливаться программным путем.

В микроЭВМ обычно используется одноуровневая система прерываний, т. е. сигналы "Запрос на прерывание" от всех ВУ поступают на один вход процессора. Поэтому возникает проблема идентификации ВУ, запросившего обслуживание, и реализации заданной очередности (приоритета) обслуживания ВУ при одновременном поступлении нескольких сигналов прерывания. Существуют два основных способа идентификации ВУ, запросивших обслуживания:   
- программный опрос регистров состояния (разряд "Готовность ВУ") контроллеров всех ВУ;   
- использование векторов прерывания.

Организация прерываний с программным опросом готовности предполагает наличие в памяти микроконтроллера единой подпрограммы обслуживания прерываний от всех внешних устройств.

Обслуживание ВУ с помощью единой подпрограммы обработки прерываний производится следующим образом. В конце последнего машинного цикла выполнения очередной команды основной программы проверяется наличие требования прерывания от ВУ. Если сигнал прерывания есть и в микропроцессоре прерывание разрешено, то он переключается на выполнение подпрограммы обработки прерываний.

После сохранения содержимого регистров, используемых в подпрограмме, начинается последовательный опрос регистров состояния контроллеров всех ВУ, работающих в режиме прерывания. Как только подпрограмма обнаружит готовое к обмену ВУ, сразу выполняются действия по его обслуживанию. Завершается подпрограмма обработки прерывания после опроса готовности всех ВУ и восстановления содержимого регистров процессора.

Приоритет ВУ с программным опросом готовности внешнего устройства однозначно определяется порядком их опроса в подпрограмме обработки прерываний. Чем раньше в подпрограмме опрашивается готовность ВУ, тем меньше время реакции на его запрос и выше приоритет. Необходимость проверки готовности всех внешних устройств существенно увеличивает время обслуживания тех ВУ, которые опрашиваются последними. Это является основным недостатком рассматриваемого способа организации прерываний. Поэтому обслуживание прерываний с опросом готовности ВУ используется только в тех случаях, когда отсутствуют жесткие требования на время обработки сигналов прерывания внешних устройств.

Организация системы прерываний с использованием векторов прерываний позволяет устранить указанный недостаток. При такой организации системы прерываний ВУ, запросившее обслуживания, само идентифицирует себя с помощью вектора прерывания - адреса ячейки основной памяти, в которой хранится либо первая команда подпрограммы обслуживания прерывания данного ВУ, либо адрес начала такой подпрограммы. Таким образом, процессор, получив вектор прерывания, сразу переключается на выполнение требуемой подпрограммы обработки прерывания. В микроконтроллерах с векторной системой прерывания каждое ВУ должно иметь собственную подпрограмму обработки прерывания.

Различают векторные системы с интерфейсным и внеинтерфейсным вектором. В первом случае вектор прерывания формирует контроллер ВУ, запросившего обслуживания, во втором - контроллер прерываний, общий для всех устройств, работающих в режиме прерываний.

**17. Работа с прерываниями в МК Atmega 163**

Контроллер Atmega163 имеет 17 внешних и 15 внутренних прерываний. Каждый узел МК является источником прерываний и может прервать выполняемую программу. Каждое прерывание имеет фиксированный приоритет и вектор прерывания.

Вектора прерываний занимают в пространстве памяти программа адреса с $0 до $22. $0 – определен, как сигнал сброса. Адрес последующего больше предыдущего на 2. Адреса $2 и $4 соответствуют внешним прерываниям int0 и int1. Приоритет прерываний уменьшается с возрастанием адресов и векторов. Любое прерывание может быть запрещено специальными битами в регистрах I/O МК.

Для выхода из любой подпрограммы обработки прерываний в системе МК предусмотрена команда reti, восстанавливающая содержимое программного счетчика из стека. Для правильного функционирования системы прерываний в начале прграммы должен быть загружен указатель стека.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| биты | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| SREG $3F | I |  |  |  |  |  |  |  |
| MCUCR $35 |  |  |  |  | ISC11 | ISC10 | ISC01 | ISC00 |
| GIMSK $3B | INT1 | INT0 |  |  |  |  |  |  |

Внешние прерывания вызываются с контактов int1 и int0. для их обработки в МК задействованы отдельные биты трех регистров:

Регистр статуса SREG

Регистр управления процессорным ядром MCUCR

Регистр маски прерываний.

Прерывания разрешаются только при единичном значении бита I в регистре SREG. Сброс этого бита запрещает все прерывания. Если прерывания разрешены, то при появлении любого запроса бит I сбрасывается, а дальнейшие прерывания запрещаются. Прога usera может вновь установить этот бит, разрешив вложенные прерывания.

Биты ISC в регистре MCUCR описывают уровни и фронты прерывающих сигналов на контактах int0/1

|  |  |  |
| --- | --- | --- |
| ISC11/01 | ISC10/00 |  |
| 0 | 0 | Низкий уровень сигнала INT1/0 генерирует запрос на прерывание |
| 0 | 1 | Любое логическое изменение INT1/0 генерирует запрос на прерывание |
| 1 | 0 | Задний фронт импульса на INT1/0 генерирует запрос на прерывание |
| 1 | 1 | Переднийфронт импульса на INT1/0 генерирует запрос на прерывание |

биты 3 и 2, биты активизации входа INT1

биты 1 и 1, биты активизации входа INT0

В регистре маскирования прерываний GIMSK содержаться флаги внешних прерываний. Когда счетчик команд МК устанавливается на конкретный вектор прерывания, соотв. Флаг в GIMSK аппаратно сбрасывается. Флаги можно сбросить, записав туда лог. 1.

Бит 7 – int1 внешнее прерывание int 1 разрешено. Если бит int1 установлен и бит I в SREG =1, то внешний выход запроса на прерывание int1 становится активным

Бит 6 – int0 тоже самое.

**18. Организация обмена прямым доступом к памяти (ПДП)**

Одним из способов обмена данными с ВУ является обмен в режиме прямого доступа к памяти (ПДП). В этом режиме обмен данными между ВУ и основной памятью микроЭВМ происходит без участия процессора. Схема, управляющая обменом в режиме ПДП, размещаются в специальном контроллере, который называется КПДП. Обмен данными в режиме ПДП позволяет использовать в микроЭВМ быстродействующие внешние запоминающие устройства, такие, например, как накопители на жестких магнитных дисках, поскольку ПДП может обеспечить время обмена одним байтом данных между памятью и ВЗУ, равное циклу обращения к памяти.



Для реализации режима прямого доступа к памяти необходимо обеспечить непосредственную связь контроллера ПДП и памяти микроЭВМ. Контроллер ПДП подключается к памяти посредством шин адреса и данных системного интерфейса.

Существуют две разновидности прямого доступа к памяти с "захватом цикла". Наиболее простой способ организации ПДП состоит в том, что для обмена используются те машинные циклы процессора, в которых он не обменивается данными с памятью. В такие циклы контроллер ПДП может обмениваться данными с памятью, не мешая работе процессора. Однако возникает необходимость выделения таких циклов, чтобы не произошло временного перекрытия обмена ПДП с операциями обмена, инициируемыми процессором. В некоторых процессорах формируется специальный управляющий сигнал, указывающий циклы, в которых процессор не обращается к системному интерфейсу. Такой обмен в режиме ПДП возможен только в случайные моменты времени одиночными байтами или словами.

Более распространенным является ПДП с "захватом цикла" и принудительным отключением процессора от шин системного интерфейса. Для реализации такого режима ПДП системный интерфейс микроЭВМ дополняется двумя линиями для передачи управляющих сигналов "Требование прямого доступа к памяти" (ТПДП) и "Предоставление прямого доступа к памяти" (ППДП).

Управляющий сигнал ТПДП формируется контроллером прямого доступа к памяти. Процессор, получив этот сигнал, приостанавливает выполнение очередной команды, не дожидаясь ее завершения, выдает на системный интерфейс управляющий сигнал ППДП и отключается от шин системного интерфейса. С этого момента все шины системного интерфейса управляются контроллером ПДП. Контроллер ПДП, используя шины системного интерфейса, осуществляет обмен одним байтом или словом данных с памятью микроЭВМ и затем, сняв сигнал ТПДП, возвращает управление системным интерфейсом процессору. Как только контроллер ПДП будет готов к обмену следующим байтом, он вновь "захватывает" цикл процессора и т.д. В промежутках между сигналами ТПДП процессор продолжает выполнять команды программы. Тем самым выполнение программы замедляется, но в меньшей степени, чем при обмене в режиме прерываний.

Применение в микроЭВМ обмена данными с ВУ в режиме ПДП всегда требует предварительной подготовки, а именно: для каждого ВУ необходимо выделить область памяти, используемую при обмене, и указать ее размер, т.е. количество записываемых в память или читаемых из памяти байт (слов) информации. Следовательно, контроллер ПДП должен обязательно иметь в своем составе регистр адреса и счетчик байт (слов). Перед началом обмена с ВУ в режиме ПДП процессор должен выполнить программу загрузки. Эта программа обеспечивает запись в указанные регистры контроллера ПДП начального адреса выделенной ВУ памяти и ее размера в байтах или словах в зависимости от того, какими порциями информации ведется обмен. Сказанное не относится к начальной загрузке программ в память в режиме ПДП. В этом случае содержимое регистра адреса и счетчика байт слов устанавливается переключателями или перемычками непосредственно на плате контроллера.

**21. Интерфейс UART.**

Асинхронный последовательный интерфейс UART (Universal Asynchronous Receiver Transmitter – универсальный асинхронный приемопередатчик) обеспечивает полудуплексный режим обмена по трем линиям. В обмене всегда участвуют только два устройства, одно из которых является передатчиком, второе – приемником.

В режиме асинхронной передачи каждое слово данных передается автономно и передача может быть начата в любой момент времени. Стандартный формат асинхронной передачи:



Передача начинается со стартового (нулевого) бита. Затем передается от 5 до 8 бит данных.

Подразумевается, что приемник и передатчик работают на одной скорости. Внутренний генератор синхронизации запускается при обнаружении стартового бита.

В простейшем случае асинхронный приемопередатчик имеет две сигнальные линии:

TxD (Transmit Data)-выход,

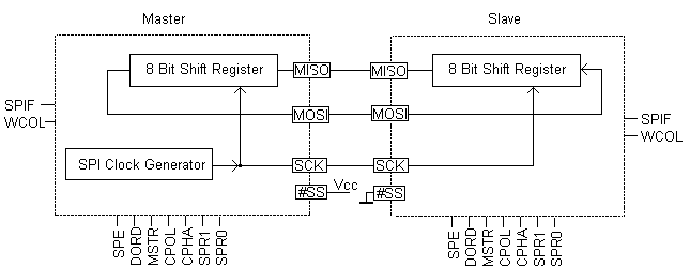
RxD (Receive Data)-вход.

