1. Процессор. Назначение, функции, основные блоки. Регистры процессора, классификация. Основные характеристики процессора.

Микропроцессор (Процессор) - ядро любой микропроцессорной системы, т.к. именно он производит всю обработку информации в ней. Микропроцессоры то же что и процессоры только маленькие. Микропроцессор – это микроэлектронное программируемое устройство, предназначенное для обработки информации и управления процессами обмена этой информацией в составе микропроцессорной системы (компьютера).

Основные функции процессора:

1. Дешифрация и выполнение команд программы;
2. Организация обращения к оперативной памяти;
3. Организация обращения к устройствам ввода-вывода (УВВ);
4. Прием и обработка запросов от устройств вычислительной системы (ВС) и внешней среды (запросов на прерывание).

Обязательными компонентами микропроцессора являются регистры, арифметико-логическое устройство (АЛУ) и блок управления.

Структура процессора показана на рис.4.

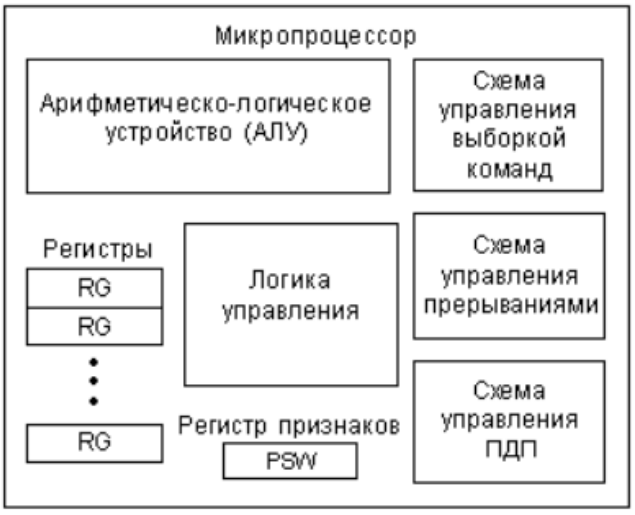


Рис. 4

Блок управления отвечает за последовательное выполнение команд программы, правильное перенаправление потоков данных и взаимодействие узлов процессора. Содержит следующие элементы:

1. Схема управления выборкой команд (выборка команд из памяти и их дешифрация);

2. Схема управления прерываниями (обрабатывает поступающий на процессор запрос прерывания, определяет адрес начала программы обработки прерывания, сохраняет в памяти текущее состояние регистров процессора и обеспечивает переход к программе обработки прерывания);

3. Схема управления прямым доступом к памяти (ПДП, DMA – Direct Memory Access) (отключает процессор от внешних шин на время предоставления прямого доступа к памяти запросившему его устройству);

4. Логика управления (Организует взаимодействие узлов процессора, синхронизирует работу процессора с внешними сигналами и реализует обмен данными между процессором и памятью или устройствами ввода-вывода);

Арифметико-логическое устройство. Отвечает за выполнения арифметических и логических операций (т. е. для обработки данных)в соответствии с командой, дешифрированной блоком управления.

Регистры предназначены для временного хранения данных, адресов и специальных кодов в целях сокращения количества обращений к медленной ОП.

Классификация регистров:

1. Функциональные регистры

а) Регистры общего назначения (РОН) (запоминают операнды и результаты операций);

б) Специальные регистры (Только для вычисления адресов. Счетчик команд это спец. регистр);

в) Индексные (помнят номера элементов массивов);

г) Регистры-указатели (указывают на начало некоторой области памяти. Пример: указатель стека);

д) Регистр флагов (его биты содержат информацию о результате предыдущей команды);

2. Системные регистры

а) Управляющие регистры (Задают режимы работы процессора, кэш-памяти и т.д.);

б) Регистры системных адресов (помнят специальные адреса, например, адреса таблиц дескрипторов при виртуальной организации памяти);

3. Вспомогательные регистры

а) Теневые регистры (используются при виртуальной организации памяти для хранения дескрипторов с целью сокращения числа обращений к памяти);

б) Регистры отладки и тестирования;

в) Буферные регистры;

г) Стеки

Основные характеристики процессора:

1. Тактовая частота (зачастую процессор в разы быстрее остальных элементов системы, поэтому используется внешняя и внутренняя тактовые частоты. Внешняя - частота системной шины, для синхронизации обмена данных между процессором и другими устройствами. Внутренняя - частота синхронизации самого процессора);

2. Разрядность данных (Тоже есть внешняя и внутренняя. Внешняя - число бит, которое за один раз можно передать между процессором и другими устройствами. Внутренняя - число бит, которое процессор может обрабатывать одновременно);

3. Адресное пространство (совокупность ячеек памяти ПЗУ, ОЗУ и устр. I/O. Максимальный объём = 2^n, где n - разрядность шины адреса);

4. Система команд и способы адресации (все команды, выполняемые процессором);

5. Система прерываний;

2. Классификация микропроцессоров по архитектуре, по области применения, по типу управления, по производительности и стоимости.

Классификация микропроцессоров:

1. По архитектуре (архитектура – функциональная организация, в которую закладываются основные идеи и принципы для достижения наивысшей производительности)

1.1. По степени полноты системы команд:

а) CISC (Complex Instruction Set Computer);

б) RISC (Reduced Instruction Set Computer).

1.2. По общему структурному составу:

а) Скалярные процессоры (в АЛУ одновременно за одну команду могут обрабатываться 1-2 операнда);

б) Векторные процессоры (По одной команде обрабатывается множество операндов, которые называют вектором или матрицей операндов);

в) Суперскалярные процессоры (Строятся на базе нескольких параллельно работающих АЛУ. Ресурсы распределяются динамически);

г) VLIW (Very Long Instruction Word, очень длинное командное слово) (Тот же суперскалярный, но ресурсы процессора распределяются компилятором, который группирует команды в длинные командные слова со многими кодами операций).

2. По областям применения:

а) Универсальные процессоры (для решения широкого круга задач);

б) Микроконтроллеры (для построения устройств управления различной аппаратурой);

в) Цифровые процессоры обработки сигналов (для цифровой обработки аналоговых сигналов);

г) Медийные процессоры (для преобразования графической и звуковой информации);

3. По типу управления

а) Асинхронные (устройство управления рассылает по блокам процессора приказы и принимает от них ответные сигналы об окончании заданной работы);

б) Синхронные (Устройство управления рассылает по блокам приказы и не ожидает ответных сигналов. Блоки выполняют заданные действия в течение фиксированного, заранее известного интервала времени);

4. По производительности и стоимости

а) Низкая (для встроенных систем);

б) Средняя (для ПК);

в) Высокая (для рабочих станций, серверов, суперкомпьютеров).

3. Использование микропроцессоров в системах управления. Структурная схема системы на базе микропроцессора.

(Сделал полностью в GPT тк нет информации нигде)

Микропроцессоры широко используются в системах управления благодаря своей высокой производительности и гибкости. Они могут обрабатывать большой объем данных и выполнять сложные алгоритмы управления в режиме реального времени. Примеры применения микропроцессоров в системах управления:

1. Автоматические системы контроля и управления производственными процессами, такие как управление температурой, давлением, скоростью и т.д.
2. Системы управления транспортными средствами, такие как автоматические тормозные системы, системы контроля стабильности и т.д.
3. Системы управления энергетическими процессами, такие как управление энергоснабжением, солнечными батареями и т.д.
4. Системы управления домашними устройствами, такие как умный дом, умный кондиционер и т.д.
5. Системы управления роботами и автоматическими машинами, такие как промышленные роботы, автоматические станки и т.д.

Микропроцессоры обеспечивают точность и надежность в системах управления, что позволяет снизить затраты на обслуживание и повысить эффективность работы системы

Структурная схема системы на базе микропроцессора обычно включает в себя следующие блоки:

1. Микропроцессор (CPU) - является сердцем системы и отвечает за обработку данных и выполнение команд.
2. Память (RAM, ROM, EEPROM) - используется для хранения программного кода и данных.
3. Внешние устройства ввода-вывода (клавиатура, мышь, дисплей, принтер, датчики и т.д.) - используются для взаимодействия с пользователем и получения данных из внешней среды.
4. Контроллеры периферийных устройств (DMA, UART, SPI и т.д.) - управляют передачей данных между микропроцессором и внешними устройствами ввода-вывода.
5. Интерфейсы для связи с другими устройствами (Ethernet, USB, RS-232 и т.д.) - используются для обмена данными между системой и другими устройствами.
6. Часы реального времени (RTC) - используются для хранения и управления временем и датой в системе.
7. Питание и управление энергопотреблением - управляют питанием системы и устройствами ввода-вывода для уменьшения энергопотребления.
8. Интерфейсная шина для связи между блоками системы (например, шина данных и адресов).
9. Система прерываний - используется для управления операциями системы и взаимодействия с внешними устройствами.
10. ПО системы - программный код для управления системой и взаимодействия с пользователем.

Структурная схема системы на базе микропроцессора может отличаться в зависимости от конкретного устройства и его функциональности.

4. Микроконтроллеры. Разрядность микроконтроллеров. Основные семейства 8- разрядных микроконтроллеров. Архитектуры – гарвардская и фон-неймановская; открытая и закрытая; RISC и CISC.

МК можно классифицировать

1) по разрядности данных:

- 8-ми разрядные МК (наиболее многочисленная группа. Это простые и дешевые МК с относительно низкой производительностью)

- 16-ти разрядные МК (модификация 8-ми разрядных. Характеризуются расширенной системой команд и способов адресации, увеличенным набором регистров и объемом адресуемой памяти)

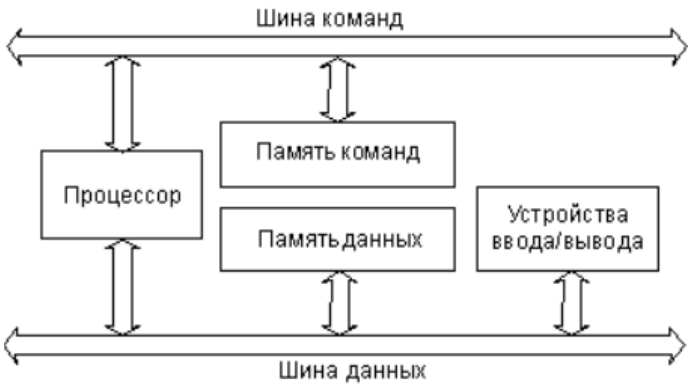
- 32-х разрядные МК (есть высокопроизводительный процессор, почти такой как у процессоров общего назначений)

2) По шинной организации:

- архитектура с общей, единой шиной для данных и команд (одношинная, или принстонская, фон-неймановская архитектура).

Достоинство: наличие единой памяти данных и команд позволяет гибко распределять ее объем между кодами данных и команд.

- альтернативный тип архитектуры микропроцессорной системы — это архитектура с раздельными шинами данных и команд (двухшинная, или гарвардская, архитектура). Обмен процессора с каждым из двух типов памяти происходит по своей шине. **Достоинство**: возможность одновременной выборки команд и данных.



3) По архитектуре

а) CISC (До конца 70-х годов прошлого века полагалось, что производительность процессора тем выше, чем мощнее его система команд. В результате процессоры CISC развивались за счет расширения микрокода и усложнения операций)

Особенности CISC-процессоров:

1. Поддержка большого количества команд и способов адресации;
2. Реализация сложных команд на аппаратном уровне;
3. Различные команды имеют различную длительность исполнения;
4. Многие команды могут обращаться к ОП;
5. Небольшой объем регистровой памяти (8 – 16 регистров общего назначения).

Достоинства:

- выполнение сложных операций одной командой;
- возможность выполнять обработку данных непосредственно в ОП.

Недостатки:

- переменная длительность исполнения различных команд требует построения сложных асинхронных конвейеров;
- исполнение многих команд требует обращения к медленной ОП;
- малый объем регистровой памяти приводит к частым обращениям к ОП.

б) RISC (В процессорах с RISC-архитектурой реализована идея о том, что производительность

процессора напрямую зависит от простоты команд, которая возникла в 80-х годах прошлого века. В RISC-архитектуре аппаратно реализовали 20% (от команд CISC) самых используемых команд, а остальные 80% вынесли на программный уровень.)

Особенности RISC-процессоров:

1. Все команды разделены на 2 группы:
   - команды обмена с памятью;
   - команды обработки данных (Команды обработки данных работают только с регистрами. Вначале данные должны быть записаны в регистры с помощью команд первого типа, и только затем над ними выполняется операция с помощью команд второго типа.).
2. Команды имеют фиксированный формат и длительность исполнения;
3. Большой объем регистровой памяти (десятки и сотни РОН).

Достоинства:

- фиксированная длительность исполнения команд позволяет строить более простые синхронные конвейеры;
- сокращение количества обращений к памяти снижает требования к ее пропускной способности;
- более короткий цикл разработки процессора, чем у CISC;
- простота RISC-процессора позволяет высвободить площадь кристалла для реализации дополнительных структур, увеличивающих производительность (больше РОН, кэш и т.д.)

Недостатки:

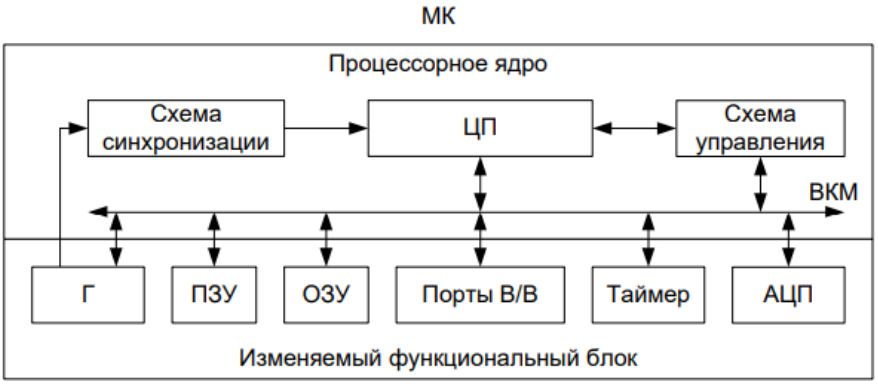
- программа для RISC-процессора содержит большее количество команд, чем аналогичная программа для CISC процессора.

Закрытая архитектура – отсутствие линий ША и ШД на выводах корпуса МК.

Открытая архитектура - наоборот.

5. Обобщенная структурная схема микроконтроллера. Основные блоки. Память программ и данных. Виды ПЗУ, применяемые для памяти программ.

Модульный принцип построения МК: МК одного семейства содержат базовый функциональный блок (Процессорное ядро), одинаковый для всех МК данного семейства, и изменяемый функциональный блок, который отличается в МК разных моделей в пределах семейства.



Процессорное ядро включает:

- ЦП;
- внутренние магистрали адреса, данных и управления (внутриконтроллерные магистрали, ВКМ);
- схему формирования импульсной последовательности для тактирования ЦП и межмодульных магистралей;
- устройство управления режимами работы МК: состояние начального запуска (сброс), активный режим, в котором МК выполняет прикладную программу, режимы пониженного энергопотребления.

Изменяемый функциональный блок включает:

- модули различных типов памяти;
- модули генераторов синхронизации (Г);
- модули периферийных устройств (таймеры, параллельные порты ввода/вывода, АЦП, ЦАП, контроллеры ЖК-индикаторов и др.)

В простых МК модуль обработки прерываний входит в состав процессорного ядра. В более сложных МК он представляет собой отдельный модуль.

В МК используется три основных вида памяти.

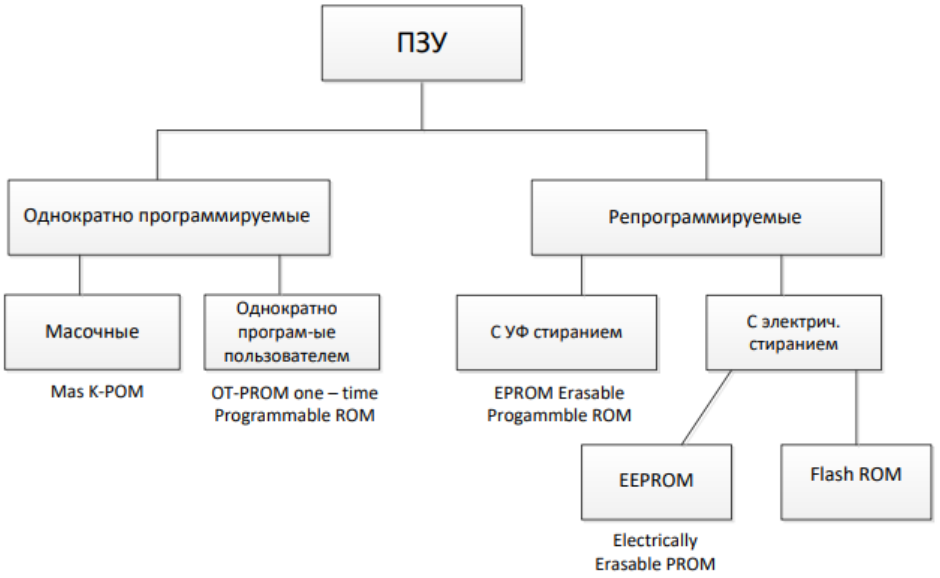
- память программ – ПЗУ, предназначена для хранения программного кода (команд) и констант. Ее содержимое в ходе выполнения программы не изменяется;

- память данных – ОЗУ, предназначена для хранения переменных в процессе выполнения программы;

- регистры МК – внутренние регистры ЦП и регистры управления периферийными устройствами (регистры специальных функций).

Память программ

Все ПЗУ представленные ниже являются энергонезависимыми.



1. ПЗУ масочного типа (mask-ROM).

Информация заносится в память при ее изготовлении и не может быть впоследствии изменена.

Достоинство: высокая надежность хранения информации, дешевизна при больших объемах производства.

Недостаток: изменение программы требует новой серии МК. Выпуск памяти такого типа целесообразен только в случае массового производства.

2. Однократно программируемые пользователем ПЗУ – OTPROM (One Time Programmable ROM).

В незапрограммированном состоянии каждая ячейка памяти при считывании возвращает код $FF. Программируются те разряды, которые должны

содержать 0. После установки в 0 разрядов ячейки невозможно восстановить их единичное значение. Это возможно за счет многократного приложения импульсов повышенного напряжения к битам ячейки памяти.

3. Репрограммируемые ПЗУ с ультрафиолетовым стиранием – EPROM (Erasable Programmable ROM).

Перед каждым сеансом программирования для восстановления единичных значений весь модуль подвергается стиранию при помощи ультрафиолетового облучения. Для этого корпус МК выполнен со специальным стеклянным окном. Число сеансов стирания/программирования составляет 25-100 раз. МК с таким ПЗУ стоят дорого.

4. Репрограммируемые ПЗУ с электрическим стиранием – EEPROM (Electrically Erasable Programmable ROM).

Стирание ячеек памяти производится электрическими сигналами. Дешевле, чем EPROM, но дороже, чем OTPROM. Максимальное число циклов стирания/программирования равно 10000.

Достоинство: возможность стирать и программировать МК, не снимая его с платы, что позволяет производить отладку и модернизацию ПО.

Используются редко, т.к. Flash-ПЗУ дешевле и имеют сходные характеристики

5. ПЗУ с электрическим стиранием типа Flash – Flash-ROM.

Отличается от EEPROM способом стирания информации. В EEPROM стирание производится отдельно для каждой ячейки, а во Flash-памяти стирать можно только блоками или страницами. Страница составляет 8, 16 или 32 байта. Стоимость не намного выше, чем OTPROM.

В современных МК используется два типа ПЗУ: Flash-ROM – для хранения программ, EEPROM – для хранения перепрограммируемых констант (настроек пользователя). Пример: запоминание каналов в музыкальных центрах и телевизорах. Число циклов стирания/программирования Flash-ROM равно 105 .

Наиболее совершенные МК содержат: ПЗУ программ типа Flash-ROM, масочное ПЗУ монитора связи, EEPROM ПЗУ для хранения изменяемых констант и ОЗУ данных.

Память данных

Как правило, выполняется на основе статического ОЗУ. Содержимое ячеек ОЗУ сохраняется при снижении тактовой частоты МК до сколь угодно малых значений, что позволяет снизить энергопотребление.

Информация сохраняется в памяти данных при напряжении питания не ниже напряжения хранения информации – U_STANDBY (обычно около 1 В). При снижении напряжения ниже минимально допустимого уровня U_DDMIN, но выше уровня U_STANDBY выполнение программы МК прекращается, но информация в ОЗУ сохраняется. При восстановлении напряжения питания происходит сброс МК, и выполнение программы продолжается без потери данных.

Объем памяти данных обычно составляет десятки и сотни байт, поэтому в некоторых случаях возникает необходимость подключения дополнительной внешней памяти (как памяти программ, так и данных).

6. Порты ввода-вывода микроконтроллеров. Виды портов; структурная схема одного разряда параллельного порта с альтернативной функцией.

Параллельные I/O порты

Каждый МК имеет некоторое количество линий ввода/вывода, которые объединены в многоразрядные (чаще 8-разрядные) параллельные порты ввода/вывода. В памяти МК каждому порту ввода/вывода соответствует свой адрес регистра данных. Обращение к регистру данных порта ввода/вывода производится теми же командами, что и обращение к памяти данных.

В зависимости от реализуемых функций различают следующие типы параллельных портов:

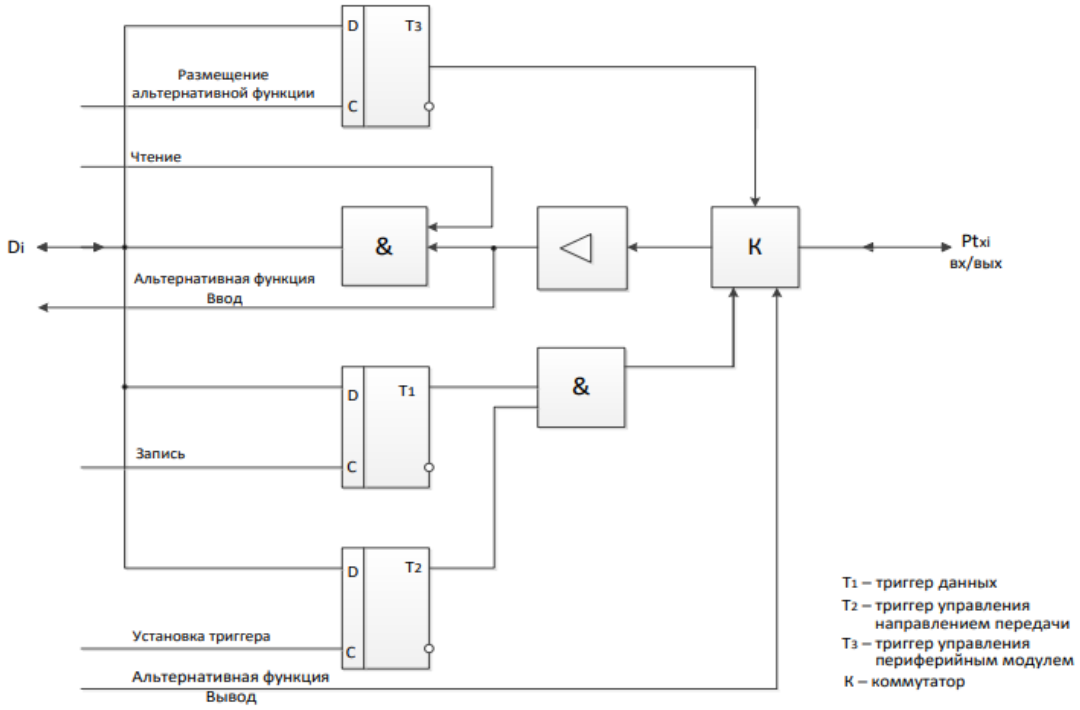
- однонаправленные порты, предназначенные только для ввода или только для вывода информации;

- двунаправленные порты, направление передачи которых (ввод или вывод) определяется в процессе инициализации МК;

- порты с альтернативной функцией (мультиплексированные порты). Отдельные линии этих портов используются совместно со встроенными периферийными устройствами МК, такими как таймеры, АЦП, контроллеры последовательных интерфейсов;

- порты с программно управляемой схемотехникой входного/выходного буфера.

Схема управления одним разрядом параллельного порта двунаправленного ввода вывода с альтернативной функцией



В режимах записи и чтения триггер Т3 должен быть установлен в «0», тем самым коммутатор подключает порт Ptxi к линии входа/ выхода микроконтроллера Di

Триггер управления Т2 разрешает вывод данных на внешний вывод.

Если Т2 находится в нуле, то разрешается ввод данных. При этом считывается значение сигнала, поступающее с входа/ выхода Ptxi по сигналу «чтения» на линию Di. (а не содержимое триггера Т1).

Если T2 в единице, то контакт Ptxi порта применяется для вывода данных и по сигналу «запись» на выход подается предварительно записанный бит из триггера T1.
В современных МК, как правило, обеспечивается индивидуальный доступ к триггерам данных и управления, что позволяет использовать каждую линию порта независимо в режиме ввода или вывода путем установки триггера T2.
Для разрешения обмена периферии микроконтроллера с внешним объектом триггер T3 необходимо установить в «1». При этом коммутатор подключает порт Ptxi к линиям входа/ выхода периферийного устройства.
Последовательные I/O порты
Задачи, которые решаются средствами модуля контроллера последовательного ввода/вывода, можно разделить на три основные группы:
1. Связь встроенной микроконтроллерной системы с системой управления верхнего уровня, например, с персональным компьютером. Чаще всего для этой цели используются интерфейсы RS-232C и RS-485;
2. Связь с внешними по отношению к МК периферийными ИС, а также с датчиками физических величин с последовательным выходом. Для этих целей используются интерфейсы I2C (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), а также нестандартные протоколы обмена;
3. Интерфейс связи с локальной сетью в мультимикроконтроллерных системах. В системах с числом МК до пяти обычно используются сети на основе интерфейсов I2C, RS-232C и RS-485 с собственными сетевыми протоколами высокого уровня. В более сложных системах все более популярным становится протокол CAN (Controller Area Network).
С точки зрения организации обмена информацией упомянутые типы интерфейсов последовательной связи отличаются:
1) режимом передачи данных (синхронный или асинхронный)

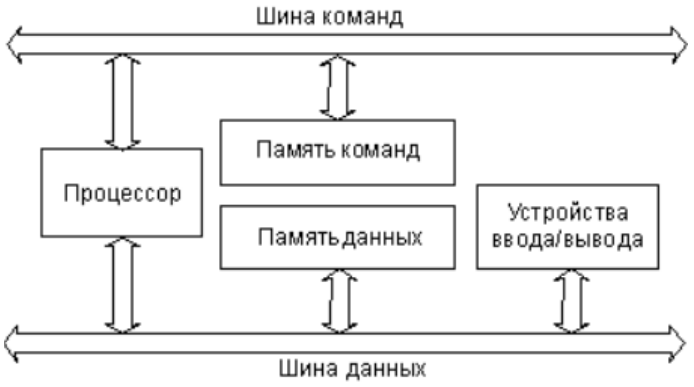
2) форматом кадра (число бит в посылке при передаче байта полезной информации)
3) временными диаграммами сигналов на линиях (уровни сигналов и положение фронтов при переключениях)
4) режимом обмена информацией: симплексный (от simplex – простой), дуплексный, полудуплексный (от duplex – двойной);
5) числом линий, по которым происходит передача в последовательном коде, обычно равно двум (I2C, RS-232C, RS-485) или трем (SPI).

Модули контроллеров последовательных интерфейсов реализуют на аппаратном уровне логику протоколов обмена заданных интерфейсов. Одноименные модули для различных МК даже одного семейства могут иметь отличия с точки зрения аппаратной реализации, но протоколы обмена должны обеспечиваться в соответствии с выбранным стандартом.

короткие вопросы

1. Какая архитектура процессора называется гарвардской?

Это архитектура с раздельными шинами данных и команд



(Обмен процессора с каждым из двух типов памяти происходит по своей шине. **Достоинство**: возможность одновременной выборки команд и данных.)

2. Какая архитектура процессора называется фон-неймановской?

Это архитектура с общей, единой шиной для данных и команд

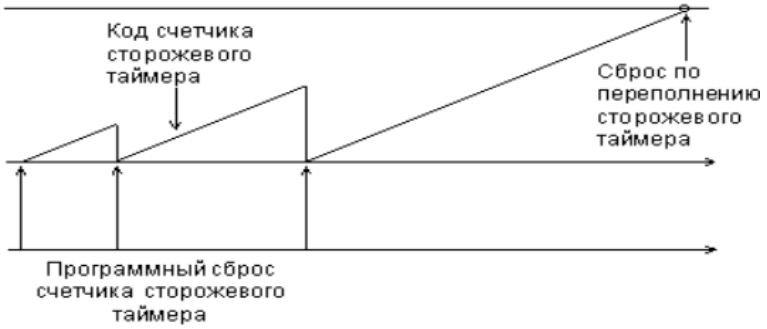
(**Достоинство**: наличие единой памяти данных и команд позволяет гибко распределять ее объем между кодами данных и команд.)

3. Для чего в состав микроконтроллеров входит счетчик-таймер?

Для эффективного решения задач отсчёта равных интервалов времени, измерения длительности сигнала, подсчёта числа импульсов на временном интервале, формирования на линии вывода сигнала с задержкой или определенной частотой.

4. Для чего нужен такое сторожевой таймер?

На случай выхода из состояния на котором завис МК.



(Он сбрасывается устройством через равные промежутки времени, а если в течение некоторого периода времени сброса не было, то таймер делает вывод, что система зависла, и перегружает ее.)

7. **Таймеры в микроконтроллерах. Основные функции; режимы работы. Функциональная схема простого счетчика-таймера. Измерение временного интервала; временные диаграммы. Недостатки простого таймера.**

Для эффективного управления в режиме реального времени МК должен решать следующие задачи:

- отсчет равных интервалов времени заданной длительности, повтор алгоритма управления по истечении каждого такого интервала (обычно эту функцию называют формированием меток реального времени);

- измерение длительности сигнала на линии ввода;

- подсчет числа импульсов внешнего сигнала на заданном временном интервале;

- формирование на линии вывода сигнала заданного логического уровня с программируемой задержкой по отношению к изменению сигнала на линии ввода;

- формирование на линии вывода импульсного сигнала с программируемой частотой и коэффициентом заполнения.

Решение перечисленных задач программным путем без использования аппаратных средств является неэффективным, так как занимает ресурсы, необходимые для вычислений. Поэтому для решения задач управления в 5 реальном времени используют специальные аппаратные средства, которые называют таймерами.

Типичный таймер представляет собой 16-разрядный счетчик со схемой управления (Рис. 8).

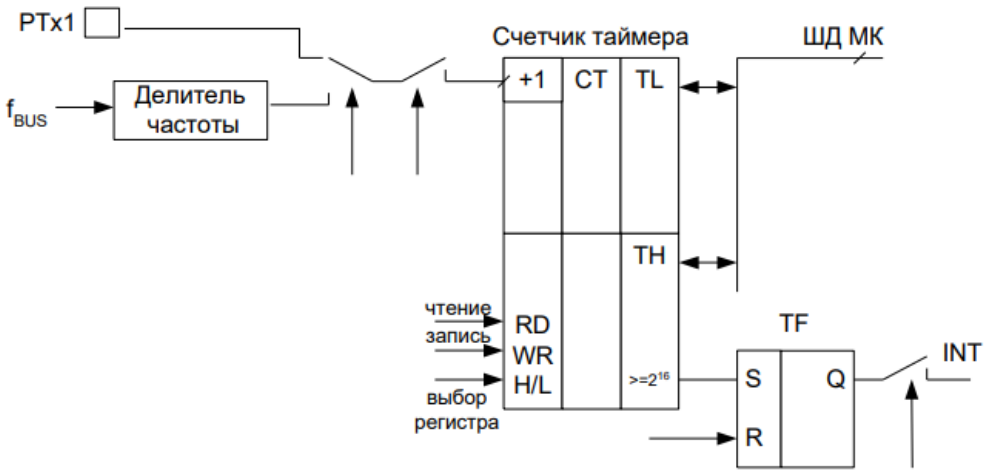


Рис. 8

В памяти МК счетчик отображается двумя регистрами: TH – старший байт счетчика, TL – младший

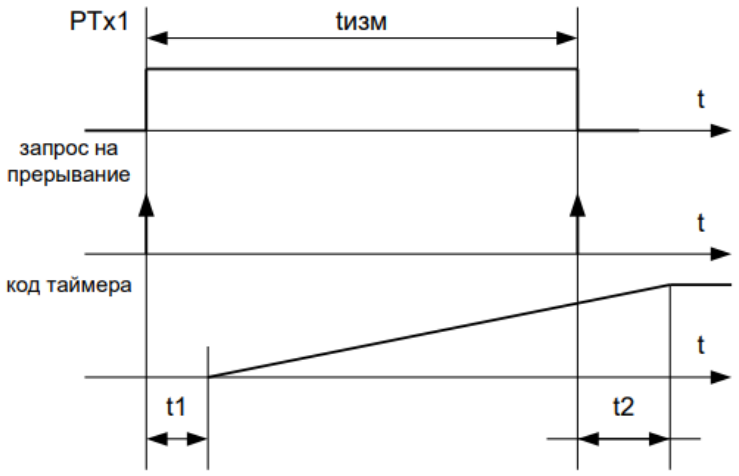
байт. Регистры доступны для чтения и для записи. Направление счета – только прямое, (т.е. при поступлении входных импульсов содержимое счетчика инкрементируется).

В зависимости от настройки счетчик может использовать один из источников входных сигналов:

- импульсную последовательность с выхода управляемого делителя частоты fBUS (режим таймера);
- внешнюю импульсную последовательность, поступающую на один из входов МК (режим счетчика событий).