При этом два устройства-приемопередатчика должны быть соединены между собой тремя линиями, или мак называемым нуль-модемным кабелем:



**Интерфейс SPI.**

Последовательный интерфейс SPI(Serial Peripheral Interface) обеспечивает полный дуплексный обмен данными между двумя контроллерами. При этом один контроллер считаемся ведущим (master), второй – ведомым (slave). Ведущий контроллер является источником сигнала синхронизации (SCK).



Ведущий и ведомый контроллеры связаны тремя сигнальными линиями:

MISO (Master In Slave Out) – вход ведущего – выход ведомого.

MOSI (Master Out Slave In) – выход ведущего – вход ведомого.

SCK (Serial Clock) – синхронизация.

Контакт #SS (Slave Select) – предназначен для выбора ведомого контроллера. Контроллер является ведомым если #SS=0.

Регистры сдвига (Shift Register) ведущего и ведомого контроллеров по линиям MOSI и MOSO соединяются в кольцо. Запись в регистр данных ведущего контроллера запускает генератор синхронизации (SPI clock generator) и данные сдвигаются в регистрах сдвига соединенных в кольцо ведущего и ведомого контроллеров.

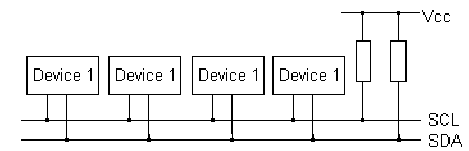
**Интерфейс I2C.**

2-проводной последовательный интерфейс (2-Wire Serial Interface) или I2C (Inter-Integrated Circuit) поддерживает двунаправленную последовательную связь нескольких устройств в полудуплексном режиме. В системе все устройства, участвующие в обмене, связываются двумя сигнальными линиями:

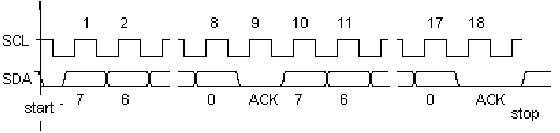
SDA (Serial Data) – данные.

SCA (Serial Clock) – синхронизация.

Типичная 2-х проводная последовательная конфигурация шины:



Каждое из устройств может выступать в роли передатчика или приемника. Синхронизацию обмена обеспечивает передатчик. Двунаправленную линию данных, выполненную по схеме «открытый коллектор» используют передатчик и приемник поочередно.



Начало любой операции (Start) инициализируется переводом сигнала SDA из высокого уровня в низкий при высоком уровне SCL. Завершается обмен переводом сигнала SDA из низкого уровня в высокий при высоком уровне SCL (Stop).

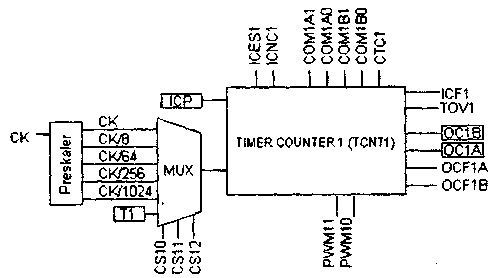
При передаче данных состояние линии SDA может изменяться только при низком уровне SCL. Биты данных стробируются положительным фронтом SCL.

Каждая посылка, формируемая передатчиком, состоит из байта данных. Посылка начинается со старшего бита. После чего передатчик на один такт освобождает линию, а приемник формирует нулевой сигнал подтверждения Ack (Acknowledge).

Каждое ведомое устройство имеет свой 7-битный адрес. Семь бит адреса передаются ведущим устройством в битах [7-1] первого байта.

**25.** 16-бит т/с1 мик-ра доступен проц-му ядру для чтения и записи, он может считать импульсы синхронизации CK импульсы с выхода предделителя или импульсы с внешнего вывода Т1.

Шесть 16 бит рег-в обеспечивают режимы вх-го захвата, вых-го сравнения и ШИМ.



16-бит т/с1 считает импульсы со входа Т1 или тактовые импульсы СК.

Переключением входа управляют биты CS10....CS12.

При переполнении - флаг запроса на прерыв-е TOV1.

регистры упр т/с1 TCCR1A и TCCR1B ;

регистры данных TCNT1H и TCNT1L;

регистры выходного сравнения OCR1AH и OCR1AL ;

регистры выходного сравнения OCR1BH и OCR1BL ;

регистры входного захвата - ICR1H и ICR1L ;

регистр флагов прерывания TIFR;

регистр маски прерывания TIMSK ;

регистр состояния микроконтроллера SREG .

Регистр масок прерываний TIMSK уст-ся разрешения/запрещения прерываний т/с1:

Бит 5 -TICIE1 бит разрешения прерывания при срабатывании входа захвата.

Бит 4 - OCIE1A бит разрешения прерывания при равенстве содержимого счетчика и содержимого регистра сравнения OCR1 А.

Бит 3 - OCIE1B бит разрешения прерывания при равенстве содержимого счетчика и содержимого регистра сравнения OCR1B.

Бит 2 - TOIE1 бит разрешения прерывания при переполнении таймера/счетчика 1.

Регистр флагов прерываний T1FR фиксируются события, являющиеся источниками прерываний:

Бит 5 - ICF1 флаг прерывания при возникновении захвата.

Бит 4 - OCF1A флаг прерывания при равенстве содержимого счетчика и содержимого регистра сравнения OCR1A.

Бит 3 - OCF1B флаг прерывания при равенстве содержимого счетчика и содержимого регистра сравнения OCR1B.

Бит 2 - TOV1 флаг прерывания при переполнении таймера/счетчика 1.

В регистре управления TCCR1B находятся биты для переключения входа счетчика/таймера 1.

Упр-е входом т/с1. Рег-ы данных TCNT1H и TCNT1L содержат зн-е т/с 1.Если проц-е ядро производит запись в TCNT1H, то данные размещаются в регистре TEMP. Затем, когда процессорное ядро производит запись вTCNT1L данные младшего байта объед-я с байтом данных регистра TEMP и все 16 битов одновременно переписываются в регистр таймера/счетчика TCNT1.Если CPU считывает TCNT1L, то содержимое TCNT1L направляются непосредственно в процессорное ядро, а содержимое TCNT1Н размещается в регистре TEMP. При считывании TCNT1H его содержимое будет изъято из регистра TEMP.

ВХОД ЗАХВАТА. Ф-я захвата заключается в копировании содержимого т/с1 в регистр входа захвата ICR1.

Регистры входа захвата ICR1H и ICR1L образуют 16-битный регистр ICR1, доступный только для чтения.

При нарастающем или падающем фронте сигнала ICP (опр уст-ой бита ICES1) состояние т/с 1 пересылается в регистр входа захвата ICR1. Одновременно устанавливается в состояние 1 флаг захвата входа ICF1 в регистре флагов прерывания TIFR

Для упр-я входом захвата используются биты TCCR1B :

Бит 7 - ICNC1 установка режима подавления шума на входе захвата 1. При установленном в 1 бите ICNC1 импульс, поступивший на вход захвата IC1 подвергается серьезной проверке - состояние входа IC1 опр-ся 4 раза.

Бит 6 - ICES1 выбор фронта срабатывания на входе захвата 1. При сброшенном в состояние 0 бите ICES1 содержимое т/с 1 пересылается в регистр захвата входа ICR1 по падающему фронту на входе.

ВЫХОДЫ СРАВНЕНИЯ. Т/с1 поддерживает два выхода сравнения, OCR1А и OCR1B в качестве источников данных, сравниваемых с содержимым таймера/счетчика 1.

Регистры выходов сравнения OCR1AH и OCR1AL , OCR1BH и OCR1BL образуют два 16-разрядных регистра OCR1A и OCR1B. OCR1A и OCR1B 16-битные то для обеспечения одновременного занесения старшего и младшего байтов данных в регистры OCR1A/B используется регистр временного хранения TEMP.

Регистр управления TCCR1A: Биты 7,6 - СОМ1А1, СОМ1А0 биты задания режима выхода А Биты 5,4 - СОМ1В1, СОМ1В0 биты задания режима выхода В. При изм битов СОМ1Х1/СОМ1Х0 прерывания должны быть запрешены.

Регистр TCCR1B: Бит 3 - СТС1 очистка таймера/счетчика 1 no совпадению. ШИМ.Т/с 1 может быть использован в качестве 8, 9 или 10-разрядного ШИМ. Работой PWM управляют отдельные биты регистра управления TCCR1A:

Биты 1,0 - PWM11, PWM 10 биты выбора режима широтно-импульсной модуляции. Данные биты определяют установку режима ШИМ.

Таймер/счетчик1 в этом случае работает как реверсивный счетчик, считающий от $0000 до значения ТОР, при котором направление счета меняется и отсчет ведется до нуля, после чего цикл повторяется.

ТОР значения таймера и частота ШИМ

|  |  |  |
| --- | --- | --- |
| Разрешение ШИМ | ТОР значения таймера | Частота ШИМ |
| 8-разрядное | $OOFF (255) | f/510 |
| 9-разрядное | $01FF(511) | f/1022 |
| 10-разрядное | $03FF(1023) | f /2046 |

В ШИМ режиме флаг переполнения таймера 1 (TOV1) устанавливается при смене направления счета по достижении значения $0. Прерывание по переполнению таймера/счетчика 1 работает так же как и в обычном режиме таймера/счетчика, т.е. оно выполняется когда флаг TOV1 в регистре T1FR установлен, установлен бит I в регистре SREG и разрешены прерывания по переполнению таймера 1.

**27.Шина современного персонального компьютера.**

Шина – общий канал связи, используемый в компьютере. Применяется она для организации взаимодействия между двумя или более компонентами системы.

Шина процессора.

Это высокоскоростная шина является ядром набора микросхем системной платы. Используется в основном процессором для передачи данных между кэш-памятью или основной памятью и компонентом North Bridge набора микросхем. В системах на базе процессора Pentium 2 эта шина работает на частоте 66 или 100МГц и имеет ширину 64 разряда. В системах на базе процессора Pentium 4 – 800МГц.

Шина AGP.

Эта 32 разрядная шина работает на частоте 66МГц и предназначена для подключения видеоадаптера. Она подключается к компоненту North Bridge набора микросхем системной логики.

Шина PCI.

Эта 32 разрядная шина работает на частоте 33,3; 66,6; 133,3 МГц. Используется, начиная с систем на базе процессора 486. Находится под управлением контроллера PCI – части компонента North Bridge набора микросхем. На системной плате устанавливаются разъёмы, 4 и более, в которые можно подключать SCSI, сетевые и видеоадаптеры, а также другое оборудование. К шине PCI подключается компонент South Bridge набора микросхем, который содержит реализацию интерфейса IDE и USB.

Шина ISA.

Эта 16 разрядная шина работает на частоте 8МГц; впервые стала использоваться в 1984г. Реализуется с помощью компонента South Bridge.