При переполнении счетчика устанавливается триггер переполнения TF, который генерирует запрос на прерывание, если прерывания от таймера разрешены.

1. прерывается выполнение текущей программы при изменении сигнала на линии PTx1 с 0 на 1; в подпрограмме прерывания устанавливается регистр счетчика таймера в 0 и разрешается счет;
2. при изменении сигнала на линии с 1 на 0 еще раз прерывается выполнение программы; в п/п прерывания останавливается счет; код в регистрах счетчика будет равен длительности интервала, выраженной числом периодов частоты тактирования счетчика.



Моменты разрешения и остановки счета t1 и t2 не совпадают с моментами изменения сигнала на входе PTx1, так как пуск и останов выполняются в п/п прерывания. Ошибка счета равна t1-t2. Каждое из значений t1 и t2 определяется временем перехода МК к

выполнению п/п прерывания и временем выполнения некоторого числа команд. Максимальная ошибка может составить несколько десятков мкс, поэтому рассмотренный метод не может быть использован для измерения интервалов микросекундного диапазона.

Еще одним недостатком простейшего таймера-счетчика является невозможность формировать метки реального времени с периодом, отличным от периода полного коэффициента счета, равного 2 16 .

Эти недостатки устранены в усовершенствованном модуле таймерасчетчика, используемом в МК семейства MCS-51.

Реализован аппаратный пуск и останов таймера – это повышает точность измерения временных интервалов, так как интервалы t1 и t2 не являются 7 составляющими погрешности измерения.

Реализован режим перезагрузки счетчика произвольным кодом в момент переполнения – это позволяет формировать метки реального времени с периодом, отличным от периода полного коэффициента счета.

Главным недостатком модуля типичного таймера-счетчика является невозможность одновременного обслуживания нескольких каналов. Увеличение числа каналов производят следующими способами:

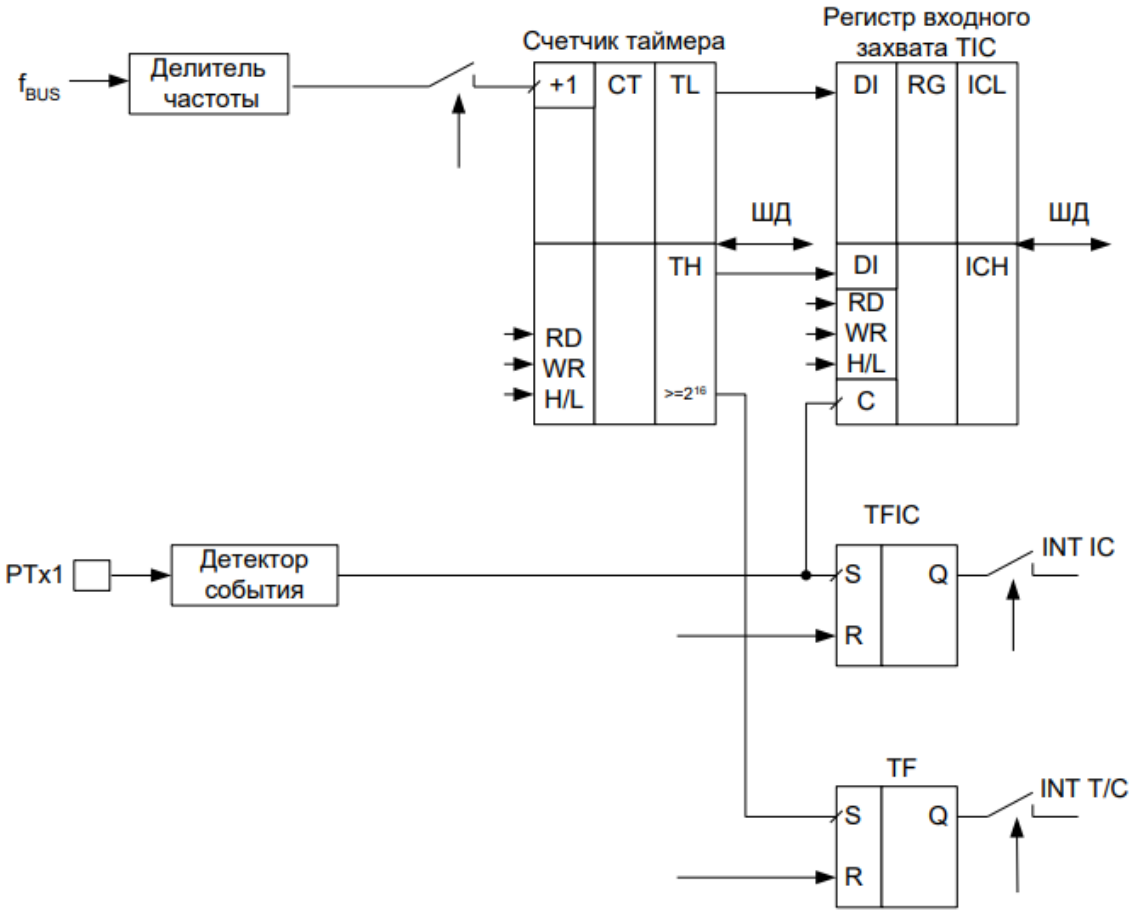
- увеличение числа модулей таймеров-счетчиков. Примеры: МК семейства MCS-51, МК компаний Mitsubishi и Hitachi;

- введение в структуру таймера-счетчика дополнительных аппаратных средств: канала входного захвата (Input Capture – IC) и канала выходного сравнения (Output Compare – OC).

8. Счетчик-таймер с каналом входного захвата. Функциональная схема. Измерение временного интервала; временные диаграммы.

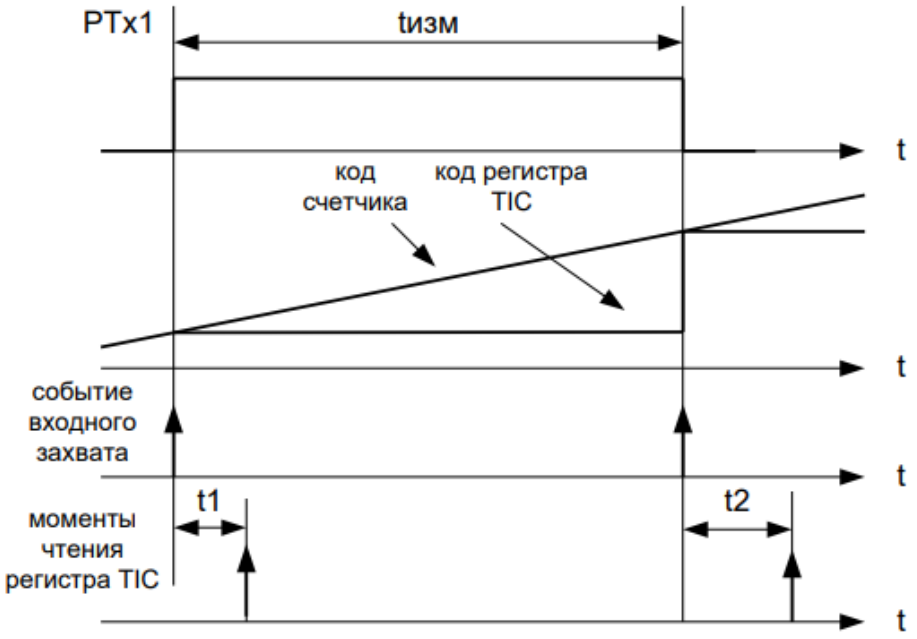
Детектор событий следит за уровнем напряжения на одном из входов МК (рис. 10). Обычно это одна из линий порта в/в. При изменении уровня логического сигнала детектор события вырабатывает строб записи, и текущее состояние счетчика-таймера записывается в 16-разрядный регистр входного захвата TIC. Это называется событием захвата. Возможны следующие варианты событий захвата:

- изменение логического сигнала с 0 на 1 (передний фронт сигнала);
- изменение логического сигнала с 1 на 0 (задний фронт входного сигнала);
- любое изменение логического уровня сигнала



Измерение временного интервала с помощью канала входного захвата показано на рис. 11. При изменении уровня входного сигнала с 0 на 1 код счетчика

К1 копируется в регистр TIC. Триггер TFIC устанавливается в 1. Формируется запрос на прерывание (таймер сообщает МК, что интервал начался). С задержкой t1 МК считывает код К1 из регистра TIC, сбрасывает триггер TFIC и инициализирует детектор событий на контроль за падающим фронтом сигнала PTx1. При изменении сигнала с 1 на 0 детектор снова фиксирует событие захвата, и код счетчика К2 копируется в регистр TIC. Снова выставляется запрос на прерывание. С задержкой t2 код считывается в память МК. Разность кодов К2-К1 и есть длительность измеряемого интервала



Достоинство: текущее состояние счетчика сохраняется аппаратными средствами, поэтому исключаются ошибки измерения входного интервала времени, связанные со временем перехода к подпрограмме обработки прерывания.

Недостаток: содержимое регистра входного захвата после первого события должно быть считано МК до того, как произойдет второе событие захвата. Время перехода к п/п обработки прерывания

накладывает ограничение на длительность измеряемого интервала времени

9. Счетчик-таймер с каналом выходного сравнения. Функциональная схема. Формирование на выходе микроконтроллера временного интервала заданной длительности; временные диаграммы.

Компаратор сравнивает текущий код счетчика таймера с кодом, который записан в 16-разрядном регистре выходного сравнения TOC (рис. 9.6). В момент равенства кодов на одном из выходов МК (PTx2) устанавливается заданный уровень логического сигнала. Рассмотренное действие называют событием выходного сравнения.

Возможны три типа изменения сигнала на выходе PTx2 в момент события выходного сравнения:

- установка высокого логического уровня;
- установка низкого логического уровня;
- инвертирование сигнала на выходе.

При наступлении события сравнения устанавливается в 1 триггер выходного сравнения TFOC. Состояние триггера выходного сравнения может 10 быть считано программно. Если разрешены прерывания по событию сравнения, то формируется запрос на прерывание INT OC.

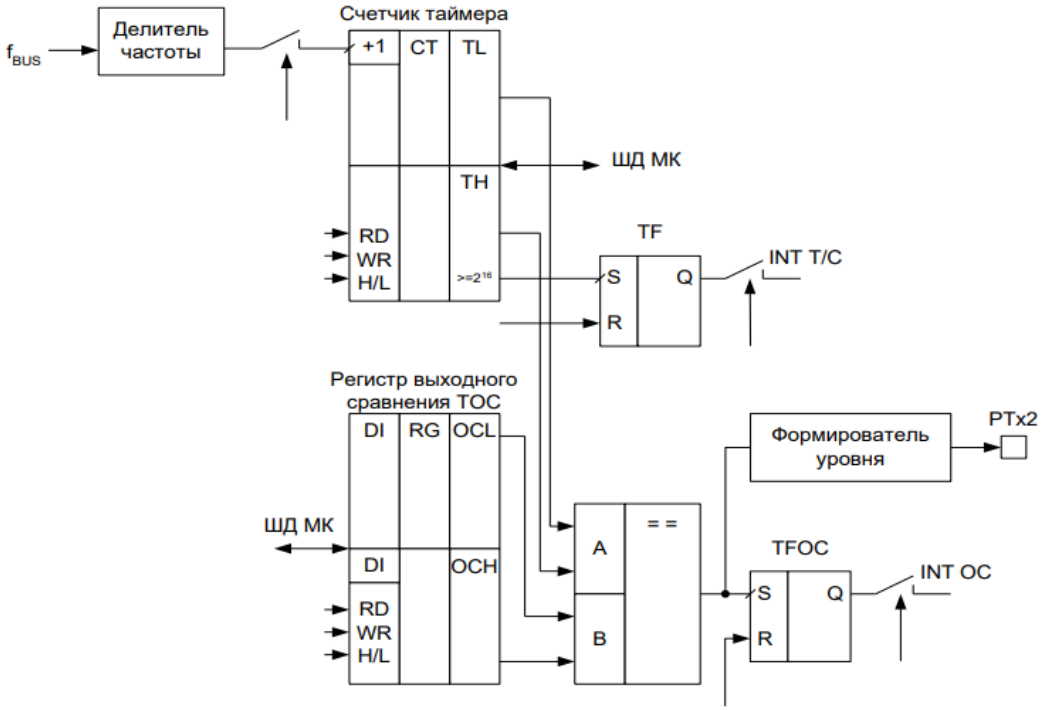
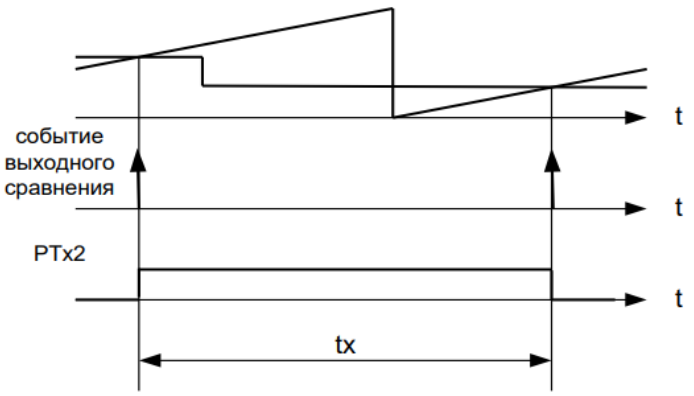


Рис. 12

Канал выходного сравнения позволяет сформировать на выходе временной интервал заданной длительности tx.



Первое событие сравнения в момент t1 формирует нарастающий фронт сигнала PTx2. Одновременно генерируется запрос на прерывание. В п/п прерывания происходит загрузка нового кода сравнения К2. Время, необходимое для записи нового значения в регистр сравнения TOC, ограничивает минимальную длительность формируемого интервала. В момент t2 наступает второе событие, и выход PTx2 устанавливается в 0. Длительность

сформированного интервала tx определяется только разностью кодов. В отличие от типичного таймера, где счетчик используется непосредственно для формирования кода измеряемого интервала, в усовершенствованном таймере счетчик лишь создает образ времени, подобно часам. Все действия по формированию или измерению временных интервалов производят аппаратные средства захвата и сравнения.

Поэтому счетчик в составе усовершенствованного таймера называют счетчиком временной базы.

Число каналов входного захвата и выходного сравнения в модуле усовершенствованного таймера МК может быть различным.

Для семейства HC05 Motorola типовыми решениями являются: 2IC+2OC или 1IC+1OC.

Усовершенствованный таймер позволяет решить многие задачи управления в реальном времени, но имеет следующие ограничения:

- недостаточное число каналов захвата и сравнения, принадлежащих одному счетчику;
- однозначно определенная конфигурация канала (или захват или сравнение)

10. Процессор событий. Структурная схема, режимы работы. Режим ШИМ, временная диаграмма.

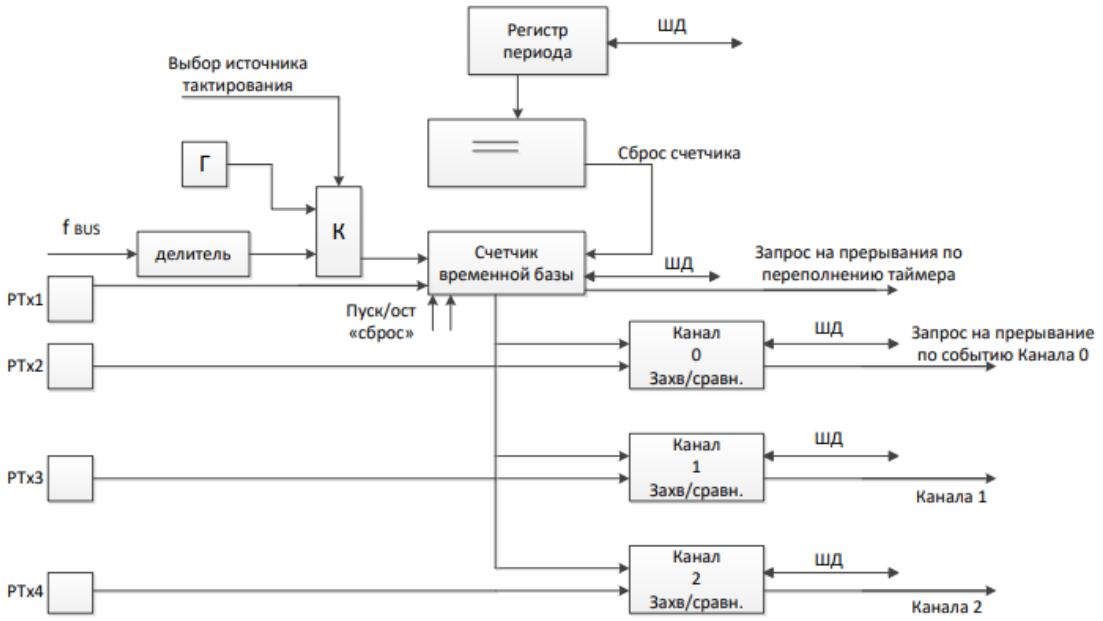
В процессорах событий устранены ограничения усовершенствованных таймеров. Процессоры событий были впервые использованы в МК семейства Intel 8xC51Fx.

Процессор событий в этом МК получил название программируемого счетного массива (Programmable Counter Array, PCA).

Процессор событий состоит из 16-разрядного счетчика временной базы и нескольких универсальных каналов захвата/сравнения.

Счетчик может тактироваться импульсной последовательностью с выхода делителя частоты или внешним генератором. Счетчик имеет опции пуска/останова и сброса в 0.

Некоторые модели процессора событий допускают изменение 12 коэффициента счета счетчика временной базы (изменение периода его работы). Для этого в составе модуля имеется двухбайтовый программно доступный регистр периода и компаратор. При совпадении текущего кода счетчика с кодом периода триггеры счетчика автоматически сбрасываются в 0.



Режимы работы универсальных каналов:

- режим входного захвата
- режим выходного сравнения
- режим широтно-импульсной модуляции (ШИМ)

Первые два режима аналогичны режимам модуля усовершенствованного таймера. Каждый канал

включает двухбайтовый регистр данных и триггер события. В зависимости от выбранного режима регистр данных используется для записи кода счетчика в момент наступления входного захвата или для хранения кода выходного сравнения. Триггер устанавливается при наступлении любого из этих событий.

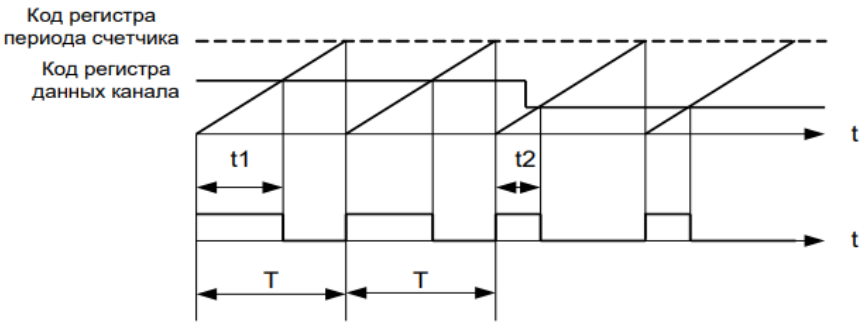


Рис. 15

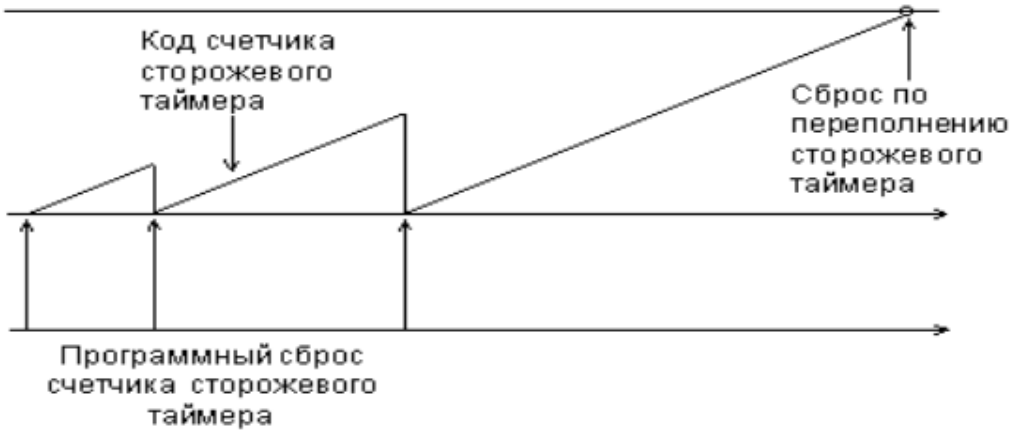
В режиме ШИМ на соответствующем выводе МК формируется последовательность импульсов с периодом, равным периоду работы счетчика временной базы. Длительность импульса прямо пропорциональна коду в регистре данных канала.

Коэффициент заполнения $\gamma=t1/T$.

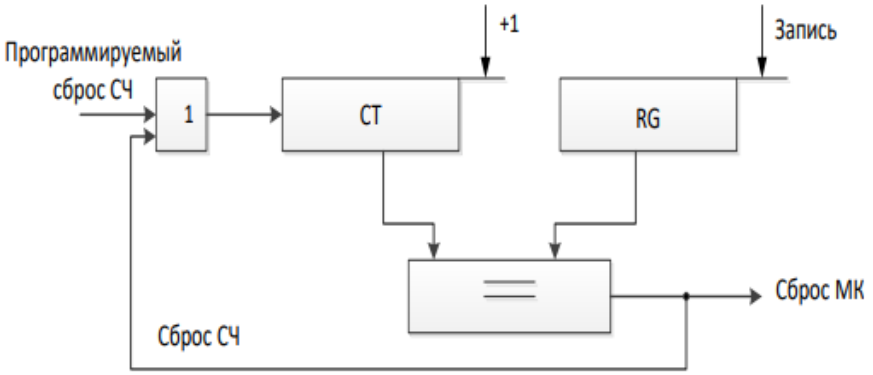
С помощью сигнала ШИМ осуществляется управление аналоговыми внешними устройствами, например, электродвигателями, светодиодами и т.д.

11. Сторожевой таймер: назначение, принцип работы. Пояснить временной диаграммой. Функциональная схема.

Если МК неожиданно "завис", то на случай выхода из этого состояния все современные контроллеры имеют встроенный модуль сторожевого таймера. Принцип действия сторожевого таймера показан на рис. 16



Структурная схема сторожевого таймера показана на рис 17. Максимальный код счетчика записывается в специальном регистре. Основу сторожевого таймера составляет многоразрядный счетчик.



При сбросе МК счетчик обнуляется. После перехода МК в активный режим работы значение счетчика начинает увеличиваться независимо от выполняемой программы. При достижении счетчиком максимального кода генерируется сигнал внутреннего сброса, и МК начинает выполнять рабочую программу сначала.

Для исключения сброса по переполнению сторожевого таймера рабочая программа МК должна периодически сбрасывать счетчик. Сброс счетчика сторожевого таймера осуществляется путем исполнения специальной команды (например, CLRWDT) или посредством записи некоторого указанного кода в один из регистров специальных функций. Тогда при

нормальном, предусмотренном разработчиком, порядке исполнения рабочей программы, переполнения счетчика сторожевого таймера не происходит, и он не оказывает влияния на работу МК. Однако, если исполнение рабочей программы было нарушено, например, вследствие "зависания", то велика вероятность того, что счетчик не будет сброшен вовремя. Тогда произойдет сброс по переполнению сторожевого таймера, и нормальный ход выполнения рабочей программы будет восстановлен. Сброс счетчика может осуществляться: 1) путем исполнения специальной команды; 2) периодически меняя значение регистра (рис. 17).

Модули сторожевых таймеров конкретных МК могут иметь различные особенности:

- в ряде МК векторы внешнего сброса и сброса по переполнению сторожевого таймера совпадают. Это не позволяет выявить причину сброса программным путем и затрудняет написание рабочей программы. Более высокоуровневые МК имеют либо различные векторы сброса, либо отмечают 1 CT RG Сброс МК +1 Сброс СЧ Программируемый сброс СЧ Запись 15 событие сброса по переполнению сторожевого таймера установкой специального бита в одном из регистров специальных функций;

- в некоторых МК при переходе в один из режимов пониженного энергопотребления, когда рабочая программа не выполняется, автоматически приостанавливается работа сторожевого таймера. В других МК сторожевой таймер имеет независимый тактовый генератор, который продолжает функционировать и в режиме ожидания. В этом случае необходимо периодически выводить МК из состояния ожидания для сброса сторожевого таймера. В PIC-контроллерах фирмы Microchip выработка таких сбросов может быть запрещена путем записи нуля в специальный бит конфигурации WDTE.

На выполнение функций сторожевого таймера может быть запрограммирован один из универсальных каналов процессора событий.

Использование сторожевого таймера существенно повышает способность к самовосстановлению системы на основе МК.

12. Модуль аналогового ввода-вывода в составе микроконтроллера (АЦП). Функциональная схема, режимы работы.

Прикладная программа, записанная в память программ МК, должна обеспечивать его надежное функционирование. Однако в результате электромагнитных помех, колебаний напряжения питания и других внешних факторов могут произойти сбои в работе МК..С целью обеспечения надежного запуска, контроля работы МК и восстановления работоспособности системы в отсутствие оператора все современные МК снабжаются аппаратными средствами обеспечения надежной работы. К ним относятся:

- схема формирования сигнала сброса МК;
- модуль мониторинга напряжения питания;
- сторожевой таймер.

Необходимость приема и формирования аналоговых сигналов требует наличия в МК модулей аналогового ввода/вывода.

Простейшим устройством аналогового ввода в МК является встроенный компаратор напряжения. Основная функция компаратора – это сравнение двух напряжений, одно из которых образцовое или опорное, а другое собственно измеряемое. Выходной сигнал компаратора может принимать лишь два значения: логический ноль, и логическая же единица

Компараторы удобнее всего использовать для контроля определенного значения входного напряжения, например, в термостатах. Компаратором можно отслеживать уровень заряда батареи по просадке напряжения или момент перехода переменного напряжения через ноль. Также он может вызывать

прерывания, если результат сравнения изменился, и управлять схемой захвата таймера.

Работу встроенного аналогового компаратора можно понять, взглянув на временную диаграмму (Рис.20), на которой видно, что на инвертирующий вход подано опорное напряжение, лежащее в диапазоне (0-5)V. Если напряжение на неинверсном входе меньше опорного, то в бите AC0 - 0, а если больше, то в бите AC0 — 1. Этот бит находится в регистре управления компаратором

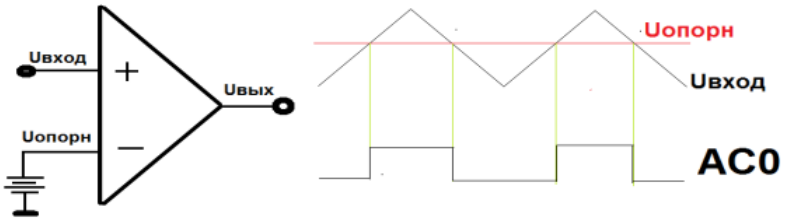
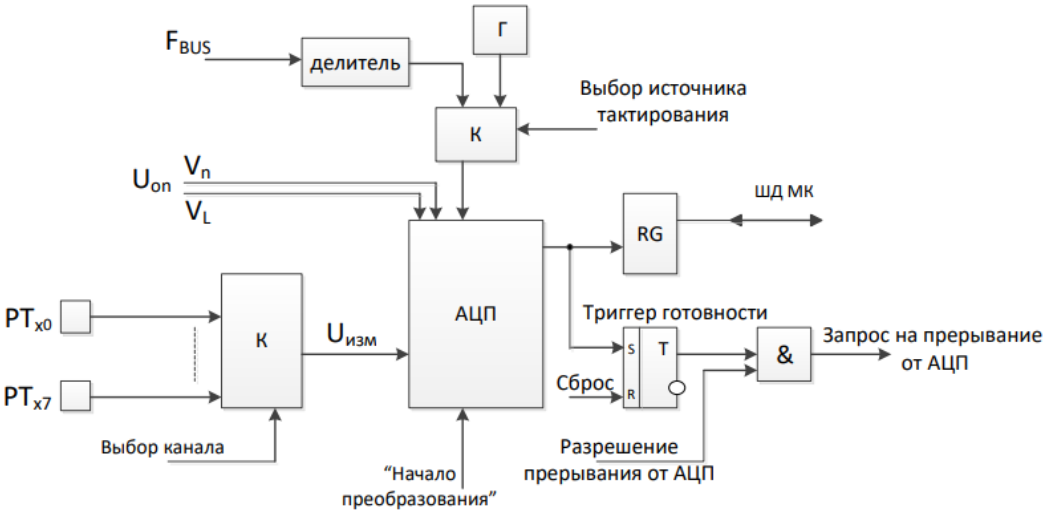


Рис.20

В комбинации с внешним генератором линейно изменяющегося напряжения встроенный компаратор позволяет реализовать на МК однобитный аналого-цифровой преобразователь (АЦП).

Однако более широкие возможности для работы с аналоговыми сигналами дает АЦП, встроенный в МК. Чаще всего он реализуется в виде модуля многоканального АЦП, предназначенного для ввода в МК аналоговых сигналов с датчиков физических величин и преобразования этих сигналов в двоичный код.



Структурная схема типового модуля АЦП представлена на рис. 21.

Многоканальный аналоговый коммутатор К служит для подключения одного из источников аналоговых сигналов (PTx0...PTx7) ко входу АЦП. Выбор источника сигнала для преобразования осуществляется посредством записи номера канала коммутатора в соответствующие разряды регистра управления АЦП.

Два вывода модуля АЦП используются для задания опорного напряжения Uon: VH — верхний предел, VL — нижний предел. Разность потенциалов на входах VH и VL и составляет Uon. Uon = Vn – VL,

Максимальное значение опорного напряжения, как правило, равно напряжению питания МК.

Если измеряемое напряжение Uизм > Vн, результат преобразования будет равен FF. При Uизм < VL результат 00.

Разрешающая способность АЦП составляет Uon/2n , где n — число двоичных разрядов в слове результата.

Для достижения максимальной точности измерения следует выбрать максимально допустимое значение Uon.

Собственно, аналого-цифровой преобразователь выполнен по методу последовательного приближения. Практически во всех моделях 8-разрядных МК

разрядность АЦП также составляет 8 разрядов. Соответственно, формат представления результатов измерения АЦП — однобайтовый. Исключение составляют лишь модули АЦП микроконтроллеров для управления преобразователями частоты для электроприводов, разрешающая способность которых равна 10 разрядам. Два младших разряда результата получают с помощью дополнительного емкостного делителя, не связанного с регистром последовательного приближения.

Длительность такта преобразования задает генератор синхронизации: один цикл равен двум периодам частоты генератора tADC. Время преобразования для типовых модулей АЦП микроконтроллеров составляет от единиц до десятков микросекунд.

Источником синхронизации модуля АЦП может служить встроенный RC-генератор (Г) или импульсная последовательность тактирования межмодульных магистралей МК. В первом случае частота синхронизации АЦП обязательно окажется оптимальной, то есть той, которая рекомендуется в техническом описании. Во втором случае выбранная по другим соображениям fBUS может оказаться неподходящей для модуля АЦП. На этот случай в составе некоторых модулей предусмотрен программируемый делитель частоты fBUS.

Момент завершения каждого цикла преобразования отмечается установкой триггера готовности данных. Если прерывания от модуля АЦП разрешены, то генерируется запрос на прерывания. Как правило, чтение регистра результата сбрасывает триггер готовности.

Большинство модулей АЦП имеют только режим программного запуска: установка одного из битов регистра режима запускает очередное измерение. Наиболее универсальные модули АЦП имеют также режим автоматического запуска, при котором после завершения одного цикла преобразования немедленно

начинается следующий. Однако данные измерения каждого цикла должны быть считаны программным способом.

Цифро-аналоговые преобразователи в составе МК являются большой редкостью. Функция цифро-аналогового преобразователя реализуется средствами модуля программируемого таймера в режиме ШИМ. На одном из выводов МК формируется высокочастотная импульсная последовательность с регулируемой длительностью импульса. Полученный сигнал сглаживается фильтром нижних частот на операционном усилителе. Разрешающая способность такого ЦАП определяется дискретностью регулирования коэффициента заполнения в режиме ШИМ

короткие вопросы

5. Для чего нужен канал входного захвата?

Канал входного захвата в таймерах микроконтроллеров используется для измерения временных интервалов между событиями, например, для измерения периода сигнала или длительности импульса. Канал захвата позволяет фиксировать момент точного срабатывания таймера по внешнему событию, что позволяет получать более точные данные для контроля времени. Например, канал захвата может быть использован для измерения скорости вращения вала двигателя на предприятии.

6. Для чего нужен канал выходного сравнения?

Канал выходного сравнения в таймерах микроконтроллеров используется для управления внешними устройствами, такими как светодиоды, моторы, реле и другие устройства, которые нужно включать или выключать в зависимости от состояния таймера. Как правило, канал выходного сравнения позволяет генерировать прерывание, когда значение таймера достигает заданного уровня, что позволяет

более точно контролировать время работы устройств. Также канал выходного сравнения может использоваться для реализации PWM

7. Чем характеризуется режим ожидания в микроконтроллерах?

Режим ожидания (sleep mode) в микроконтроллерах характеризуется низким энергопотреблением и отсутствием выполнения программных инструкций. В этом режиме микроконтроллер может находиться в течение длительного времени, пока не произойдет какое-либо событие, которое приведет к переходу в другой режим работы (например, прерывание от внешнего источника, изменение состояния входного пина и т.д.). Режим ожидания используется для экономии энергии в батарейных и портативных устройствах, а также в системах, где не требуется непрерывной работы микроконтроллера.

8. Где применяется CAN-протокол?