**Шина памяти.**

Предназначена для обмена информацией между процессором и основной памятью. Эта шина связана с помощью основного компонента North Bridge набора микросхем системной логики. Заметим, что разрядность шины памяти равна разрядности шины процессора. Разрядность шины определяет размер банка памяти.

|  |  |  |  |
| --- | --- | --- | --- |
| Тип шины | Разрядность, бит | Частота, МГц | Скорость передачи данных, Мбайт/с |
| 8 разрядная ISA | 8 | 4,77 | 2,39 |
| 16 разрядная ISA | 16 | 8,33 | 8,33 |
| EISA\* | 32 | 8,33 | 33,3 |
| VLB\* | 32 | 33,3 | 133,3 |
| PCI | 32 | 33,3 | 133,3 |
| PCI-2x | 32 | 66,6 | 266,6 |
| 64 разрядная PCI | 64 | 33,3 | 266,6 |
| 64 разрядная PCI-2x | 64 | 66,6 | 533,3 |
| 64 разрядная PCI-X | 64 | 133,3 | 1066,6 |
| AGP | 32 | 66,6 | 266,6 |
| AGP-2x | 32 | 66,6 | 533,3 |
| AGP-4x | 32 | 66,6 | 1066,6 |
| AGP-8x | 32 | 66,6 | 2133,3 |

\* - В настоящее время эти шины не используются.

Сегодняшние многогигагерцевые микросхемы нуждаются в постоянном потоке данных, а устаревающие шины стандартов PCI и AGP могут оказаться недостаточно быстрыми для перемещения данных между компонентами. Именно поэтому компания Intel разработала спецификацию на шину ввода-вывода третьего поколения под кодовым названием Arapahoe, скорость передачи данных по которой может в десять раз превосходить соответствующий показатель нынешнего «короля скорости» — шины PCI-X. Последняя выполняет параллельную передачу данных по 64 линиям, достигая максимальной скорости 1 Гбайт/с. Шина Arapahoe использует от 1 до 32 каналов, причем каждый из них состоит из пары проводников и достигает производительности более 200 Мбайт/с, перенося данные между ЦП и платами расширения или интегрированными компонентами. Arapahoe также может устанавливать приоритеты при обработке данных таким образом, что потоки информации в реальном времени будут обрабатываться быстрее. (Основное назначение этой шины все-таки обеспечение связи между южным и северным мостами набора микросхем.) Появление- начало 2004 г.

Процессор Pentium 3

Кэш-память L1

Кэш-память L2

Шина процессора 100МГц

Частота = (Частота процессора)/2

AGP

North Bridge

Шина PCI

Video

Шина ISA

100МГц

100МГц

SDRAM

DIMM

Разъёмы PCI

South Bridge

IDE1

USB11

IDE2

USB21

GMOS

Super I/O

Разъёмы ISA

ROM

Клав. Мышь

COM1

Дисков.

COM2

LPT1

Flash BIOS

**Рис.1. Архитектура системы на базе процессора Pentium 3.**

Поскольку шина процессора должна обмениваться информацией с процессором с максимально высокой скоростью, в компьютерах она функционирует намного быстрее любой другой шины. Сигнальные линии (линии электрической связи), представляющие шину, предназначены для передачи данных, адресов и сигналов управления между отдельными компонентами компьютера. Например, в компьютерах с процессором Pentium шина процессора состоит из 64 линий данных, 32 линий адреса и соответствующих линиях управления. Компьютеры с процессорами Pentium 2 и Pentium Pro имеют по 36 линий адреса. В большинстве современных компьютерах соотношение частот процессора и шины соответствует одному из коэффициентов: 1,5x, 2x …

**28.**

**Терминал** – объединенный в одном корпусе монитор и клавиатура, при этом м\п может находиться далеко.

**Клавиатура** – используется для ввода команд м\п.



Внутри клавы находится микроконтроллер. При нажатии клавиши – прерывание. При отпускании – еще одно.

**Мониторы.**

ЭЛТ и ЖК, на светодиодных матрицах, на газоразрядных индикационных паневях.

ЭЛТ.



**ЖК.**

Молекулы кристалла в одну сплошную линию. Если подать поляризующее напряжение – молекулы выстроятся в четкую цепь. Оптические свойства зависят только от освещенности.



**Мышь.**

Механическая, оптико-механическая, оптическая.

Механическая – на 2-х колесиках.

Оптико-механическая – шарик вращает шестеренки, зубцы которых перекрывают сигнал оптопары.

Мышь передает 3 байта информации:

1б – движение по горизонтали.

2б – движение по вертикали.

3б – нажатие клавиш.

**Принтер.**

Принтер - это аппарат для распечатки текстов и графики.

Матричные, струйные, лазерные.

Матричные работают по принципу печатной машинки, то есть оставляют на бумаге оттиск от иголочки, на которую нанесена краска.

Струйные - на бумагу из специальных сопел выстреливается струйка краски.

Лазерные - внутри расположен барабан, на котором создается отпечатываемая картинка (сначала он намагничивается определенным образом, потом к разным его областям "прилипает" краска требуемых цветов), после этого прокатывается лист бумаги, вследствие чего и получается рисунок.

**29. Манипулятор мышь и печатающие устройства современного персонального компьютера.**

Существует три типа мышей. Это:

Механические, в них основным элементом являются датчики, отслеживающие движение шарика. Датчики механические, отсюда и название мышей. В результате этого движение мыши происходит не так плавно, да и долговечность ее тоже невелика

Оптомеханические. Аналогичны механическим, но движение шарика отслеживаются оптическими датчиками.

Оптические. Оптическая мышь посылает луч на непрозрачную поверхность, а после отражения луч поступает обратно в мышь и там анализируется электроникой, которая в зависимости от характеристик полученного сигнала и отслеживает два направления движения мыши, основываясь либо на углах падения, либо на каких-либо других признаках. Преимущество такой мыши - очень высокая надежность, достоверность и плавность движения.

Интерфейсы:

- COM, PS/2 Особой разницы между ними не наблюдается.

-USB лучше других, потому что, обеспечивают большее количество отсчетов в единицу времени, возможно горячее включение.

Разрешение мыши измеряется в dpi. Нормальное разрешение мыши составляет 200-900 dpi. Оптические мыши могут иметь разрешающую способность 1000 dpi и более.

Количество отсчетов в секунду зависит не от мыши, а от интерфейса. COM и PS/2 это число 40, а у USB больше 100.

Принтеры:

Лазерный: основа - барабан, отвечает за перенос изображения на бумагу. Он выполнен в виде металлического цилиндра, с тонкой пленкой фотопроводящего полупроводника. Во время печати по поверхности барабана с помощью тонкой проволоки или сетки, называемой коронирующим проводом, распределяется статический заряд. Лазер, управляемый соответствующей электроникой, генерирует световой луч, который, отражаясь от вращающегося зеркала, попадает на барабан и изменяет его статистический заряд в точке прикосновения. Лазер попадает на барабан только в тех точках, в которых должно быть изображаение, таким образом на барабане получается скрытая копия отпечатка.

Проходя мимо тонера, барабан притягивает его в точках, подвергшихся облучению лазером. Затем на поверхность барабана накладывается бумага, и тонер переходит на нее. Бумага проходит между двумя роликами с температурой около 180 градусов.

Струйный:

Используются тонкие сопла, которые выплевывают чернила. Методом осуществляется перенос чернил на бумагу. Их всего три:

Пьезоэлектрический метод. В каждое сопло установлен пьезокристалл, связанный с диафрагмой. Пьезоэлемент, сжимая и разжимая трубку, наполняет сопло чернилами. Краска, которая выдавилась наружу, оставляет на бумаге точку.

Метод газовых пузырей. Каждое сопло оборудовано электрическим нагревательным элементом, который при пропускании через него тока за несколько микросекунд нагревается до 500o С. Чернила закипают и образовывают пузырьки, которые стремятся вытолкнуть часть краски наружу. Затем нагревательный элемент остывает, газовый пузырь уменьшается в объеме и через отверстие из резервуара поступает новая порция чернил.

drop-on-demand. для подачи чернил в сопла используется нагревательный элемент. Однако вместе с ним дополнительно работает еще и специальный механизм. Такой способ обеспечивает более быстрое впрыскивание чернил

Матричный:

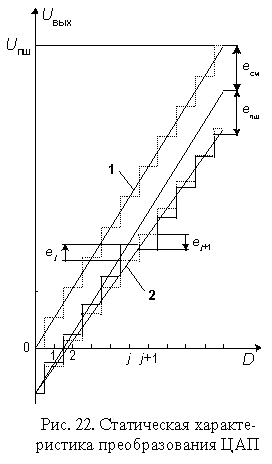
Используется головка с иголками, перемещающаяся по горизонтальной штанге. Из точек, оставляемых иголками с помощью красящей ленты, и формируется желаемое изображение. Так как головка представляет собой матрицу иголок, то принтеры обычно называют матричными.

**31. Цифро-аналоговые преобразователи**

Схемы применения цифро-аналоговых преобразователей относятся не только к области преобразования код - аналог. Пользуясь их свойствами можно определять произведения двух или более сигналов, строить делители функций, аналоговые звенья, управляемые от микроконтроллеров, такие как аттенюаторы, т.е. регуляторы уровня сигнала, интеграторы. Важной областью применения ЦАП являются также генераторы сигналов, в том числе сигналов произвольной формы. ЦАП используются в связи и передаче данных: модемах, фильтрах, устройствах самонастройки; измерительной и испытательной технике: источниках питания, генераторах, измерительных приборах; в технологических линиях: исполнительных устройствах роботов, станков и т.д.

   Параметры ЦАП

При последовательном возрастании значений входного цифрового сигнала D(t) от 0 до 2N-1 через единицу младшего разряда (ЕМР) выходной сигнал Uвых(t) образует ступенчатую кривую. Такую зависимость называют обычно характеристикой преобразования ЦАП. В отсутствие аппаратных погрешностей средние точки ступенек расположены на идеальной прямой 1 (рис. 22), которой соответствует идеальная характеристика преобразования. Реальная характеристика преобразования может существенно отличаться от идеальной размерами и формой ступенек, а также расположением на плоскости координат. Для количественного описания этих различий существует целый ряд параметров.



***Статические параметры***

Разрешающая способность - приращение Uвых при преобразовании смежных значений Dj, т.е. отличающихся на ЕМР. Это приращение является шагом квантования. Для двоичных кодов преобразования номинальное значение шага квантования h=Uпш/(2N-1), где Uпш - номинальное максимальное выходное напряжение ЦАП (напряжение полной шкалы), N - разрядность ЦАП. Чем больше разрядность преобразователя, тем выше его разрешающая способность.

Погрешность смещения нуля - значение Uвых, когда входной код ЦАП равен нулю. Является аддитивной составляющей полной погрешности. Обычно указывается в милливольтах или в процентах от полной шкалы:

.



**Нелинейность** - максимальное отклонение реальной характеристики преобразования Uвых(D) от оптимальной (линия 2 на рис. 22). Оптимальная характеристика находится эмпирически так, чтобы минимизировать значение погрешности нелинейности. Нелинейность обычно определяется в относительных единицах, но в справочных данных приводится также и в ЕМР. Для характеристики, приведенной на рис. 22



***Динамические параметры***

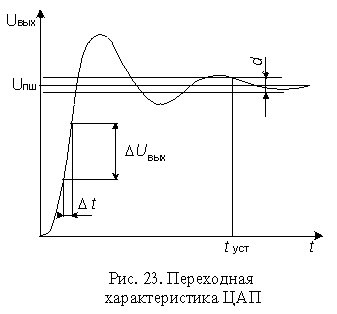
    Динамические параметры ЦАП определяются по изменению выходного сигнала при скачкообразном изменении входного кода, обычно от величины "все нули" до "все единицы" (рис. 23).