CAN-протокол (Controller Area Network) применяется в автомобильной промышленности для передачи данных между различными электронными блоками управления автомобиля, такими как двигатель, трансмиссия, электронная система стабилизации и т.д. Он также используется в других областях, где требуется передача данных с высокой скоростью, таких как промышленность, авиация, морская и железнодорожная техника, а также в медицине и коммуникационной оборудовании.

13. Модуль прерываний микроконтроллера. Виды прерываний; источники прерываний. Поллинг.

Модуль прерываний МК

Обработка прерываний в МК происходит в соответствии с общими принципами обработки прерываний в МПС. Модуль прерываний принимает запросы прерывания и организует переход к выполнению

определенной прерывающей программы. Запросы прерывания могут поступать как от внешних источников, так и от источников, расположенных в различных внутренних модулях МК. В качестве входов для приема запросов от внешних источников чаще всего используются выводы параллельных портов ввода/вывода, для которых эта функция является альтернативной. Источниками запросов внешних прерываний также могут быть любые изменения внешних сигналов на некоторых специально выделенных линиях портов ввода/вывода.

Источниками внутренних запросов прерываний могут служить следующие события: ·

- переполнение таймеров/счетчиков;
- сигналы от каналов входного захвата и выходного сравнения таймеров/счетчиков или от процессора событий;
- готовность памяти EEPROM;
- сигналы прерывания от дополнительных модулей МК, включая завершение передачи или приема информации по одному из последовательных портов и другие.

Любой запрос прерывания поступает на обработку, если прерывания в МК разрешены и разрешено прерывание по данному запросу. Адрес, который загружается в программный счетчик при переходе к обработке прерывания, называется "вектор прерывания". В зависимости от организации модуля прерываний конкретного МК различные источники прерываний могут иметь разные векторы или использовать некоторые из них совместно.

Вопрос о приоритетах при одновременном поступлении нескольких запросов на прерывание решается в различных МК по-разному. Есть МК с одноуровневой системой приоритетов (все запросы равноценны), многоуровневой системой с фиксированными приоритетами и многоуровневой программируемой системой приоритетов.

Если система многоуровневая, то к одному уровню могут относиться несколько запросов. Внутри одного уровня для каждого источника фиксируется его старшинство. И тогда в случае появления запросов одного уровня, очередность их обслуживания определяется с помощью внутренней процедуры - так называемого *поллинга*. Если одновременно

приняты запросы с одинаковым уровнем приоритета, обработка их будет производиться в порядке, задаваемом последовательностью внутреннего опроса флагов прерываний. Таким образом, в пределах одного приоритетного уровня существует еще одна структура приоритетов.

Поллинг - последовательный опрос по старшинству источников одного уровня.

Высший уровень бывает у внешних источников, а низший у последовательных источников.

Отдельно необходимо описать аппаратные прерывания, связанные с включением питания, подачей сигнала "сброс" и переполнением сторожевого таймера. Они имеют немаскируемый характер и чаще всего разделяют один общий вектор прерывания. Это вполне логично, поскольку результатом каждого из событий является начальный сброс МК.

14. Режимы энергопотребления в микроконтроллерах: активный, ожидания, останова. Мощность потребления; группы микроконтроллеров по величине напряжения питания.

Малый уровень энергопотребления является зачастую определяющим фактором при выборе способа реализации цифровой управляющей системы. Современные МК предоставляют пользователю большие возможности в плане экономии энергопотребления и имеют, как правило, следующие о***сновные режимы работы***:

1) активный режим (Run mode) — основной режим работы МК. В этом режиме МК исполняет рабочую программу, и все его ресурсы доступны. Потребляемая мощность имеет максимальное значение PRUN. Большинство современных МК выполнено по КМОП-технологии, поэтому мощность потребления в активном режиме сильно зависит от тактовой частоты;

2) режим ожидания (Wait mode, Idle mode или Halt mode – холостой ход). В этом режиме прекращает работу центральный процессор, но продолжают функционировать периферийные модули, которые контролируют состояние объекта управления. При необходимости сигналы

от периферийных модулей переводят МК в активный режим, и рабочая программа формирует необходимые управляющие воздействия. Перевод МК из режима ожидания в рабочий режим осуществляется по прерываниям от внешних источников или периферийных модулей, либо при сбросе МК. В режиме ожидания мощность потребления МК PWAIT снижается по сравнению с активным режимом в 5…10 раз; PRUN=(5-10)PWAIT.

3) режим останова (Stop mode, Sleep mode или Power Down mode, режим микропотребления). В этом режиме прекращает работу как центральный процессор, так и большинство периферийных модулей. Приостанавливается выполнение всех функций, т.к. прекращает работать синхрогенератор. Значения регистров специальных функций теряется, но сохраняется содержимое ОЗУ. Переход МК из состояния останова в рабочий режим возможен, как правило, только по прерываниям от внешних источников или после подачи сигнала сброса. В режиме останова мощность потребления МК PSTOP снижается по сравнению с активным режимом примерно на три порядка и составляет единицы микроватт: PRUN ≈ 10^3 PSTOP.

Два последних режима называют режимами пониженного энергопотребления. *Минимизация* энергопотребления системы на МК достигается за счет оптимизации мощности потребления МК в активном режиме, а также использования режимов пониженного энергопотребления. При этом необходимо иметь в виду, что режимы ожидания и останова существенно отличаются временем перехода из режима пониженного энергопотребления в активный режим. Выход из режима ожидания обычно происходит в течение 3…5 периодов синхронизации МК, в то время как задержка выхода из режима останова составляет несколько тысяч периодов синхронизации. Кроме снижения динамики работы системы значительное время перехода в активный режим является причиной дополнительного расхода энергии.

4) Еще один режим пониженного энергопотребления - ***экономичный режим*** (Power save) – в AVR. Продолжает работать только генератор таймера, временная база сохраняется, все остальные функции выключены.

В зависимости от диапазона питающих напряжений все МК можно разделить на ***три основные группы***:

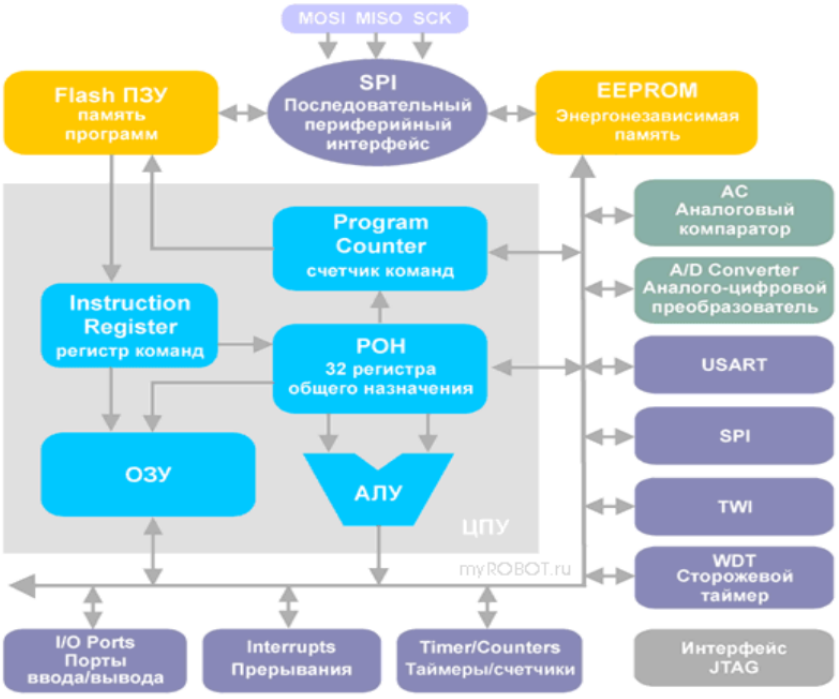
- МК с напряжением питания 5,0 В±10%. Эти МК предназначены, как правило, для работы в составе устройств с питанием от промышленной или бытовой сети, имеют развитые функциональные возможности и высокий уровень энергопотребления.
- lМК с расширенным диапазоном напряжений питания: от 2,0…3,0 В до 5,0-7,0 В. МК данной группы могут работать в составе устройств как с сетевым, так и с автономным питанием.
- МК с пониженным напряжением питания: от 1,8 до 3 В. Эти МК предназначены для работы в устройствах с автономным питанием и обеспечивают экономный расход энергии элементов питания

15. Способы задания тактовой частоты микроконтроллеров. Виды времязадающих элементов, их характеристики. Структурная схема системы синхронизации на примере микроконтроллеров AVR фирмы Atmel. Какие могут быть источники тактового сигнала?

На практике используются три основных способа задания тактовой частоты генератора: с помощью кварцевого резонатора, керамического резонатора и внешней RC-цепи.

В качестве времязадающих можно использовать емкостно-резисторные или индуктивно-резисторные линейные формирующие цепи, линии задержки и колебательные контуры. В большинстве генераторов импульсов в качестве времязадающих используются RC - цепи, что объясняется их простотой и технологичностью изготовления. Индуктивно-резисторные цепи и колебательные LC-контуры используются существенно реже и в основном в генераторах импульсов на активных двухполюсниках, что объясняется плохой технологичностью катушек индуктивности.

Обобщенная структурная схема МК AVR приведена на рис.63.



Для того чтобы МК заработал, необходимо подать на ЦПУ тактовые импульсы. Чем выше их частота, тем быстрее выполняются операции, а чем ниже их частота, тем меньше потребление тока. Формированием тактовых частот занимается подсистема синхронизации (рис.22). На её структурной схеме имеется несколько встроенных генераторных узлов. Расшифровка сокращений: HF (High Frequency) — высокочастотный, LF (Low Frequency) — низкочастотный, CLK (CLocK) — тактирование.

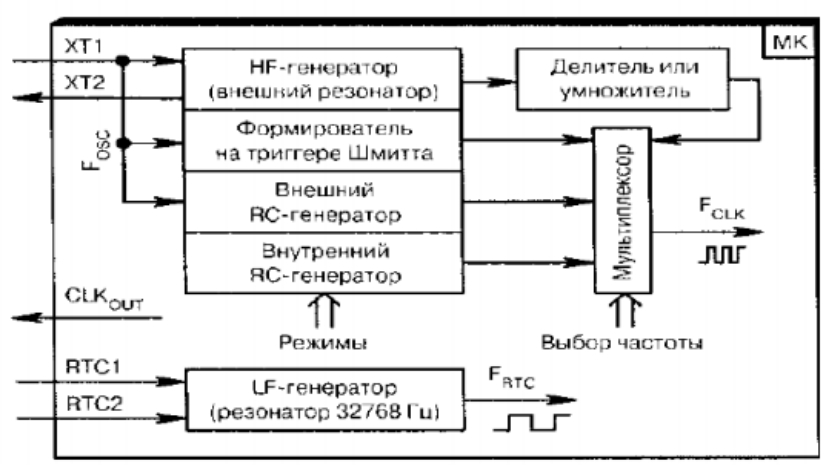


Рис.22

Источники синхронизации

- Внешний кварцевый/керамический резонатор
- Внешний низкочастотный кварцевый резонатор
- Внешний RC-генератор
- Встроенный калиброванный RC-генератор
- Внешняя синхронизация

16. Аппаратные средства обеспечения надежной работы микроконтроллеров. Типовые схемы формирования сигнала сброса. Блок детектирования пониженного напряжения питания. Подсистема начального сброса на примере микроконтроллеров AVR фирмы Atmel.

Типовые схемы формирования сигнала внешнего сброса для МК с высоким активным уровнем сигнала сброса (а) и низким активным уровнем сигнала сброса (б) представлены на рис. 18.

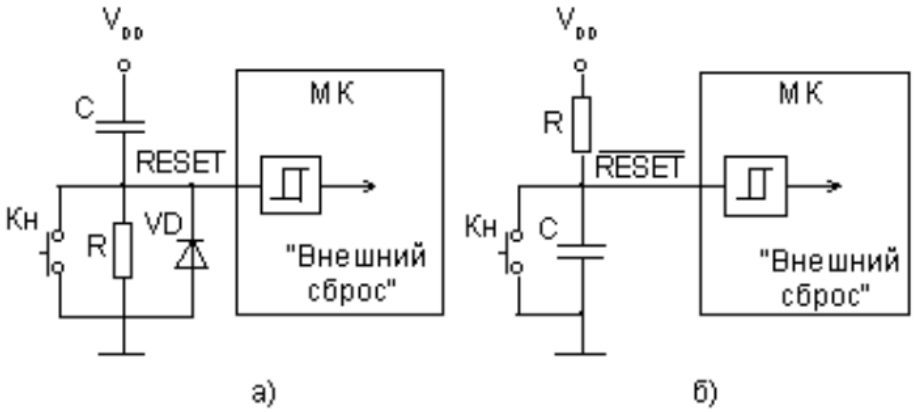


Рис. 18

Блок детектирования пониженного напряжения питания

В реальных условиях эксплуатации может сложиться такая ситуация, при которой напряжение питания МК опустится ниже минимально допустимого, но не достигнет порога отпускания схемы POR. В этих условиях МК может "зависнуть". При восстановлении напряжения питания до номинального значения МК останется неработоспособным.

Для восстановления работоспособности системы после "просадки" напряжения питания МК необходимо снова сбросить. Для этой цели в

современных МК реализован дополнительный блок детектирования пониженного напряжения питания. Такой модуль используется в МК семейства HC08 фирмы Motorola, аналогичный модуль имеется в составе семейства PIC17 фирмы Microchip. Рассматриваемый модуль отслеживает изменение напряжения питания и при его снижении до уровня чуть ниже минимально допустимого генерирует сигнал внутреннего сброса. Когда напряжение питания возрастет до порогового, запускается таймер задержки сброса. После задержки сигнал внутреннего сброса снимается, и происходит запуск МК. Уровень срабатывания блока детектирования пониженного напряжения питания значительно превышает напряжение сохранения данных в ОЗУ МК. Событие сброса по сигналу блока пониженного напряжения питания отмечается специальным битом в одном из регистров МК. Следовательно, программно анализируя этот бит после сброса МК, можно установить, что данные целы, и продолжить выполнение программы.

На рис.19 показана ***подсистема начального сброса микроконтроллеров*** AVR семейств ATmega, ATtiny. Схема включает модуль мониторинга напряжения питания и сторожевой таймер:

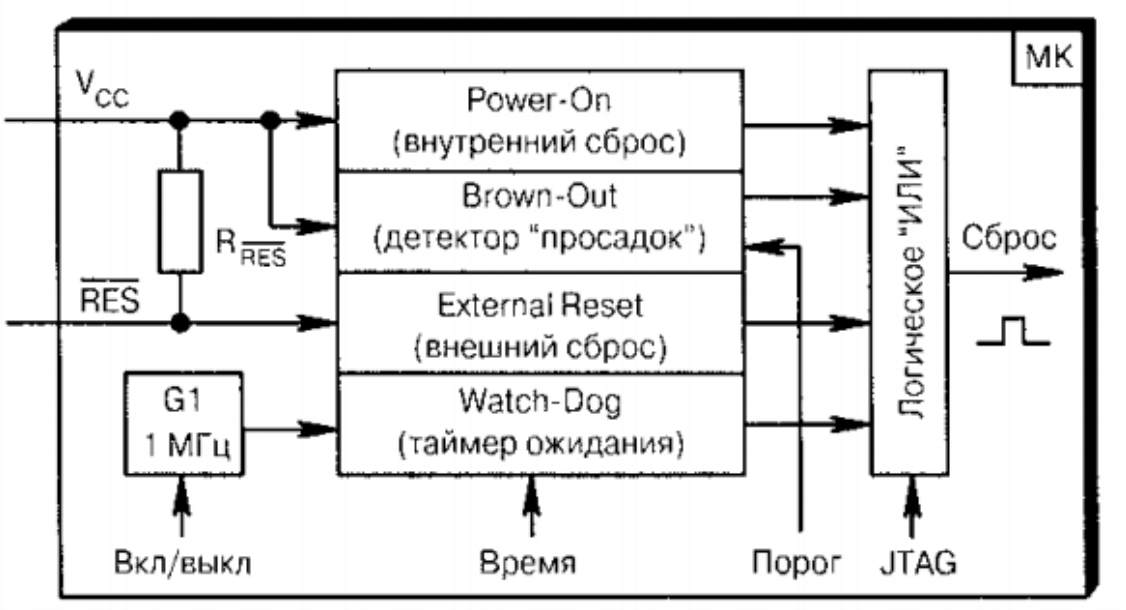


Рис.19

Источники сброса:

- Power-On — внутренний автоматический сброс, который активизируется сразу после подачи питания; ·
- Brown-Out — сброс от внутреннего детектора «просадок» питающего напряжения; ·
- External Reset — внешний сброс НИЗКИМ уровнем на выводе ; ·
- Watch-Dog — сброс от внутреннего «сторожевого» таймера при случайной остановке работы ЦПУ или зависании программы;
- JTAG — программный сброс через отладочный интерфейс JTAG.

Все источники сброса равноценны, что подчёркивает логический элемент «ИЛИ», находящийся внутри МК. Установка режимов сброса производится конфигурационными битами, а также программно-доступными регистрами из области SFR. Настраиваться могут: порог срабатывания детектора «просадок» напряжения, длительность времени задержки таймера ожидания Watch-Dog, моменты включения/отключения генератора G1. При наступлении одного из перечисленных событий формируется внутренний сигнал сброса ВЫСОКОГО уровня, во все регистры ввода/вывода заносятся их начальные значения, а в счетчик команд загружается значение $000 (адрес вектора сброса).

Одновременно запускается таймер формирования задержки сброса. По истечении определенного промежутка времени (предполагается, что за это время микроконтроллер перейдет в определенное устойчивое состояние) внутренний сигнал сброса переводится в НИЗКИЙ уровень

Там очень много, все вставлять не буду(можно прочитать в пункте 3.5.3)

короткие вопросы

9. Что такое bit staffing в CAN-протоколе?

Методы обнаружения ошибок.

CAN протокол определяет пять способов обнаружения ошибок в сети:

- Bit monitoring
- ACKnowledgement Check
- Bit stuffing
- CRC Check
- Frame check
- *Разрядная ошибка* **Bit monitoring** появляется, когда передатчик

10. Чем характеризуется режим ШИМ?

В ШИМ в качестве ключевых элементов используют транзисторы (могут быть применены и другие полупроводниковые приборы), работающие не в линейном, а в ключевом режиме, то есть транзистор всё время либо разомкнут (выключен), либо замкнут (находится в состоянии насыщения).

В режиме ШИМ на соответствующем выводе МК формируется последовательность импульсов с периодом, равным периоду работы счетчика временной базы. Длительность импульса прямо пропорциональна коду в регистре данных канала.

17. Задачи, решаемые с помощью модулей контроллеров последовательного вводавывода? Классификация интерфейсов последовательного ввода-вывода?

Задачи, которые решаются средствами модуля контроллера последовательного ввода/вывода, можно разделить на три основные группы: 1)связь встроенной микроконтроллерной системы с системой управления верхнего уровня, например, с персональным компьютером(RS-232C и RS-485); 2)связь с внешними по отношению к МК периферийными ИС, а также с датчиками физических величин с последовательным выходом(I2C, SPI), а также нестандартные протоколы обмена;
3)интерфейс связи с локальной сетью в мультимикроконтроллерных системах.

В системах с числом МК до пяти обычно используются сети на основе интерфейсов I2C, RS-232C и RS-485 с собственными сетевыми протоколами высокого уровня.

С точки зрения организации обмена информацией упомянутые типы интерфейсов последовательной связи отличаются режимом передачи данных (синхронный или асинхронный), форматом кадра (число бит в посылке при передаче байта полезной информации) и временными диаграммами сигналов на линиях .

Число линий, по которым происходит передача в последовательном коде, обычно равно двум (I2C, RS-232C, RS-485) или трем (SPI, некоторые нестандартные протоколы). Данное обстоятельство позв-т спроектировать модули контроллеров послед-го обмена таким образом, чтобы с их помощью на аппаратном уровне можно было реализовать несколько типов последовательных интерфейсов. При этом режим передачи и формат кадра поддерживаются на уровне логических сигналов, а реальные физические уровни сигналов для каждого интерфейса получают с помощью специальных

ИС, которые называют приемопередатчиками, конверторами, трансиверами. Среди различных типов встроенных контроллеров последовательного обмена, которые входят в состав тех или иных 8-разрядных МК, сложился стандарт «де-факто» — модуль UART (Universal Asynchronous Receiver and Transmitter). UART — это универсальный асинхронный приемопередатчик. Однако большинство модулей UART, кроме асинхронного режима обмена, способны также реализовать режим синхронной передачи данных. Модули типа UART в асинхронном режиме работы позволяют реализовать протокол обмена для интерфейсов RS-232C, RS-422A, RS-485, в синхронном режиме — нестандартные синхронные протоколы обмена, и в некоторых моделях.

Классификация, еще раз:

Интерфейсы характеризуются следующими параметрами:

1) пропускной способностью интерфейса — количеством информации, которая может быть передана через интерфейс в единицу времени;

2) максимальной частотой передачи информационных сигналов через интерфейс;

3) информационной шириной интерфейса — числом бит или байт данных, передаваемых параллельно через интерфейс;

4) максимально допустимым расстоянием между соединяемыми устройствами;

5) динамическими параметрами интерфейса — временем передачи отдельного слова или блока данных с учетом продолжительности процедур подготовки и завершения передачи;

6) общим числом проводов (линий) в интерфейсе.

В настоящее время не существует однозначной классификации интерфейсов. Можно выделить следующие четыре классификационных признака интерфейсов:

- способ соединения компонентов системы (радиальный, магистральный, смешанный);
- способ передачи информации (параллельный, последовательный, параллельно-последовательный);
- принцип обмена информацией (асинхронный, синхронный);
- режим передачи информации (двусторонняя поочередная передача, односторонняя передача).

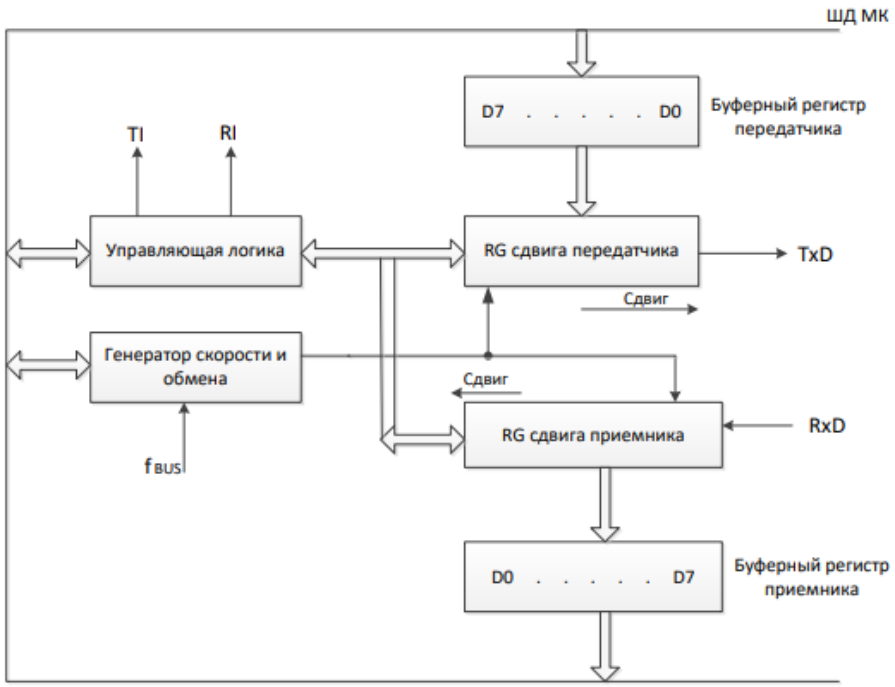
18. Последовательный обмен в МП-системах с помощью модуля UART. Структурная схема модуля UART, ее основные элементы. Передача и прием данных модулем

UART. В каком случае могут возникнуть ошибки при считывании данных в микроконтроллер?

Среди различных типов встроенных контроллеров последовательного обмена, которые входят в состав тех или иных 8-разрядных МК, сложился стандарт "де-факто" — модуль UART (Universal Asynchronous Receiver and Transmitter). UART — это универсальный асинхронный приемопередатчик, работающий в дуплексном режиме. Несмотря на то, что в названии отсутствует упоминание о синхронном обмене, в большинстве подобных модулей он, тем не менее, реализован. Однако это название не является единственным, и некоторые производители МК используют свои имена: USART – Universal Synchronous/Asynchronous Receiver and Transmitter, SCI – Serial Communication Interface, USI – Universal Serial Interface и др. Модули типа UART в асинхронном режиме работы позволяют реализовать протокол обмена для интерфейсов RS-232C, RS-422A, RS-485, в синхронном режиме — нестандартные синхронные протоколы обмена, и в некоторых моделях — SPI. Асинхронная передача данных. При асинхронной последовательной передаче, структура организации которой показана на рис. 29, между передатчиком и приемником нет линии синхронизации, импульсы на которой выделяли моменты передачи отдельных битов, и передача информационного слова может производиться асинхронно, в любой произвольный момент времени.

Структурная схема:

В модуле используется двойная буферизация, его структура содержит 2 независимые подсистемы – приемник и передатчик, поэтому возможен режим одновременного приема и передачи информации.



Основу каждой подсистемы составляют последовательный сдвиговый регистр и параллельный регистр буфера данных. Буферные регистры приемника и передатчика отображаются в памяти МК одним адресом, но при записи данные помещаются в буфер передатчика, а при чтении считываются из буфера приемника. Для настройки модуля устанавливаются некоторые биты в управляющих регистрах.

Передача данных от микроконтроллера к другим устройствам.

В отсутствие передачи на линии всегда обеспечивается уровень сигнала, соответствующий логической 1. Для передачи байт данных записывается в буферный регистр передатчика. Если передача разрешена, содержимое буферного регистра загружается в регистр сдвига передатчика.

Передача каждого кадра начинается посылкой стартового бита (изменением значения сигнала в линии из 1 в 0 на один период стробирующих сигналов). Генератор скорости обмена управляет сдвиговым регистром, и биты из него последовательно, начиная с младшего D0, передаются на выход TxD. Если в процессе обмена производится контроль по четности, то затем передается бит четности. В

завершение посылается стоп-бит, и линия данных в течение одного такта находится в состоянии простоя.

По завершении передачи байта данных устанавливается в «1» бит TI, т.е. буфер пуст и в него могут загружаться новые данные. Бит TI м.б. считан программно и генерируется запрос на прерывание, если прерывания от передатчика в МК разрешены. Старт-бит следующего кадра посылается в любой момент после стоп-бита, то есть между передачами возможны паузы произвольной длительности.

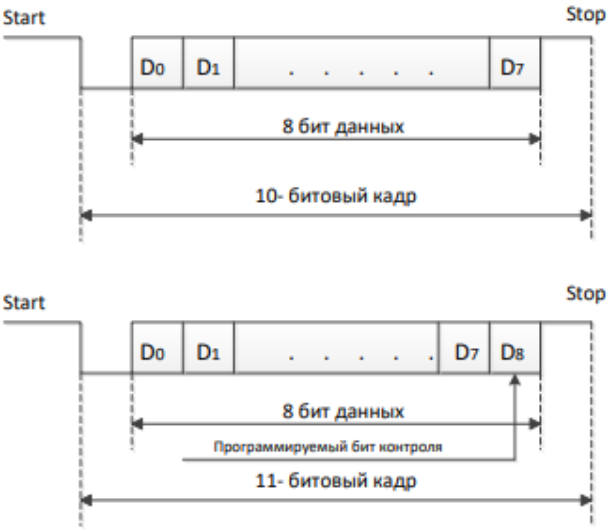
Прием данных в микроконтроллер.

Если работа приемника разрешена, распознается стартовый бит, передний фронт которого используется в приемнике как признак начала кадра и как начало отсчета времени для определения моментов прихода отдельных битов в кадре. Затем приемник начинает процедуру приема и подсчета принимаемых битов. Данные поступают на вход RxD в последовательном коде, причем для получения достоверных значений битов в период стробирования производится несколько считываний состояния линии и решение принимается по мажоритарному принципу, например, при трехкратном считывании – два из трех. Процедура завершается с приходом стоп-бита занесением результата в буферный регистр приемника. Флаг завершения приема RI устанавливается в «1». Бит RI м.б. считан программно и генерируется запрос на прерывание, если прерывания от приемника разрешены. В подпрограмме прерывания байт данных из буферного регистра считывается в память МК. После копирования байта данных из сдвигового регистра в буферный приемник может сразу формировать новый байт данных последовательно поступающий на вход RxD. Необходимо, чтобы МК успел сосчитать данные в свою память до завершения формирования следующего байта в сдвиговом регистре.

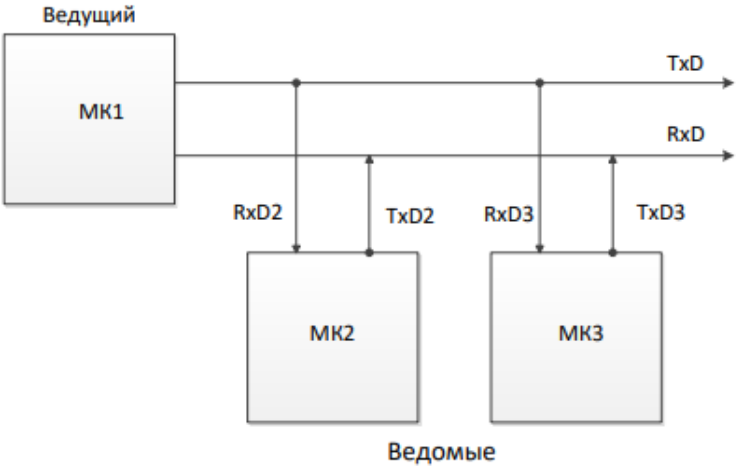
Если этого не произойдет, новый байт может затереть старый, и произойдет ошибка записи (в MSC-51). В МК фирмы Motorola имеется защита: устанавливается в «1» флаг ошибки (OR=1 Overrun), который может генерировать запрос на прерывание от приемника.

19. Типы кадров асинхронного обмена, предусмотренные модулем UART. Структурная схема локальной сети, осуществляющей связь микроконтроллеров с помощью UART. Принципы обмена между ведущим и ведомым микроконтроллерами. Структура пакета обмена.

Все модули типа UART предусматривают два кадра асинхронного обмена – 10-битовый и 11-битовый (рис.30).



Для обмена в многопроцессорной системе используется 11-битовый формат кадра. Каждый микроконтроллер в системе имеет свой сетевой адрес. При организации обмена передающий микроконтроллер (МК1) играет роль ведущего, а принимающие (МК2, МК3) - роль ведомых. Обмен между ведущим и ведомыми осуществляется пакетами, включающими не менее трех 11 битовых кадров. Первый кадр - адрес ведомого микроконтроллера. Признак адреса D8=1. Затем может посылаться несколько кадров данных, для обмена с ведомым микроконтроллером (разряд D8=0). Последний кадр должен содержать 11 нулевых битов, включая стоповый.



Порядок обмена.

1.До начала обмена в управляющем регистре должен быть установлен в единицу флаг разрешения обмена (SM2=1). Тогда кадр адреса

будет вызывать прерывание, а кадр данных - нет. Вид информации - адрес или данные определяется по значению бита D8.

2.В исходном состоянии приемники всех ведомых микроконтроллеров находятся в ожидании.

3. Ведущий посылает всем ведомым одновременно первый кадр пакета обмена, в котором D0 - D7 – адрес, D8=1.

4. Все ведомые принимают этот кадр, выходят из состояния ожидания, формируют запрос на прерывание и в подпрограмме обработки прерывания анализируют принятый адрес, сравнивая его с собственным сетевым.

5. При идентификации своего адреса ведомый микроконтроллер сбрасывает бит SM2 и читает данные от ведущего, сопровождаемые признаком D8=0. Остальные ведомые снова переходят в режим ожидания, оставляют свои биты SM2=1 и продолжают выполнять текущую программу. Данные от ведомого может принимать и ведущий.

6. Обмен завершается 11-битовым нулевым кадром. Для сброса в «0» стопового бита ведущий формирует сигнал «break». В ответ ведомый вновь переводит свой приемник модуля UART в состояние ожидания, устанавливая бит SM2=1. Сеть вновь готова к передаче следующего пакета.

Последовательный порт является источником 3-х видов внутренних прерываний:

1)Когда буфер приема полон и разрешено прерывание от приемника.

2)Когда буфер передачи пуст и разрешено прерывание от передатчика.

3)Прерывание по ошибке (когда в буфер приема поступает очередной символ до того, как процессор считывает предыдущий).

В некоторых семействах МК вектор прерывания UART один и для приема и для передачи, поэтому для определения источника прерывания в подпрограмме обслуживания производится программный опрос состояния флагов приема RI и передачи TI. В этом случае флаги RI и TI не сбрасываются автоматически при входе в подпрограмму обслуживания, а их сброс производится только после определения того, каким событием вызвано прерывание – приемом или передачей.