   Время установления - интервал времени от момента изменения входного кода (на рис. 23  t=0) до момента, когда в последний раз выполняется равенство

|Uвых-Uпш|=d/2,

причем d/2 обычно соответствует ЕМР.

    Скорость нарастания - максимальная скорость изменения Uвых(t) во время переходного процесса. Определяется как отношение приращения **Uвых ко времени **t, за которое произошло это приращение. Обычно указывается в технических характеристиках ЦАП с выходным сигналом в виде напряжения. У ЦАП с токовым выходом этот параметр в большой степени зависит от типа выходного ОУ.



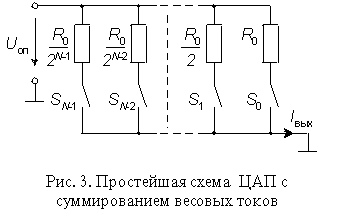
**Виды ЦАП**

Существуют последовательные и параллельные ЦАП. Последовательные –- используются в микропроцессорных системах, если не требуется высокое быстродействие. Среди параллельных - наиболее просты

**ЦАП с суммированием весовых токов**

    Большинство схем параллельных ЦАП основано на суммировании токов, сила каждого из которых пропорциональна весу цифрового двоичного разряда, причем должны суммироваться только токи разрядов, значения которых равны 1. Пусть, например, требуется преобразовать двоичный четырехразрядный код в аналоговый сигнал тока. У четвертого, старшего значащего разряда (СЗР) вес будет равен 23=8, у третьего разряда - 22=4, у второго - 21=2 и у младшего (МЗР) - 20=1. Если вес МЗР IМЗР=1 мА, то IСЗР=8 мА, а максимальный выходной ток преобразователя Iвых.макс=15 мА и соответствует коду 11112. Понятно, что коду 10012, например, будет соответствовать Iвых=9 мА и т.д. Следовательно, требуется построить схему, обеспечивающую генерацию и коммутацию по заданным законам точных весовых токов. Простейшая схема, реализующая указанный принцип, приведена на рис. 3.

    Сопротивления резисторов выбирают так, чтобы при замкнутых ключах через них протекал ток, соответствующий весу разряда. Ключ должен быть замкнут тогда, когда соответствующий ему бит входного слова равен единице. Выходной ток определяется соотношением



    При высокой разрядности ЦАП токозадающие резисторы должны быть согласованы с высокой точностью. Наиболее жесткие требования по точности предъявляются к резисторам старших разрядов, поскольку разброс токов в них не должен превышать тока младшего разряда. Поэтому разброс сопротивления в k-м разряде должен быть меньше, чем

R / R=2-k.

    Из этого условия следует, что разброс сопротивления резистора, например, в четвертом разряде не должен превышать 3%, а в 10-м разряде - 0,05% и т.д.

    Рассмотренная схема при всей ее простоте обладает целым букетом недостатков. Во-первых, при различных входных кодах ток, потребляемый от источника опорного напряжения (ИОН), будет различным, а это повлияет на величину выходного напряжения ИОН. Во-вторых, значения сопротивлений весовых резисторов могут различаться в тысячи раз, а это делает весьма затруднительной реализацию этих резисторов в полупроводниковых ИМС. Кроме того, сопротивление резисторов старших разрядов в многоразрядных ЦАП может быть соизмеримым с сопротивлением замкнутого ключа, а это приведет к погрешности преобразования. В-третьих, в этой схеме к разомкнутым ключам прикладывается значительное напряжение, что усложняет их построение.

**11. Система сброса.**

**Сброс- перевод МК в исходное состояние.** При этом все регистры микропроц. Ядра устанавливаются во вполне определенные начальные состояния, и МК переходит к выполнения программы с фиксированного адреса начального адреса (обычно $00).

Источниками сброса могут являться различные воздействия: включение питания и кратковременные его изменения, сигналы формируемые аппаратно внутри МК, а также инструкции программы. В частности, инструкция безусловного перехода на адрес $00.

Источники сброса МК Atmega163:

* Сброс при включении питания. Происходит, если напряжение питания ядра ниже определенного порога (Vpot)
* Внешний сброс. Происходит при поступлении сигнала низкого уросня длительностью >500нс на внешний контакт Reset микросхемы
* Сброс сторожевым таймером.
* Сброс при кратковременном провале напряжения питания. Происходим, если напряжение питания контроллера в процессе работы опускается ниже определенного порога(Vbot).

По любой из этих причин Мк переходит к выполнению программы с адреса $00. В этой ячейке размещают инструкцию jmp с адресом программы инициализации.

Все сигналы сброса детектируются на кристалле специальными схемами.

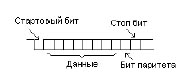
* Схема сброса при включении питания (Power on reset circuit) контролирует напряжение питания Vcc и запускается при Vcc>Vpot. При Vcc=5V номинал Vpot=1.4V
* Схема сброса при кратковременном провале Vcc(Brown out reset circuit) сравнивает Vcc с Vbot. Уровень Vbot программируется битом Bodlevel из группы з группы fuse-битов. При Bodlevel=1 Vbot=2.7V, Bodlevel=0 Vbot=4
* Схема внешнего сброса (External reset circuit) управляется внешним сигналом низкого уровня #Reset

Сигналя с этих схем и с WDT фиксируются в регистре состояния MCUSR, объединяются по схеме «или» и устанавливают RS-триггер.

Источник сброса может быть установлен путем чтения регистра MCUSR (биты 0-3)

# 20 ПОСЛЕДОВАТЕЛЬНЫЙ ВВОД-ВЫВОД

# Интерфейс UART



Асинхронный последовательный интерфейс UART (Universal Asynchronous Receiver

Transmitter – универсальный асинхронный приемопередатчик) обеспечивает

полудуплексный режим обмена по трем линиям. В обмене всегда участвуют только два

устройства, одно из которых является передатчиком, второе – приемником.

В режиме асинхронной передачи каждое слово данных передается автономно и

передача может быть начата в любой момент времени.

Стандартный формат асинхронной передачи изображен на рис. 13.1.

Передача начинается со стартового (нулевого) бита. Затем передается от 5 до 8

бит данных. Передача заканчивается необязательным битом четного (или нечетного)

паритета и одним (полутора или двумя) единичными стоповыми битами. После этого в

любой момент времени может быть начат цикл передачи следующего слова.

Подразумевается, что приемник и передатчик работают на одной скорости,

измеряемой числом бит в секунду (бод). Внутренний генератор синхронизации приемника

запускается при обнаружении стартового бита. В идеальном случае эти импульсы

располагаются в середине битовых интервалов.

Формат асинхронной передачи позволяет выявлять возможные ошибки:

\* если обнаружен стартовый бит и генератор синхронизации запущен, а по

первому импульсу синхронизации фиксируется уровень логической единице,

стартовый бит считается ложным;

\* если по импульсам синхронизации, соответствующим стоп-битам, в

приемнике фиксируется логический нуль, сообщение считается ошибочным

(ошибка кадра);

\* если контрольный бит не соответствует принятому соглашению о паритете,

фиксируется ошибка паритета.

Контроль формата позволяет обнаружить обрыв линии по отсутствию стоп-бита.

Для асинхронной передачи принят стандартный ряд скоростей: 50, 75, 110,150, 300,

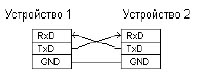
600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 бит/с.:

В простейшем случае асинхронный приемопередатчик имеет две сигнальные

линии:

TxD (Transmit Data)- выход,

RxD (Receive Data) – вход,



При этом два устройства-приемопередатчика должны быть соединены между

собой тремя линями, или так называемым нуль-модемным кабелем (рис.13.2).

# УПРАВЛЕНИЕ UART

Управление UART осуществляется через регистры ввода/вывода. В контроллере

ATmega163 для управления используется 5 регистров (рис. 13.3):



Регистр UDR (UART Data Register) – регистр данных UART

Регистр UСSRА (UART Control and Status Register A) -регистр А управления и

статуса UART

Регистр UCSRB (UART Control and Status Register B) - регистр В управления

и статуса UART

Регистры UBRRH1 и UBRR (UART Baud Rate registers) – регистры скорости

передачи.

Регистр данных UDR (UART Data Register) физически является двумя регистрами:

регистром передачи данных и регистром приема данных, использующими одни и те же

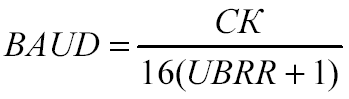
адреса $0C ($2C). При записи в регистр запись производится в регистр передачи данных

UART, при чтении происходит чтение содержимого регистра приема данных UART.

Скорость обмена данными в UART задается с помощью бод-генератора (Baud Rate

Generator). Он представляет собой делитель, генерирующий импульсы синхронизации с

частотой, определяемой выражением:



BAUD = частота в бодах (бит/сек),

CK = частота кварцевого генератора,

UBRR = содержимое 12-битного регистра UBRR (UART Baud Rate register).

Физически 12-битный регистр UBRR размещается в двух 8-битных регистрах.

Младшие 8 бит в регистре UBRR, старшие 4 бита – в регистре UBRRH1 (рис. 13.3).

При использовании стандартных кварцевых резонаторов, наиболее часто

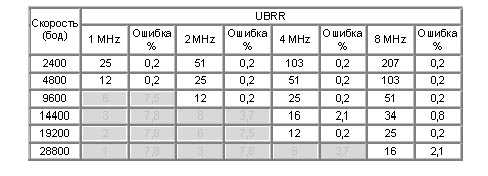
используемые скорости передачи в бодах могут быть получены установками UBRR,

представленными в таблице 13.1. При установках UBRR, указанных в таблице, реальные

скорости в бодах будут иметь отличия менее 2% от стандартных скоростей.

Таблица 13.1.

Установки UBRR при стандартных частотах синхронизации



Фактически, для регулирования скорости передачи UART достаточно только одного

регистра UDDR. В регистр UDDRH1 во всех рассмотренных случаях записывается

константа $00.

ПЕРЕДАТЧИК



Блок-схема передатчика UART показана на рис. 13.3.

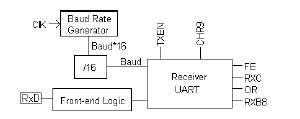


Рис. 13.3. Передатчик UART

Установленный в состояние 1 бит TXEN регистра UCSRB разрешает передачу

данных UART. Передача инициируется записью передаваемых данных в регистр данных

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 USR (UART Status Register). При установленном в состояние 1 бите UDRE

приемопередатчик готов принять следующий символ. Запись в UDR очищает бит UDRE. 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 UCSRB установлен бит CHR9 (т.е. выбран режим 9-разрядного слова

данных), то бит TXB8 регистра UCSRB пересылается в бит 9 сдвигового регистра передачи.

Сразу после пересылки данных в сдвиговый регистр тактом бод-генератора

стартовый бит сдвигается на вывод TxD. За ним следует LSB 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

управления и статуса UCSRA устанавливается флаг завершения передачи TxC (TX

Complete Flag).

ПРИЕМНИК

Структурная схема приемника UART приведена на рис. 13.4.

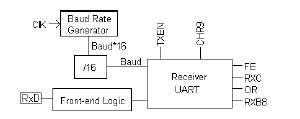


Рис. 13.4. Приемник UART

Логика восстановления данных (Front-End Logic) производит выборку состояний

вывода RxD с частотой в 16 раз большей, чем частота передачи. 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 RхD 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.

Тестирование выборок битов принимаемых символов показано на рис. 13.5.

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 при пересылке принятого байта в UDR в регистре

управления и статуса UСSRA устанавливается бит ошибки кадра FE (Framing Error). Для

обнаружения ошибки кадра пользователь перед чтением регистра UDR должен проверять

состояние бита FE. Флаг FE очищается при считывании содержимого регистра данных

UART (UDR).

Вне зависимости от того принят правильный стоповый бит или нет, данные

пересылаются в регистр UDR и устанавливается флаг RXC в регистре управления UCSRA.

Регистр UDR фактически является двумя физически отдельными регистрами, один из

которых служит для передачи данных и другой для приема. При считывании UDR

обращение ведется к регистру приема данных, при записи обращение ведется 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 из

девятого бита сдвигового регистра передачи. Если после получения символа к регистру

UDR не было обращения, начиная с последнего приема, в регистре UCSRA

устанавливается флаг переполнения OR. Это означает, что новые данные, пересылаемые

в сдвиговый регистр, не могут быть переданы в UDR и потеряны. Бит OR буферирован и

доступен тогда, когда в UDR читается байт достоверных данных. Пользователю, для

обнаружения переполнения, необходимо всегда проверять флаг OR после считывания

содержимого регистра UDR.

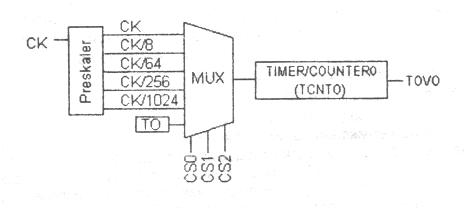
При очищенном (сброшенном в логическое состояние 0) бите RXEN регистра UCR

прием запрещен.

**24. Таймер-счетчик 0.**

8-ми разрядный Т/С 0тактируется сигналом синхронизации процессорного ядра или встроенного предделителя, или от внешнего контакта ТО.

Предназначен для выполнения простейших операций: его содержимое программно доступно процессорному ядру для чтения /записи, а при переполнении счетчика генерируется запрос на прерывание программы.



Регистры управления:

р-р управления TCCR0 - управляет тактовой частотой Т/С0 (биты 0-2)

р-р данных TCNT0 – хранит содержимое счетчика (Read/Write)

р-р флагов прерывания TIFR

р-р маски прерывания TIMSK

р-р состояния МК SREG

**22. Подключение клавиатуры и индикация.**

Большинство микропроцессорных систем в своей работе предполагают возможность изменения внутренних параметров программы, необходимость отображения для пользователя каких-либо данных. Т.о. возникает необходимость включения в систему средств ввода/вывода. Как правило для этого используются клавиатура и различные индикаторы.

Существует несколько способов подключения клавиатуры к портам ввода/вывода:

Проще всего на каждый вход непосредственно подключать одну клавишу. При этом для обеспечения помехозащищенности ненажатое состояние подключают к напряжению питания, через подтягивающий резистор, а при нажатии вход замыкается на землю.

Однако в первом случае не эффективно используются входы микроконтроллера, как правило находящиеся в дефиците, ведь обычно предполагается одновременное нажатие лишь одной клавиши. Можно подключить клавиатуру через мультиплексор (с инвертирующими входами), тогда на N входов микроконтроллера можно подключить до 2N клавиш.

Кроме того клавиатуру можно организовать в виде матрицы. Тогда в матрице из k столбцов и N строк, все клавиши одной строки подключаются к одному входу микроконтроллера/мультиплексора, а выбор столбца осуществляется подачей сканирующего сигнала на один из k выходов подключенных к столбцам матрицы. В случае, когда в ненажатом состоянии клавиши подтянуты к VCC, в качестве сканирующего сигнала ипользуется 0.

Выбор конкретной реализации зависит от неоходимого количества клавиш, свободных портов микроконтроллера, возможности использовать дополнительные микросхемы (мультиплексоров).

При разработке систем не следует не забывать и о подавлении эффекта «дребезга контактов». Защита от дребезга может осуществляться как программно, так и аппаратно. В последнем случае клавиши подключаются не на прямую, а через T-триггер.

Индикация может осуществляться с использованием: светодиодов сегментных индикаторов

Индикаторы могут быть как светодиодные, так и жидкокристаллические, и иметь различное число сегметов.

Как светодиоды, так и сегментные индикаторы могут подключаться к портам микроконтроллера напрямую, либо через дешифратор/демультиплексор.

Набор символов отображаемых сегментными индикаторами может определяться программно через таблицу декодирования, либо аппаратно логикой внешней микросхемы дешифратора.

Схемы подключения индикаторов к системе могут быть разными:

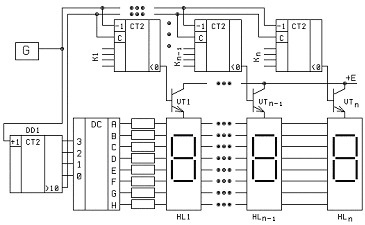
Простая: один порт – один индикатор

Различные варианты распараллеливания канала данных либо канала выбора индикатора

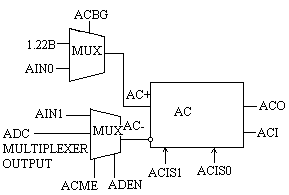
Динамическая индикация (число знаков ограничивается скважностью) Рис. 1

Динамическая индикация методом досчета (скважность постоянная) Рис. 2

|  |  |
| --- | --- |
|  |  |
| Рис. 1 | Рис. 2 |



**Билет 30. Аналоговые компараторы.**



Аналоговые компараторы (АК) осуществляют сравнение двух напряжений. Результатом сравнения является логический сигнал, фиксирующий момент равенства входных сигналов. Выход компаратора может быть использован в качестве запроса на прерывание. При этом пользователь может программировать формирование запроса по переднему или заднему фронту сигнала, либо по любому его изменению.

Схема компаратора микроконтроллера ATmega163.

Аналоговый компаратор сравнивает уровни на положительном (АС+) и отрицательном (АС-) входах. При напряжениии на входе АС+ большем, чем напряжение на входе АС- , выход аналогового компаратора АСО устанавливается в состояние 1.

В микроконтроллере ATmega163 качестве входа АС+ может быть использован внутренний источник напряжения 1,22 В либо вход AIN0(PB2). К АС- подключается AIN1(PB3), либо один из входов PA0..PA7 через мультиплексор.

Компаратор формирует запрос на прерывание если бит ACIE регистра ACSR установлен в 1. Формирование запроса возможно по переднему или заднему фронту сигнала, либо по любому его изменению.

Установка прерывания битами ACIS0/ACIS1.

|  |  |  |
| --- | --- | --- |
| ACIS0 | ACIS1 | Режим прерывания |
| 0 | 0 | Прерывание по любому переключению выхода компаратора |
| 0 | 1 | Зарезервировано |
| 1 | 0 | Прерывание по падающему (заднему) фронту на выходе компаратора |
| 1 | 1 | Прерывание по нарастающему (переднему) фронту на выходе компаратора |

При изменении состояния битов ACIS0/ACIS1 прерывание по аналоговому компаратору должно быть запрещено очисткой бита разрешения прерывания (ACIE) в регистре ACSR, иначе при изменении состояния битов может произойти прерывание.

В работе компаратора используются регистры:

ACSR (Analog Comparator Control and Status Register) – управления АК

SFIOR (Special Function Input Output Register) – специальных функций ввода/вывода

ADCSR (ADC Status Register) – состояния аналого-цифрового преобразователя

ADCMUX (ADC Multiplexer) – мультиплексора аналого-цифрового преобразователя

SREG (Status Register) – состояния микроконтролера

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Регистр | Биты | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ACSR | ADC | ACBG | ACO | ACI | ACIE | ACIC | ACIS1 | ACIS0 |
| SFIOR |  |  |  |  | ACME |  |  |  |
| ADCSR | ADEN |  |  |  |  |  |  |  |
| ADCMUX |  |  |  |  |  | MUX2 | MUX1 | MUX0 |
| SREG | I |  |  |  |  |  |  |  |

Подключение отрицательного входа компаратора

|  |  |  |  |
| --- | --- | --- | --- |
| ACME | ADEN | MUX2..0 | Отрицательный вход компаратора |
| 0 | x | xxx | AIN1 |
| 1 | 1 | xxx | AIN1 |
| 1 | 0 | 000 | PA0 |
| 1 | 0 | 001 | PA1 |
| 1 | 0 | 010 | PA2 |
| 1 | 0 | 011 | PA3 |
| 1 | 0 | 100 | PA4 |
| 1 | 0 | 101 | PA5 |
| 1 | 0 | 110 | PA6 |
| 1 | 0 | 111 | PA7 |

Значения битов:

ACD (Analog Comparator Disable) – если установлен в 1, АК отключен. При изменении бита прерывание от компаратора должно быть заблокировано очисткой бита ACIE.

ACBG (Analog Comparator Bandgap) – когда бит установлен и BOD позволяет (fuse‑бит BODEN запрограммирован), фиксированное напряжение 1,22 В поступает на вход АС+ , иначе к АС+ подключается контакт AIN0.

ACO (Analog Comparator Output) – выход аналогового компаратора

ACI (Analog Comparator Interrupt Flag) – устанавливается в 1 при формировании компаратором прерывания. Подпрограмма обработки прерывания будет выполняться при установленных битах ACIE и I(бит глобального прерывания в регистре SREG). Очищается аппаратно при выполнении подпрограммы обработки прерывания либо вручную. Очищается при модификации командами SBI, CBI других битов регистра ACSR.

ACIE (Analog Comparator Interrupt Enable) – установка в 1 разрешает прерывание по аналоговому компаратору (ANA\_COM) при установленном бите I в SREG.

ACIC (Analog Comparator Input Capture Enable) – установка в 1 разрешает захват входа таймера/счетчика1 по переключению АК.

ACIS1,ACIS0 (Analog ComparatorInterrupt Mode Select) выбор режима прерывания.

ACME (Analog Comparator Multiplexer Enable) – 1 подключает мультиплексор к АК. При подключении должен быть сброшен бит ADEN (ADC Enable) в регистре ADCSR (аналогово-цифровой преобразователь выключен).