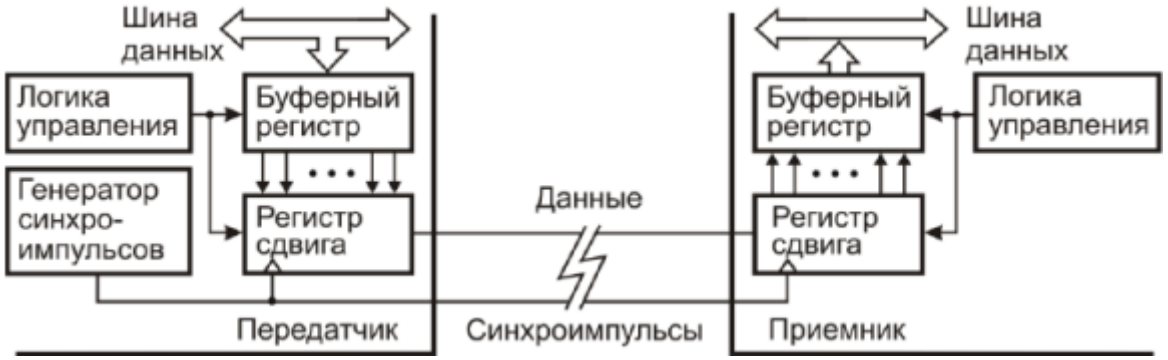
Последовательный порт может работать и в синхронном режиме

Способ синхронной передачи данных предполагает синхронизацию работы приемника и передатчика посредством включения тактовой

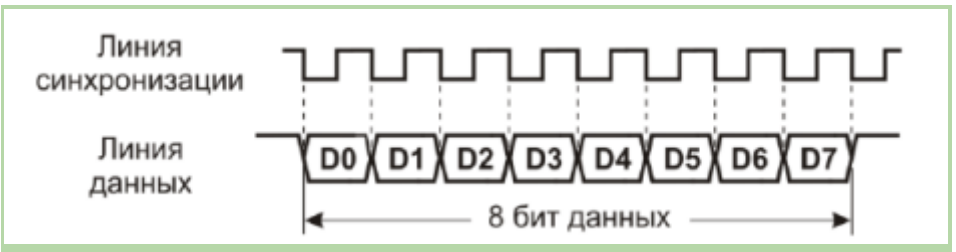
информации в передаваемый сигнал или, чаще всего, путем использования общей для источника и приемника линии синхронизации, импульсы на которой служат для выделения отдельных битов в канале.

Структура организации синхронной передачи данных показана на рис. 33.

Начало передачи инициируется занесением передаваемых данных в буферный регистр передатчика, затем передатчик загружает информацию в сдвиговый регистр, запускает генератор синхроимпульсов и счетчик. Синхроимпульсы передаются по линии TxD. С каждым синхронизирующим сигналом данные сдвигаются на одну позицию и поступают на линию данных RxD.



При этом каждый передаваемый бит данных сопровождается импульсом по линии синхронизации, информирующим приемник о наличии на линии данных информационного бита. Приемник состоит из регистра сдвига с последовательной записью, счетчика и логической схемы, которые управляются теми же синхросигналами. После того как счетчик зарегистрирует поступление необходимого количества синхросигналов (и, соответственно, бит данных), он инициирует параллельную передачу полученных данных из сдвигового регистра в буферный регистр приемника данных, откуда они могут быть считаны для дальнейшей обработки в прикладной программе.



20. Последовательный интерфейс SPI. Структурная схема сети. Обмен в стандарте SPI; обобщенные временные диаграммы.

SPI (Serial Peripheral Interface SPI bus— последовательный периферийный интерфейс, шина SPI)— последовательный синхронный стандарт передачи данных в режиме полного дуплекса, предназначенный для обеспечения простого и недорогого высокоскоростного сопряжения микроконтроллеров и периферии. Стандарт был предложен фирмой Motorola для связи микроконтроллера с периферийными устройствами микропроцессорной системы, которые располагаются на одной плате с микроконтроллером. Это могут быть простейшие устройства, такие как сдвиговые регистры, так и сложные интегральные схемы со встроенными управляющими контроллерами, например, ЦАП, ОЗУ, ЗУ типа FLASH, EEPROM.

В SPI всегда есть один ведущий и один/несколько ведомых. SPI является синхронным интерфейсом, в котором любая передача синхронизирована с общим тактовым сигналом, генерируемым ведущим устройством (процессором). Принимающая (ведомая) периферия синхронизирует получение битовой последовательности с тактовым сигналом. К одному последовательному периферийному интерфейсу ведущего устройства- микросхемы может присоединяться несколько микросхем. Ведущее устройство выбирает ведомое для передачи, активируя сигнал «выбор кристалла» на ведомой микросхеме. Периферия, не выбранная процессором, не принимает участия в передаче по SPI.

В SPI используются четыре цифровых сигнала:

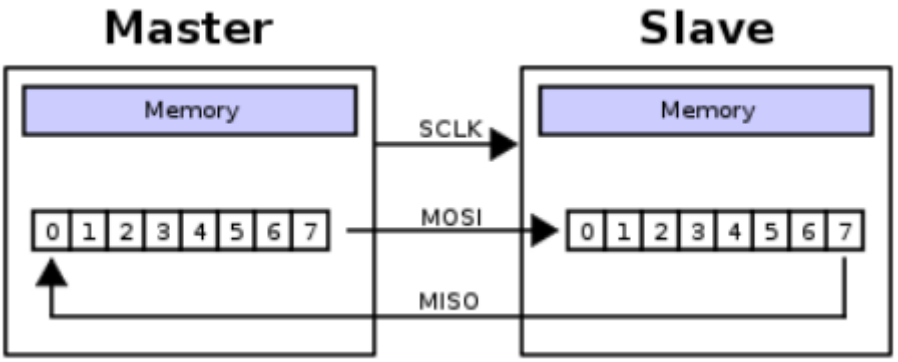
MOSI- линия передачи данных от ведущего к ведомому (Master Output Slave Input);

MISO- линия передачи от ведомого к ведущему (Master Input Slave Output);

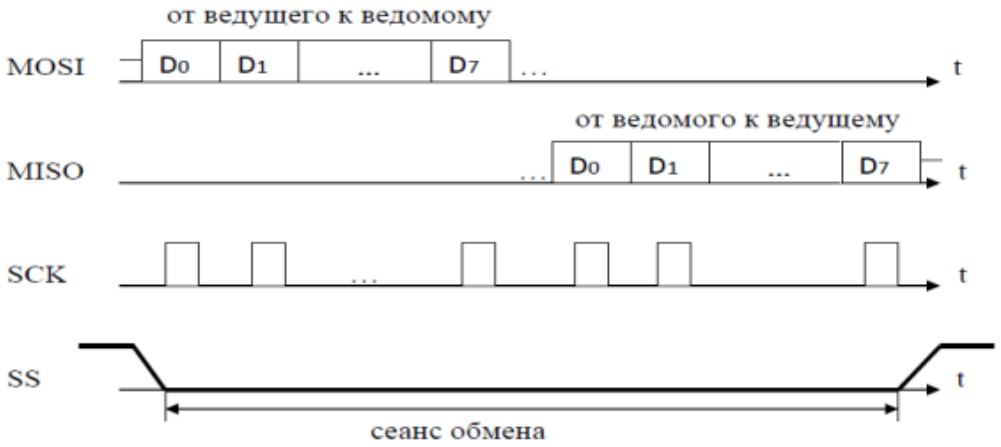
SCLK или SCK- линия стробирования данных (синхроимпульс);

SS1, SS2, SS3, SS или CS (Chip Select, Slave Select) - линии выбора ведомого устройства.

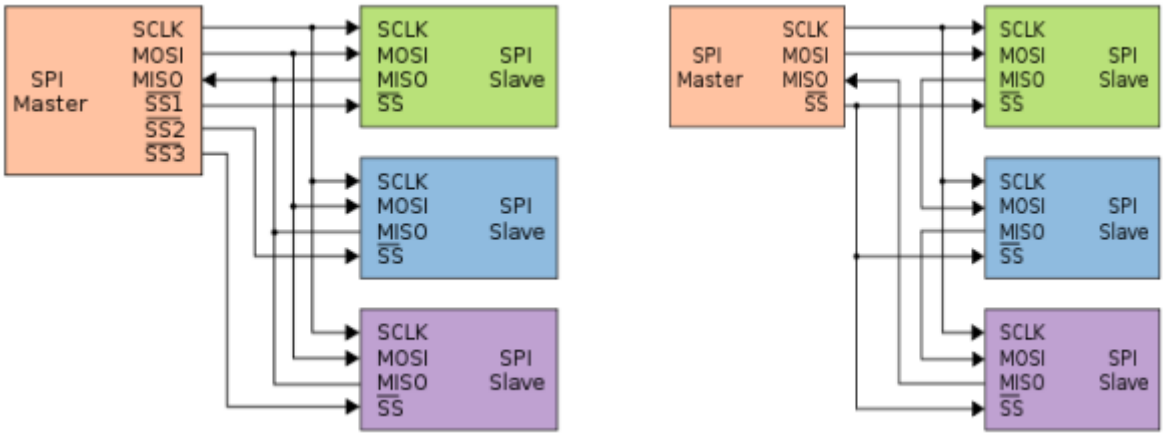
В ведущем режиме к выводу MOSI подключается выходная линия, а к MISO – входная, SCK – выход. В ведомом режиме выводы меняются ролями, SCK – вход.



Рассмотрим принцип обмена данными в стандарте SPI. Временная диаграмма этого процесса приведена на рис.37. Передача осуществляется пакетами. Длина пакета, как правило, составляет 1 байт (8 бит). Цикл связи всегда инициирует ведущее устройство установкой низкого уровня на выводе выбора подчиненного устройства (SS). Подлежащие передаче данные ведущее и ведомое устройства помещают в сдвиговые регистры.



Интерфейс SPI позволяет подключать к одному ведущему устройству несколько ведомых устройств, причем подключение может быть осуществлено несколькими способами. На рис. 38 приведены две микроконтроллерные системы, использующие модуль SPI: а) с радиальной структурой связи; б) с кольцевой структурой связи.

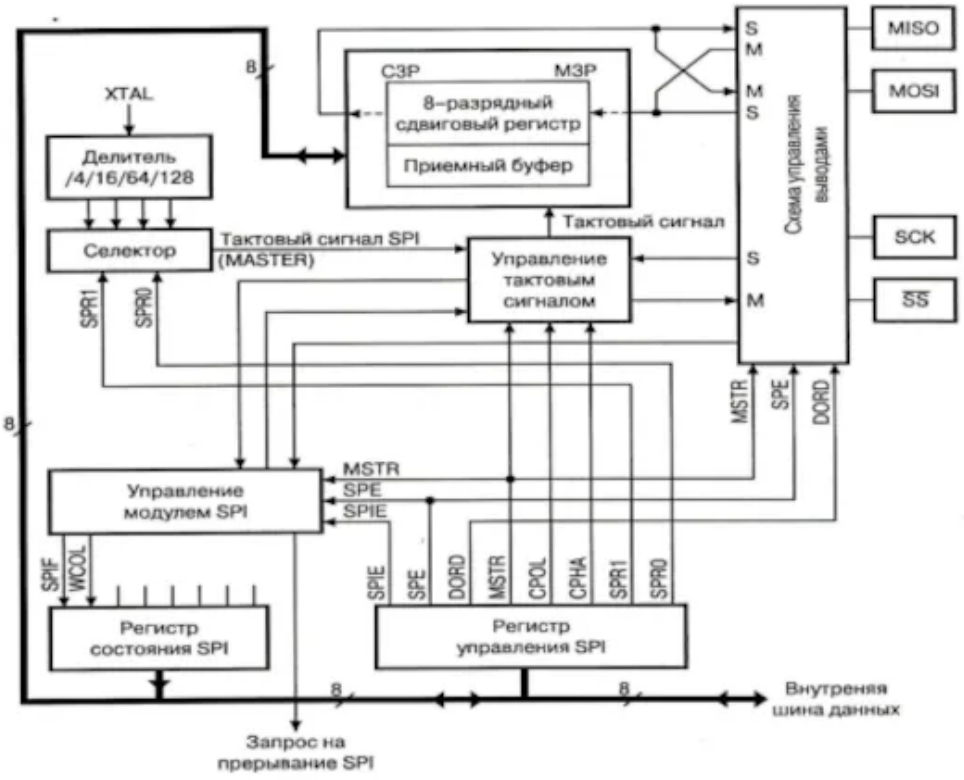


короткие вопросы

11. Какая архитектура процессора у микроконтроллеров семейства PIC?

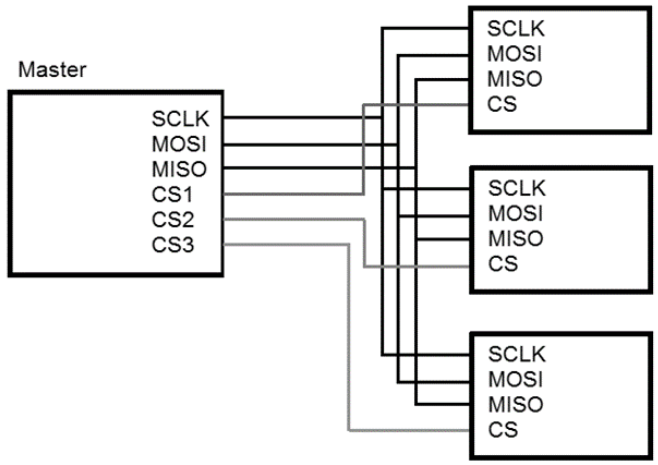
Микроконтроллеры PIC имеют RISC-архитектуру. RISC – сокращённый набор команд, используется также в процессорах для мобильных устройств.

12. Структурная схема интерфейса SPI.



21. Модуль последовательного синхронного интерфейса SPI. Структурная схема; выбор режимов работы. Последовательность обмена между ведущим и ведомым устройствами

SPI (Serial Peripheral Interface) последовательный синхронный стандарт передачи данных в режиме полного дуплекса, предназначенный для обеспечения простого и недорогого высокоскоростного сопряжения



Структура мастер и факинг слейвы

SPI обеспечивает связь между микроконтроллером (мастером) и одним или несколькими периферийными устройствами (рабами). Основные компоненты:

1. Микроконтроллер (Master): Микроконтроллер играет роль мастера в протоколе SPI. Он инициирует и контролирует передачу данных между собой и периферийными устройствами. Мастер управляет сигналами тактовой частоты (SCK), выборки данных (MISO), передачи данных (MOSI) и линии выбора устройства (SS) для каждого раба.

2. Периферийные устройства (Slaves): Периферийные устройства являются рабами в протоколе SPI. Они подключаются к мастеру и обмениваются данными по его командам. Каждое периферийное устройство имеет свою собственную линию выбора (SS), которая используется мастером для выбора конкретного раба во время передачи данных.

3. Линия выбора устройства (SS): Линии выбора устройства используются мастером для выбора конкретного раба перед передачей данных. Мастер активирует соответствующую линию выбора устройства, чтобы указать, с каким рабом он хочет обмениваться данными.

4. Сигналы SCK, MOSI и MISO: Сигнал SCK (Serial Clock) представляет собой тактовый сигнал, используемый для синхронизации передачи данных между мастером и рабами. Сигнал MOSI (Master Out Slave In) представляет собой линию передачи данных от мастера к рабу, а сигнал MISO (Master In Slave Out) представляет собой линию передачи данных от раба к мастеру.

5. Регистры данных и сдвиговые регистры: Микроконтроллер и периферийные устройства содержат регистры данных для временного хранения передаваемых и принимаемых данных. Они могут быть использованы для буферизации данных и последовательной передачи по линиям MOSI и MISO с использованием тактового сигнала SCK.

Сигналы тактирования, передачи и выбора устройства контролируются мастером, а периферийные устройства реагируют на команды мастера и передают данные обратно. Это позволяет эффективно обмениваться данными между микроконтроллером и периферийными устройствами, подключенными по интерфейсу SPI.

Режимы работы определяет способ передачи данных и синхронизацию между ведущим и ведомым устройствами. Основные режимы работы:

1. Режим 0 (CPOL = 0, CPHA = 0): В этом режиме линия SCK (тактовый сигнал) в состоянии покоя (CPOL = 0) имеет низкий уровень, а данные считываются на фронте сброса (CPHA = 0) тактового сигнала. Данные

передаются на фронте установки SCK и считываются на фронте сброса SCK.

2. Режим 1 (CPOL = 0, CPHA = 1): В этом режиме линия SCK в состоянии покоя имеет низкий уровень, а данные считываются на фронте установки SCK. Данные передаются на фронте сброса SCK и считываются на фронте установки SCK.

3. Режим 2 (CPOL = 1, CPHA = 0): В этом режиме линия SCK в состоянии покоя имеет высокий уровень, а данные считываются на фронте сброса SCK. Данные передаются на фронте установки SCK и считываются на фронте сброса SCK.

4. Режим 3 (CPOL = 1, CPHA = 1): В этом режиме линия SCK в состоянии покоя имеет высокий уровень, а данные считываются на фронте установки SCK. Данные передаются на фронте сброса SCK и считываются на фронте установки SCK.

CPOL (Clock Polarity) определяет уровень, который принимает линия SCK в состоянии покоя, а CPHA (Clock Phase) определяет момент считывания и передачи данных относительно тактового сигнала SCK.

Последовательность обмена между ведущим и ведомым

Ведущий (микроконтроллера)
Ведомый (периферийное устройство)

1. Микроконтроллер (ведущее устройство) активирует линию выбора устройства (SS) для выбранного ведомого устройства, устанавливая ее в низкий уровень.
2. Мастер генерирует тактовый сигнал (SCK), определяющий скорость передачи данных.
3. Мастер подает данные на линию MOSI (Master Out Slave In), которые он хочет передать ведомому устройству.