Битами MUX2..0 в регистре мультиплексора ADMUX выбирается контакт порта PORTA (PA7..PA0).

**9. ПОРТЫ ВВОДА-ВЫВОДА**

9.1. Организация ввода/вывода

Порты ввода-вывода обеспечивают ввод и вывод данных в параллельном формате. Обычно порты ввода-вывода выполняются 8-разрядными. В режиме ввода данные с внешних контактов порта пересылаются в регистры микроконтроллера. В режиме вывода данные из регистров перемещаются на контакты микроконтроллера. Вывод данных, как правило, производится в «защелку» порта. Данные при этом присутствуют на выходных контактах до новой операции вывода в этот порт. В системе команд микроконтроллера для ввода и вывода данных обычно предусматриваются специальные команды.

В зависимости от выполняемых функций порты могут быть:

0 однонаправленными, предназначенными только для выполнения одной из

операций (ввод или вывод) по всем линиям;

° двунаправленными, предназначенными для выполнения любой из операций

ввода-вывода по всем линиям одновременно; направление передачи может

быть изменено программно в процессе работы;

° с индивидуальной настройкой линий, направление передачи данных по

каждой линии программируется независимо от остальных. Последний вариант построения схемы порта в настоящее время наиболее распространен. Например, микроконтроллер АТтеда163 имеет 32 линии ввода-вывода с индивидуальной настройкой, сгруппированных в 4 параллельных порта: порт А порт В, порт С, порт О. Направление передачи данных любого вывода любого порта может быть изменено независимо от направлений других выводов Для работы с портами в микроконтроллере предусмотрено 12 регистров, по три на каждый из портов (рис. 58):

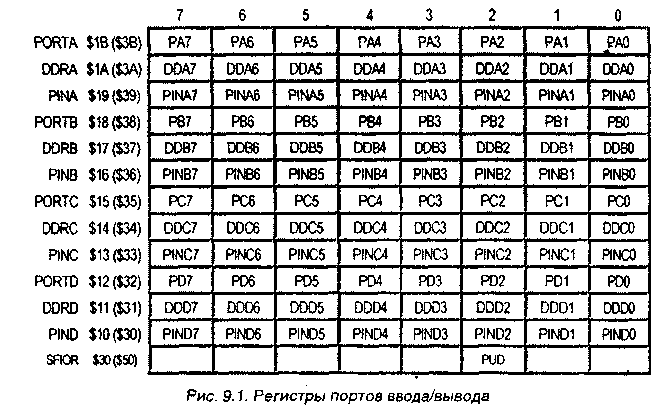
° регистры данных (Dafa Register): PORTA , POF< tb, PORTC и PORTD;

0 регистры направления (Data Direction Register): DDRA, DDRB, DDRC и

DDRD;

° регистры входных контактов (Port A Input Pins). PINA; PINB, PINC и FIND.

Все выводы портов имеют индивидуальные подтягивающие резисторы (pull-up resistors). Для подключения этих резисторов в регистре специальных функций ввода/вывода SFIOR (Special Function Input Output Register) предусмотрен бит PUD (Pull-up Disabled).



Любая линия порта выполняет функции выхода при записи логической единицы в соответствующий бит регистра направления DDRx (x e А, В, С, D).

Регистры PINx не хранят информацию и фактически не являются настоящими регистрами. Они разрешают доступ к физическим сигналам на линиях соответствующего порта. При чтении PORTx читается защелка данных выбранного порта, а при чтении PINx -значение на контактах порта. Регистры PINx доступны только для чтения, е то время как регистры PORTx и DDRx -для чтения и записи.

Внутренние подтягивающие резисторы подключаются только при PUD=0, если контакты портов сконфигурированы как входы

Выходы портов выдерживают втекающий ток до 20mA и могут быть непосредственно подключены к светодиодным индикаторам. Однако, вытекающий ток порта не должен быть более 4 мА, а суммарная загрузка порта - не более 80 мА.

9.2. Алгоритмы обмена данными

Порты ввода-вывода предназначены для связи микроконтроллера с различными объектами и могут реализовывать различные алгоритмы обмена данными:

0 асинхронный программный обмен,

0 синхронный обмен,

0 ввод-вывод с сигналами квитирования.

Обмен данными между портами и объектами обеспечивается специальными подпрограммами-драйверами, создаваемыми индивидуально для каждого объекта.

АСИНХРОННЫЙ ОБМЕН

В режиме асинхронного программного обмена ввод и вывод данных производится

По программе в моменты выполнения инструкций ввода и вывода данных. Предполагается,

Что объект всегда готов к обмену: при вводе - данные в момент выполнения инструкции in

присутствуют на линиях порта, при выводе -данные будут прочитаны с линий порта до следующего вывода.

Например, микроконтроллер ATmega163 осуществляет асинхронный вывод данных nпри выполнении фрагмента программы:

equporta = $1B ;

.equ ddra = $1А ;

cseg

ldir16,$FF ;запись$РРвг16

out ddra, r16 включение порта А на вывод

out porta, rO ;вывод данных из регистра гО в порт А.

Время выполнения команды вывода равно двум периодам тактового сигнала.

Тот же микроконтроллер в течении двух тактов введет данные с линий порта при имении фрагмента программы:

equpina = $19

cseg

in rO, pina ; ввод данных из порта pina в регистр гО.

По адресу $19 в пространстве ввода-вывода микроконтроллера размещен регистр pina, с входных линий которого и будут взяты данные во время выполнения инструкции.

симплексный обмен

Симплексным считается однонаправленный обмен данными. Такой обмен обычно является синхронным. В этом случае каждое изменение данных на линиях порта •провождается сигналом синхронизации (стробом). Строб генерируется источником данных и предназначается для задания момента записи данных в регистр приемника.

При выводе данных сигнал строба должен сформировать микроконтроллер, используя для этого специальные линии шины управления или отдельные биты портов ввода-вывода.

На рис. 9.2 показан вариант соединения микроконтроллера ATmega163 с посимвольным принтером, имеющим 8-битный вход для приема данных DATA, выход сигнала готовности READY и вход стробирования #STB. Активным на входе стробирования является сигнал низкого уровня.

Порт PORTA микроконтроллера и бит РВ6 порта PORTB программируются на вывод данных, бит РВ2 порта PORTB на ввод. Подпрограмма вывода должна:

0 осуществить проверку готовности принтера (чтение сигнала готовности

принтера READY и его анализ),

° при обнаружении сигнала READY=1 вывести данные в порт PORTA,

° подтвердить вывод данных выводом сигнала стробирования #STB=0 для

записи данных в принтер.

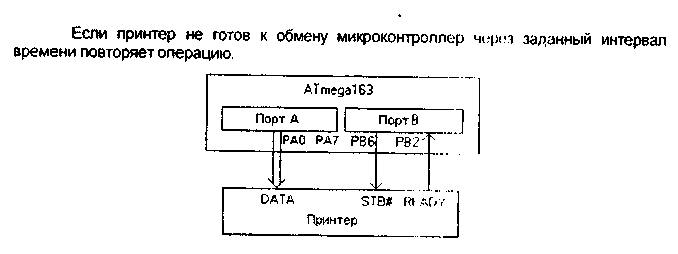


Рис. 9.2. Подключение принтера к микроконтроллеру

Режим синхронного ввода позволяет точно задать момент считывания данных с объекта. В этом режиме сигнал стробирования, подтверждающий готовность данных, поступает с объекта на микроконтроллер. При появлении строба микроконтроллер должен осуществить ввод данных с порта.

полудуплексный обмен

Полудуплексным считается двунаправленный синхронный обмен, при котором в любой момент времени передача данных может производиться только в одном направлении. Направление передачи данных порта меняется в процессе работы в зависимости от решаемой в текущий момент задачи. На рис. 9.3 изображен вариант соединения микроконтроллера ATmega163 с объектом для обмена данными в полудуплексном режиме.

В схеме на рис. 9.3 передача данных в параллельном формате осуществляется по линиям порта PORTA. Для выдачи и приема четырех сигналов управления обменом (сигналы квитирования): строб ввода #STB IN (Strobe Input), строб вывода #STB OUT (Strobe Out), подтверждение ввода SACK IN (Acknowledge Input) и подтверждение вывода #АСК OUT (Acknowledge Out), использованы две линии порта PORTB и входы запросов на прерывания INTO и INT1.

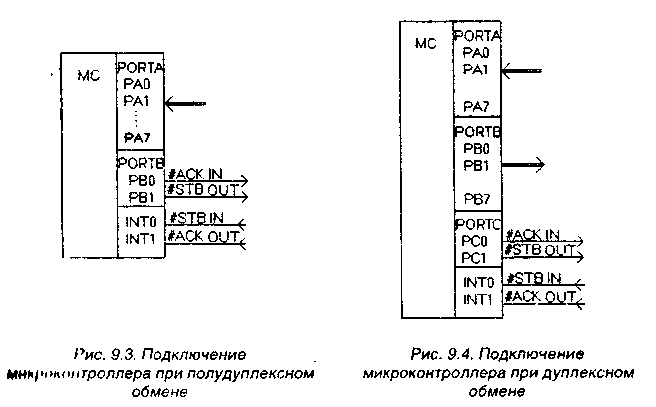
При поступлении сигнала #STB IN = 0 контроллер должен выставить сигнал подтверждения #АСК IN = 0 и осуществить запись во входной регистр порта PORTA.. При низком уровне сигнала ASK IN объекту запрещается формировать новый сигнал #STB !N. По окончании записи контроллер снимает сигнал #АСК IN, разрешая повторную передачу данных.

Работа микроконтроллера в режиме вывода аналогична. Микроконтроллер выводит данные в порт PORTA, подтверждая вывод сигналом #STB OUT = 0. Объект формирует сигнал подтверждения #АСК OUT = 0, сообщая контроллеру о готовности к приему данных. При снятии сигнала #АСК OUT контроллер должен снять сигнал #STB OUT и может вновь перейти к выводу данных.

Одновременная передача данных в двух направлениях при полудуплексном обмене невозможна

дуплексный обмен

Дуплексным считается двунаправленный синхронный обмен, при котором в любой момент времени возможна передача данных в двух направлениях. В этом случае для передачи данных в каждом направлении выделяется свой однонаправленный порт. На рис. И 4 изображен вариант соединения микроконтроллера ATmega 163 с объектом для обмена айнными в дуплексном режиме.

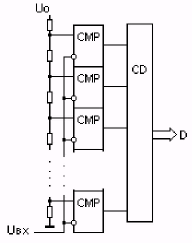


11 схеме на рис. 9.4 порт PORTA работает на ввод данных, порт PORTB - на вывод, | пинии I 'СО и РС1 порта PORTC использованы для вывода сигналов квитирования #АСК И #ЛСК OUT. Для ввода сигналов квитирования от объекта #STB IN и #АСК OUT использованы входы запросов на прерывания INTO и INT1. При таком подключении задача ввода данных в микроконтроллер получает больший приоритет, чем задача вывода.

**32. Аналого-цифровые преобразователи. Работа с АЦП микроконтроллера ATmega 163**

Аналого-цифровой преобразователь ADC (Analog Digital Converter) осуществляет преобразование напряжения в цифровой код. Он предназначен для оцифровки и ввода в микроконтроллер аналоговых сигналов с различных датчиков физических величин. Схемы преобразователей различны. В зависимости от типа построения меняются и свойства преобразователя.

Параллельный преобразователь



В параллельном преобразователе (рис. 1) входной сигнал подается сразу на множество компараторов, осуществляющих сравнение сигнала с опорным напряжением. Опорные напряжения формируются цепочкой резисторов, делящих эталонное напряжение U0 на равные части.

Рис. 1. Параллельный аналого-цифровой преобразователь

Такие схемы ADC работают очень быстро, но сложны и используются редко.

Преобразователь последовательного приближения

Основными элементами преобразователя (рис. 2) является регистр последовательных приближений, код из которого с помощью цифроаналогового преобразователя преобразуется в напряжение. Компаратор СМР сравнивает входное напряжение с выходным напряжением преобразователя и через устройство управления воздействует на регистр.

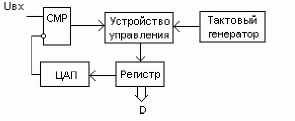


Рис. 2. Преобразователь последовательного приближения

Преобразование выполняется за несколько тактов. В первом такте в старший разряд регистра последовательных приближений записывается единица. Если в результате сравнения на выходе компаратора устанавливается единичный сигнал, единица в старшем разряде регистра сохраняется. В противном случае - сбрасывается. Далее, в том же порядке, формируется второй по старшинству разряд результата, потом - третий и т.д. Для получения результата необходимо n тактов, где число n равно разрядности преобразователя.

# Интегрирующий преобразователь

Интегрирующий ADC для сравнения входного сигнала с эталонным использует заряд конденсатора. Сначала (рис. 3) конденсатор в течении фиксированного промежутка времени Т1 заряжается током, пропорциональным входному сигналу. После это он разряжается постоянным током с определенным значением. Время разряда конденсатора Т2 пропорционально значению входного напряжения. Оно фиксируется с помощью счетчика и поступает на выход схемы (рис. 4).

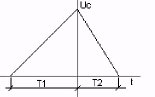


Рис. 3. Интегрирование сигнала в преобразователе

Интервал времени T1 задается включением ключа S1. По окончании T1 ключ S1 размыкается, a S2 - замыкается. Опорное напряжение U0 должно иметь знак противоположный знаку напряжения входного. Компаратор, устройство управления и счетчик определяют выходной код D, пропорциональный интервалу T2.

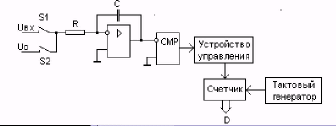
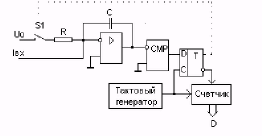


Рис. 10.4. Интегрирующий ADC

Интегрирующие схемы ADC имеют 8-16 разрядов и могут представлять результат в двоичном или двоично-десятичном коде.



Сигма-дельта преобразователь

Сигма-дельта преобразователи являются разновидностью интегрирующих ADC, в которых входной ток компенсируется коммутируемым зарядом от встроенного источника (рис. 5). Импульсы тока фиксированной длительности на каждом такте могут быть подключены к входу интегратора. В суммирующей точке интегратора поддерживается нулевой средний ток. Счетчик подсчитывает количество импульсов, поступающих в суммирующую точку за фиксированный период времени. Результат счета пропорционален входному напряжению. Рис. 5. Сигма-дельта преобразователь

**Управление АЦП микроконтроллера**

В состав микроконтроллеров обычно включают 8 - 16-битные многоканальные преобразователи с большим набором встроенных функций. При этом все функции преобразователя программируются и могут быть изменены в процессе работы.

Например, микроконтроллер ATmega163 оснащен 10-разрядным ADC последовательных приближений (рис. 10.6). ADC подсоединен к 10-канальному аналоговому мультиплексору (MUX), позволяющему подать на вход преобразователя любой из восьми входных сигналов со входов ADCO...ADC7, либо эталонное напряжение 1,22В. либо сигнал со входа AGND. Вывод AGND рекомендуется подсоединить к точке с нулевым потенциалом GND (Ground). ADC содержит схему выборки/хранения SHC (Sample&Hold Comparator),удерживающую напряжение входа во время преобразования на неизменном уровне.

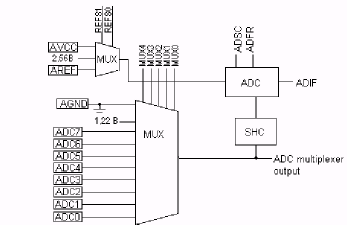


Рис. 6. Структура аналого-цифрового преобразователя

Аналого-цифровой преобразователь преобразует напряжение аналогового входного сигнала в 10-разрядное цифровое значение методом последовательных приближений. Минимальное значение входного напряжения равно напряжению на контакте AGND. максимальное значение не должно превышать напряжение на контакте AREF. Результат в виде 10-битного двоичного числа D равен:



где U-входное напряжение, a U0- опорное напряжение преобразователя.

В качестве источника опорного напряжения преобразователя можно использовать внешний сигнал с вывода AREF, внутренний источник 2.56В, либо напряжение питания аналоговой части микроконтроллера с вывода AVCC. Напряжение на выводе AVCC не должно отличаться от напряжения питания Vcc более чем на ±0,3 В.

Например, если аналоговый мультиплексор подключает ко входу ADC эталонное напряжение U =1,22B, а в качестве опорного напряжения использовать источник U0=2,56В, то результат преобразования:D=1,22\*1024/2,56=488=$1Е8=0b111101000.

**3. Общее понятие микропроцессора**

Микропроцессор - программно управляемое устройство для обработки цифровой информации и управления процессом этой обработки кристалл с элементами и программа.

Характеристики: Программный опрос, Обмен через прерывания, Обмен программным доступом к памяти.

Микропроцессорный комплект – совокупность микросхем и других интегральных схем, совместимых по архитектуре, конструктивному исполнению и электрическим параметрам, обеспечивающих возможность совместного применения

Архитектура – функциональные возможности аппаратных средств системы, используемые для представления программных данных и управления процессом вычислений.

Микропроцессоры — один из стремительно развивающихся и, безусловно, перспективных видов техники. Их отличие от обычных БИС состоит в том, что они содержат в своем составе управляющие элементы, позволяющие настроить эти БИС на выполнение любых операций, т. е. на реализацию любой зависимости между последовательностями входных и выходных сигналов. То обстоятельство, что БИС с перестраиваемой логикой способны при соответствующей «настройке» выполнять любые функции, делает их универсальными и полностью снимает противоречие между степенью интеграции и требуемым объемом производства.

Набор управляющих сигналов, настраивающий БИС на выполнение определенной функции, называется микрокомандой. Дальнейшее развитие программно - перестраиваемой логики привело к тому, что БИС стали выполнять не одну, а последовательность микрокоманд, т. е. алгоритм. Возникли БИС с микропрограммным управлением.

Одной из реализаций БИС с микропрограммным управлением и явились микропроцессоры. Всякий микропроцессор воплощает в себе главное свойство упомянутых БИС — на его основе можно построить систему, реализующую любое преобразование последовательности электрических сигналов. Если эти сигналы являются носителями информации, то, очевидно, что создаваемая микропроцессорная система способна перерабатывать любую информацию.

Итак, микропроцессор (МП) — это программно-управляемое устройство, осуществляющее процесс обработки информации и построенное на одной или нескольких больших интегральных схемах.

Первый микропроцессор появился в 1971 г

Возможны два способа решения поставленной задачи: аппаратный и программный.

Аппаратный способ имеет особенности:

для выполнения каждой операции используется индивидуальный операционный блок;

распределение переменных по входам и выходам операционного блока не изменяется в процессе реализации алгоритма;

порядок реализации алгоритма определяется схемой соединения операционных блоков;

число операционных блоков резко увеличивается с ростом сложности алгоритма.

Программный способ реализации алгоритма имеет по сравнению с аппаратным два основных преимущества:

во-первых, с усложнением алгоритма объем оборудования увеличивается незначительно;

во-вторых, путем изменения программы можно на одном оборудовании решать различные задачи.

Микропроцессор представляет собой оптимальное объединение аппаратных и программных средств для решения конкретной задачи.

Целесообразность применения МП в разрабатываемых системах:

Сложные алгоритмы;

Обработка больших массивов информации;

Частое обращение к ЗУ;

Число интегральных схем более 30;

Микропроцессоры могут быть классифицированы по ряду признаков.

По типу архитектуры различаются однокристальные и многокристальные секционные микропроцессоры.

Однокристальные микропроцессоры (КР580, КР581, К588, К1801, КА1808, КМ1810, КН1811) получаются при реализации всех аппаратных средств процессора в виде одной БИС. По мере увеличения степени интеграции элементов в кристалле и числа выводов корпуса параметры однокристальных микропроцессоров улучшаются. Как правило, для расширения функциональных возможностей однокристального микропроцессора его дополняют другими типами микросхем. Однако возможности однокристальных микропроцессоров ограничены аппаратурными ресурсами кристалла и корпуса. Поэтому более распространены многокристальные секционные микропроцессоры.

Многокристальные секционные микропроцессоры (К583, К584, КР587, К589, К1800, КР1802, КМ 1804) получаются в том случае, когда в виде БИС реализуются части (секции) логической структуры процессора. Микропроцессорная секция — это БИС, предназначенная для обработки нескольких разрядов данных или выполнения определенных управляющих операций. Секционность БИС микропроцессора определяет возможность наращивания разрядности обрабатываемых данных или усложнения устройств управления микропроцессором при «параллельном» включении большего числа БИС. Многокристальные секционные микропроцессоры имеют разрядность от 2 ... 4 до 8 ... 16 бит и позволяют создавать высокопроизводительные процессоры ЭВМ.

Микропроцессоры можно классифицировать и по технологии изготовления. Ниже приведены практически все современные технологии:

р - МДП - технология (К1814);

я - МДП - технология (КР580, КР581, К1801, К1809, КМ1810, КН1811, КМ1813, КР1816);

ТТЛДШ - технология (К589, КР1802, КМ1804);

КМДП - технология (К586, К588...);

И2Л-технология (К583, К584, КА1808);

ЭСЛ - технология (К1800).

По назначению различаются универсальные и специализированные микропроцессоры.

Универсальные микропроцессоры можно применять для решения разнообразных задач. Их производительность мало зависит от проблемной специфики решаемых задач.

Специализация МП, т. е. его проблемная ориентация на ускоренное выполнение определенных функций, позволяет резко увеличить производительность при решении только определенных задач.

По виду обрабатываемых входных сигналов микропроцессоры подразделяются на цифровые (1 или 0) и аналоговые.

Сами микропроцессоры — это цифровые устройства, однако могут иметь встроенные аналого-цифровые и цифроаналоговые преобразователи. Поэтому входные аналоговые сигналы передаются в МП через преобразователь в цифровой форме, обрабатываются и после обратного преобразования в аналоговую форму поступают на выход.

По разрядности данных, обрабатываемых микропроцессором, выделим:

двухразрядные (К589);

четырехразрядные (К584, КР587, К1800, КМ1804, К1814);

восьмиразрядные (КР580, К583, КР1802, КМ1813, КР1816);

шестнадцатиразрядные (КР581, К588, К1801, К1809, КМ1810, КН1811).

По характеру временной организации работы различаются синхронные и асинхронные микропроцессоры.

Синхронные микропроцессоры — это микропроцессоры, в которых начало и конец выполнения операций задаются устройством управления (время выполнения операций в этом случае не зависит от вида выполняемых команд и значений операндов).

Асинхронные микропроцессоры позволяют начало каждой следующей операции определить по сигналу фактического окончания выполнения предыдущей операции.

По числу выполняемых программ микропроцессоры делятся на одно- и многопрограммные.

В однопрограммных микропроцессорах выполняется только одна программа. Переход к выполнению другой программы происходит после завершения текущей программы.

В много- или мультипрограммных микропроцессорах одновременно выполняется несколько (обычно несколько десятков) программ. Организация мультипрограммной работы микропроцессорных управляющих систем позволяет осуществлять контроль за их состоянием и управлять большим числом источников или приемников информации.

**2. Логическая структура микропроцессорной системы**

Для выполнения своих функций микропроцессор должен иметь регистры, сумматоры, логические схемы, дешифраторы и другие элементы, которые соединены между собой с помощью шин — соединительных линий. Схема, которая показывает, из каких блоков состоит микропроцессор, и связь между этими блоками, называется структурной схемой микропроцессора.

Функции, реализуемые микропроцессором, определяются не только его структурой, но и последовательностью управляющих слов (команд). При изменении этой последовательности изменяется и выполняемая микропроцессором функция. Для комплексной характеристики возможностей микропроцессора пользуются понятием «архитектура микропроцессора».

Архитектура микропроцессора — это его логическая организация, определяемая возможностями микропроцессора по аппаратной или программной реализации функций, необходимых для построения микроЭВМ.

Архитектура микропроцессора отражает:

а) структуру микропроцессора;

б) способы обращения ко всем элементам структуры;

в) способы представления и форматы данных;

г) набор операций, выполняемых микропроцессором;

д) способы указания адреса данных;

е) форматы управляющих слов, поступающих извне;

ж) характеристики управляющих сигналов;

з) реакцию микропроцессора на внешние сигналы.

Зная архитектуру микропроцессора и всей микропроцессорной системы, можно составить определенную последовательность команд, называемую программой, которая обеспечит выполнение заданной функции.

Обобщенную схему микропроцессора можно представить в виде, приведенном на рис. 7.2. В этой структуре используется трехшинная организация работы. Шиной называют группу линий передачи информации, объединенных общим функциональным признаком. Приняты такие условные обозначения:

БА,

БД — буферные регистры адресов и данных;

БП — блок прерывания;

РОН — регистры общего назначения;

ИР — индексные регистры;

СТЕК — блок регистров;

УС — указатель стека;

ПС - программный счетчик;

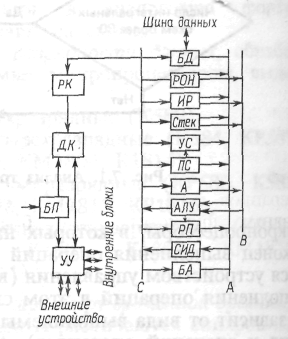
А — аккумулятор;

АЛУ — арифметическо - логическое устройство;

РП — регистр признаков;

СИД — схема «инкремент-декремент»;

УУ — устройство управления;



РК — регистр команд;

ДК — дешифратор команд.

Конкретные микропроцессоры, как правило, не содержат всех узлов, показанных на рис. В этих случаях соответствующие функции могут выполняться программно, а в качестве некоторых специальных регистров могут использоваться РОН или ячейки памяти. В ряде микропроцессорных комплексов отдельные функциональные узлы могут выполняться в виде отдельных БИС.

Для управления преобразователем в микроконтроллере используются регистры:

Регистр управления мультиплексором ADMUX (ADC Multiplexer Selection Register);

Регистр управления аналого-цифровым преобразователем ADCSR (ADC Control and Status Register);

Регистры данных ADCL и ADCH (ADC Low и ADC High);

Регистр состояния микроконтроллера SREG (Status Register).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Биты | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADMUX $07($27) | REFS1 | REFS0 | ADLAR | MUX4 | MUX3 | MUX2 | MUX1 | MUX0 |
| ADCSR $06($26) | ADEN | ADSC | ADFR | ADIF | ADIE | ADPS2 | ADPS1 | ADPS0 |
| ADCH $05($25) | SIGN |  |  |  |  |  | ADC9 | ADC8 |
| ADCL $04($24) | ADC7 | ADC6 | ADC5 | ADC4 | ADC3 | ADC2 | ADC1 | ADC0 |
| SREG $3F($5F) | I |  |  |  |  |  |  |  |

Рис. 7. Регистры, используемые АЦП

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

Работа аналого-цифрового преобразователя разрешается установкой в состояние 1 бита ADEN в регистре ADCSR. Преобразование начинается с установки в состояние 1 бита начала преобразования ADSC (ADC Start Conversion). Если в процессе выполнения преобразования производится смена канала данных, то ADC вначале закончит текущее преобразование, а потом выполнит переход к другому каналу.

Поскольку аналого-цифровой преобразователь формирует 10-разрядный результат, то по завершении преобразования результирующие данные размещаются в двух регистрах данных 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. Если регистр ADCL считан, обращение аналого-цифрового преобразователя 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 аналого-цифрового преобразователя к регистрам ADCH и ADCL разрешается по завершении считывания содержимого регистра ADCH.

Аналого-цифровой преобразователь имеет свое собственное прерывание ADC (вектор $1С), которое может быть активизировано по завершению преобразования. Когда обращение к регистрам запрещено, в процессе считывания регистров ADCL и ADCH, прерывание будет активизироваться, даже при потере результата.

Регистр ADMUX (ADC Multiplexer Selection Register) предназначен для управления входным аналоговым мультиплексором

Биты 7 и 6 - REFS1..0 (Reference Selection Bits) - обеспечивают выбор эталонного напряжения на входе AREF аналого-цифрового преобразователя. Выбор производится в соответствии с таблицей 1. Изменение этих битов во время процесса преобразования приводит к ошибке. Для ее исключения пользователь должен игнорировать первый результат после изменения битов. Внутренние источники напряжения не могут быть использованы, если к контакту AREF приложено внешнее напряжение.

Таблица 1. Выбор источника опорного напряжения АЦП

|  |  |  |
| --- | --- | --- |
| REFS1 | REFS0 | Выбор источника напряжения |
| 0 | 0 | AREF, внутреннее напряжение Vref отключено |
| 0 | 1 | AVCC с внешним конденсатором на контакте AREF |
| 1 | 0 | Резерв |

Бит 5 - ADLAR (ADC Left Adjust Result) - воздействует на запись результата в регистры данных ADCL и ADCH. При ADLAR=0 можно использовать упрощенное 8-битное преобразование.

Биты 4..О - MUX4..MUXO (Multiplexer bits) - предназначены для выбора входа, коммутируемого на вход преобразователя. Выбор осуществляется в соответствии с таблицей 2. Изменение этих битов в процессе преобразования, когда флаг ADIF в регистре ADCSR установлен, не приводит к изменению результата.

Таблица 2. Выход входного сигнала ADC

|  |  |
| --- | --- |
| MUX4..0 | Подключаемый контакт |
| 00000 | ADCO |
| 00001 | ADC1 |
| 00010 | ADC2 |
| 00011 | ADC3 |
| 00100 | ADC4 |
| 00101 | ADC5 |
| 00110 | ADC6 |
| 00111 | ADC7 |
| 01000. .11101 | Резерв |
| 11110 | 1.22V |
| 11111 | 0V (AGND) |

Регистр - ADCSR (ADC Control and Status Register) предназначен для управления работой аналого-цифрового преобразователя.

Бит 7 - ADEN (ADC Enable) - разрешение работы ADC. Очистка бита запрещает работу ADC. Запрещение ADC в процессе преобразования прекращает преобразование.

Бит 6 - ADSC (ADC Start Conversion) - 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 или одновременно с разрешением, будет выполняться пустое преобразование. Это пустое преобразование активизирует преобразователь. ADSC будет сохранять состояние 1 в течение всего цикла преобразования и сбрасывается по его завершению. При выполнении пустого преобразования, предшествующего активизируемому, бит ADSC остается установленным до завершения активируемого преобразования. Запись 0 в этот бит эффекта не оказывает.

Бит 5 - ADFR (ADC Free Run Select) - установка циклического режима работы ADC. При установленном в состояние 1 бите ADFR аналого-цифровой преобразователь будет работать в циклическом режиме. В этом режиме производятся выборки и обращения к регистрам непрерывно (одно за другим). Очистка бита приводит к прекращению циклического режима.

Бит 4 - ADIF (ADC Interrupt Flag) - флаг прерывания ADC. Данный бит устанавливается в состояние 1 по завершению преобразования и обновления регистров данных. Прерывание по завершению преобразования ADC выполняется, если в состояние 1 установлены бит ADIE и I - бит регистра статуса SREG. Бит ADIF сбрасывается аппаратно при выполнении подпрограммы обработки соответствующего вектора прерывания. Кроме того, бит ADIF может быть очищен записью во флаг логической 1. Этого необходимо остерегаться при чтении-модификации-записи ADCSR, поскольку может быть запрещено отложенное прерывание. Это применимо и в случаях использования команд sbi и cbi.

Бит 3 - ADIE (ADC Interrupt Enable) - разрешение прерывания ADC. При установленных в состояние 1 бите ADIE и l-бите регистра SREG активируется прерывание с вектором $1С по завершению преобразования ADC.

Биты 2..0 - ADPS2..ADPSO (ADC Prescaler Select Bits) - выбор коэффициента предварительного деления. Данные биты определяют коэффициент деления тактовой частоты микроконтроллера для получения необходимой тактовой частоты ADC.

Таблица 3. Выход входного сигнала ADC

|  |  |  |  |
| --- | --- | --- | --- |
| ADPS2 | ADPS1 | ADPSO | Коэффициент деления |
| 0 | 0 | 0 | Без деления |
| 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 |

Регистры ADCL и ADCH являются регистрами данных. Их содержимое зависит от состояния бита ADLAR регистра ADMUX.

Когда преобразование выполнено, десятибитный результат находится в этих двух регистрах. Если младший регистр ADCL считан, то регистры не изменяются до чтения старшего регистра ADCH. ADLAR бит в ADMUX воздействует на представление результата.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Биты | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ADCH $05($25) | ADC9 | ADC8 | ADC7 | ADC6 | ADC5 | ADC4 | ADC3 | ADC2 |
| ADCL $04($24) | ADC1 | ADC0 |  |  |  |  |  |  |