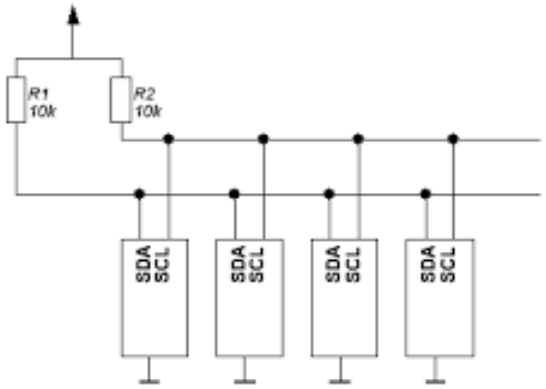
4. Ведомое устройство получает данные с линии MOSI и обрабатывает их соответствующим образом.
5. В то же время ведомое устройство может передавать данные мастеру на линии MISO (Master In Slave Out).
6. Мастер считывает данные с линии MISO.
7. Процесс передачи данных повторяется для каждого бита или байта, пока все данные не будут переданы или получены.
8. По завершении передачи данных мастер деактивирует линию выбора устройства (SS), устанавливая ее в высокий уровень.

Обмен данными в протоколе SPI осуществляется синхронно, с использованием тактового сигнала SCK.

Важно отметить, что каждое ведомое устройство в сети SPI должно иметь свою собственную линию выбора устройства (SS). Мастер активирует нужную линию выбора для выбора конкретного ведомого устройства перед передачей данных. Это позволяет мастеру общаться с несколькими ведомыми устройствами в одной сети SPI.

22. Двухпроводной последовательный интерфейс I2C: типовая структурная схема; основные свойства. Логическая схема подключения устройств к шине I2C. Какое устройство генерирует синхросигналы? Доминантный и рецессивный логические уровни на шине I2C. Временные диаграммы формирования сигналов «СТАРТ», «СТОП» и передачи данных для шины I2C.

IIC I2C Inter-Integrated Circuit последовательная асимметричная шина для связи между интегральными схемами внутри электронных приборов.



В структуре интерфейс I2C использует две линии для связи между устройствами: линию данных (SDA) и линию тактирования (SCL). Оба сигнала являются двунаправленными.

1. SDA (Serial Data Line): Линия данных используется для передачи информации между устройствами. Каждое устройство, подключенное к шине I2C, может выступать в роли передатчика или приемника данных. Линия SDA управляется только ведущим устройством (Master).

2. SCL (Serial Clock Line): Линия тактирования является сигналом тактирования, синхронизирующим передачу данных по линии SDA. Линия SCL также управляется только ведущим устройством.

3. I2C Bus: Шина I2C представляет собой физическую среду, по которой передаются сигналы SDA и SCL. Устройства, подключенные к шине, должны иметь уникальные адреса, чтобы быть идентифицированными.

Основные свойства:
1. Двухпроводная шина: I2C использует всего две двунаправленные линии для коммуникации между устройствами - линию данных (SDA) и линию тактового сигнала (SCL). Это делает его простым в реализации и экономичным по использованию пинов.

2. Мастер-ведомый протокол: I2C работает в режиме мастер-ведомый, где одно устройство выступает в роли мастера и контролирует обмен данными с одним или несколькими ведомыми устройствами. Мастер и ведомые устройства могут передавать и принимать данные друг от друга.

3. Адресация: Каждое устройство в сети I2C имеет свой уникальный адрес, который используется для идентификации устройства при передаче и получении данных. Это позволяет мастеру выбирать конкретное ведомое устройство для обмена данными.

4. Мультимастерская поддержка: I2C поддерживает концепцию мультимастерства, что означает, что несколько устройств могут выступать в роли мастера и конкурировать за доступ к шине. Встроенные механизмы арбитража и контроля конфликтов позволяют разрешать такие ситуации.

5. Синхронный протокол: I2C является синхронным протоколом, где передача данных осуществляется с использованием тактового сигнала SCL. Устройства синхронизируются по тактовому сигналу и передают данные в установленных временных интервалах.

6. Расширяемость: I2C поддерживает подключение множества устройств к одной шине, что позволяет создавать сложные сети, включающие мастера и несколько ведомых устройств.

7. Низкое энергопотребление: I2C работает на низком уровне энергопотребления, что делает его подходящим для использования в устройствах с ограниченными ресурсами энергии, таких как мобильные устройства и датчики.

8. Стандартизация: Протокол I2C является открытым стандартом и широко распространен во множестве устройств, что обеспечивает совместимость и у
Логическая схема подключения устройств к шине I2C.

Генерирует сигналы мастер-устройство. Оно инициирует передачу данных и контролирует синхронизацию всех устройств на шине I2C.

Мастер-устройство генерирует синхросигналы, устанавливая их уровни на линии тактового сигнала (SCL). Он определяет частоту и фазу синхросигналов, которые используются для синхронизации передачи данных между мастером и ведомыми устройствами.

Каждое ведомое устройство на шине I2C считывает и реагирует на синхросигналы, отправляемые мастером. Синхросигналы определяют моменты передачи и считывания битов данных, обеспечивая синхронность обмена данными между мастером и ведомыми устройствами.

Доминантный и рецессивный логические уровни на шине I2C.
Эти состояния определяются уровнем напряжения на линии данных (SDA) и линии тактового сигнала (SCL) во время передачи данных.

1. Доминантный уровень low соответствует низкому напряжению (обычно 0 В) на линиях SDA и SCL. Доминантный уровень используется для передачи "0" или сигнализации активности на шине.

2. Рецессивный уровень high: Рецессивный уровень соответствует высокому напряжению (обычно Vcc) на линиях SDA и SCL. Рецессивный уровень используется для передачи "1" или для обозначения неактивности на шине.

Мастер-устройство и ведомые устройства на шине I2C могут переключаться между доминантным и рецессивным состояниями в зависимости от их активности и управления передачей данных.

Во время передачи данных по шине I2C, мастер-устройство и ведомые устройства могут устанавливать и читать логические уровни на линиях SDA и SCL. Мастер-устройство инициирует коммуникацию и передает информацию, устанавливая логические уровни на линии SDA в соответствии с передаваемыми данными. Ведомые устройства отвечают, поддерживая и читая установленные логические уровни на линиях SDA и SCL.

Важно отметить, что в протоколе I2C используется открытый коллектор или открытый сток для реализации линии SDA. Это означает, что ведомые устройства могут только пассивно подтягивать линию SDA к высокому уровню, а мастер-устройство активно выводит линию на низкий уровень. Это позволяет мастеру контролировать линию SDA и обнаруживать конфликты с другими устройствами на шине.

Временные диаграммы формирования сигналов «СТАРТ», «СТОП» и передачи данных для шины I2C.

Надеюсь все то

23. Форматы адреса для интерфейса I2C. Структура 7-битового адреса. Обмен данными на логическом уровне в протоколе I2C: от ведущего к ведомому; от ведомого к ведущему. Структура передаваемых пакетов с 7-битовым адресом.

Формат адреса зависит от режима работы и используемого подмножества адресов.

1. 7-битный адрес:
В стандартном режиме протокола I2C используется 7-битный адрес, что позволяет адресовать до 128 ведомых устройств, т.е. от 0 до 127 (0x00 до 0x7F). Наиболее распространенный формат адреса состоит из 7 бит, где младший бит указывает режим работы (запись или чтение). Первые 6 бит представляют уникальный адрес ведомого устройства.

2. 10-битный адрес:
В режиме расширенного адресации протокола I2C используется 10-битный адрес, позволяющий адресовать до 1024 ведомых устройств. Формат 10-битного адреса состоит из двух байтов, где первый байт содержит биты 1111 0xx0 (биты F, D и A), а второй байт содержит оставшиеся биты адреса.

3. Резервированные адреса:
В протоколе I2C существуют некоторые зарезервированные адреса, которые используются для специальных целей. Например, адрес 0x00 может использоваться для общей шины (General Call), а адрес 0x78 или 0x7A может использоваться для адресации всех ведомых устройств на шине (Broadcast).

Каждое ведомое устройство в системе должно иметь уникальный адрес, чтобы мастер-устройство могло точно идентифицировать его и обращаться к нему. Формат адреса и его использование зависят от конкретной реализации и требований системы

Структура его пакета и обмен данными

1. Старт-бит (START):
Перед началом передачи данных мастер-устройство отправляет сигнал "START", который состоит из перехода линии данных (SDA) с высокого уровня (high) на низкий уровень (low), при условии, что линия тактового сигнала (SCL) находится на высоком уровне (high). Старт-бит указывает на начало передачи данных и готовность мастера к коммуникации. Ведомое устройство готово принять данные после обнаружения сигнала "START".

2. 7-битный адрес ведомого устройства:
После сигнала "START" мастер-устройство отправляет 7-битный адрес ведомого устройства. Младший бит адреса (бит 0) указывает на режим работы (запись или чтение).

3. Бит подтверждения (ACK/NACK):
После отправки адреса мастер-устройство ожидает бит подтверждения от ведомого устройства. Если ведомое устройство обнаруживает свой адрес и готово принимать данные, оно отправляет бит подтверждения (ACK), который представляет собой переход линии данных (SDA) с высокого уровня (high) на низкий уровень (low). Если ведомое устройство не готово принимать данные или не обнаруживает свой адрес, оно отправляет бит неподтверждения (NACK), который представляет собой переход линии данных (SDA) с высокого уровня (high) на низкий уровень (low).

4. Передача данных:

Если нужное ведомое устройство отправляет бит подтверждения (ACK), оно может начать передачу данных мастеру. Данные передаются по линии данных (SDA) с использованием тактового сигнала (SCL). Каждый байт данных передается последовательно, начиная со старшего бита (MSB) до младшего бита (LSB).

5. Бит подтверждения (ACK/NACK):

После передачи каждого байта данных, мастер-устройство должно отправить бит подтверждения (ACK) или неподтверждения (NACK) ведомому устройству. Ведомое устройство должно обнаружить состояние линии данных (SDA) после каждого байта данных, чтобы определить, отправляет ли мастер-устройство ACK или NACK. Если ведомое устройство обнаруживает ACK, оно продолжает передавать следующий байт данных. Если ведомое устройство обнаруживает NACK, оно прекращает передачу данных.

6. Стоп-бит (STOP):

По окончании передачи данных, мастер-устройство отправляет сигнал "STOP", переводя линию данных (SDA) с низкого уровня (low) на высокий уровень (high), при условии, что линия тактового сигнала (SCL) находится на высоком уровне (high). Стоп-бит указывает на конец передачи данных и освобождение шины для других устройств.

24. Синхронизация в протоколе I2C. Арбитраж на шине между несколькими ведущими устройствами.

Протокол I2C использует две линии для синхронизации: линию тактового сигнала (SCL) и линию данных (SDA).

Линия тактового сигнала (SCL) определяет тактовую частоту и служит для синхронизации передачи данных между устройствами. Мастер-устройство генерирует тактовые импульсы.

Линия данных (SDA) используется для передачи информации между ведущим и ведомым устройствами.

Сигналы "START" и "STOP", отправляемые мастер-устройством, также служат для синхронизации обмена данными. Сигнал "START" указывает на начало передачи данных, а сигнал "STOP" указывает на конец передачи данных. Оба сигнала синхронизируют устройства на шине и помогают определить границы передаваемых данных.

Арбитраж на шине I2C возникает, когда на шине присутствуют несколько ведущих устройств (master). В протоколе I2C может быть только одно активное ведущее устройство, которое контролирует обмен данными на шине. Когда несколько ведущих устройств одновременно пытаются взаимодействовать с шиной, возникает конфликт.

Протокол I2C решает конфликты с помощью арбитража. Арбитраж осуществляется на основе приоритетности ведущих устройств. Каждое ведущее устройство в момент своего запроса на доступ к шине контролирует уровень линии данных (SDA). Если ведущее устройство обнаруживает, что уровень на линии SDA не соответствует тому, что оно отправляет, то оно понимает, что произошел конфликт с другим ведущим устройством.

Арбитраж в I2C основан на логическом уровне "0" и "1" на линии данных (SDA). Все устройства на шине I2C могут устанавливать логический "0" на линии SDA, но только одно устройство может устанавливать логический "1". Из-за этого свойства, устройство с логическим "1" всегда будет доминантным в арбитражной ситуации.

При возникновении конфликта и одновременной попытке доступа к шине нескольких ведущих устройств, каждое ведущее устройство следит за уровнем линии

SDA, чтобы определить, является ли оно доминантным или рецессивным в данной ситуации. Устройство, которое обнаруживает, что его уровень на линии SDA не соответствует уровню, который оно отправляло, понимает, что оно не является доминантным и должно освободить шину.

25. CAN-протокол. Области применения. Типовая структурная схема. Принцип передачи сообщений; типы сообщений (фреймы). Формат фрейма данных, основные поля.
Ответ: CAN (Controller Area Network) — это серийный асинхронный протокол передачи данных, разработанный для обеспечения надежной и эффективной связи между устройствами в автомобильной и промышленной среде. Области применения:

1. Автомобильная промышленность: CAN-протокол широко применяется в автомобильной промышленности для связи и взаимодействия между различными электронными устройствами в автомобиле. Он используется для передачи данных между управляющими блоками, такими как двигатель, трансмиссия, антиблокировочная система (ABS), электронная система управления стабилизацией (ESP) и другими системами автомобиля.

2. Промышленная автоматизация: CAN-протокол применяется в системах промышленной автоматизации, где требуется связь между различными устройствами и контроллерами. Он используется для мониторинга и управления различными процессами, такими как системы управления производством, контрольные системы, системы безопасности и др.

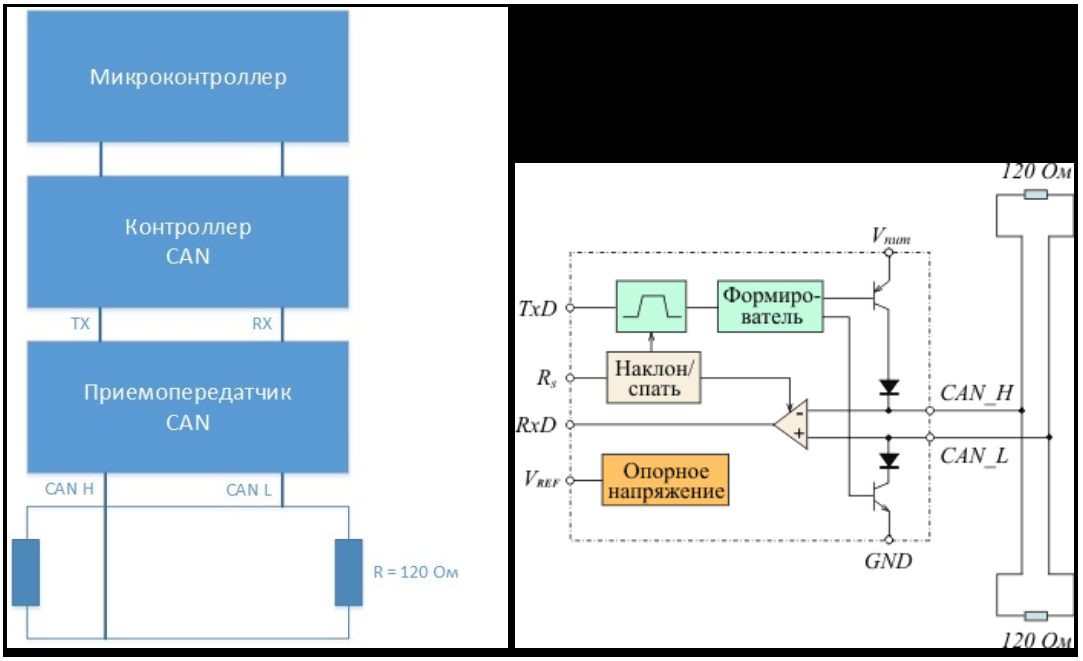
3. Медицинская техника: В области медицинской техники CAN-протокол используется для связи и обмена данными между различными медицинскими устройствами,

например, в системах мониторинга пациентов, медицинской аппаратуре, сканерах и других медицинских устройствах.

4. Транспортные системы: CAN-протокол применяется в системах управления транспортными средствами, такими как поезда, трамваи, метро и авиационные системы. Он используется для связи между различными устройствами в этих системах, обеспечивая передачу данных, контроль и управление.

5. Энергетика и солнечная энергетика: CAN-протокол применяется в системах управления энергетическими установками, такими как солнечные электростанции и распределительные сети. Он обеспечивает связь между различными устройствами для контроля и мониторинга работы систем, передачи данных о производстве энергии и управлении энергетическими ресурсами.

Структурная схема



ТРАНСИВЕР CAN

Интерфейс CAN использует две проводные линии для связи между устройствами: высокоскоростную линию

(CANH) и низкоскоростную линию (CANL). Обе линии используются для передачи данных в формате дифференциальных сигналов.

Основные компоненты структурной схемы CAN:

1. CANH (CAN High): Высокоскоростная линия CAN используется для передачи положительного дифференциального сигнала. Этот сигнал является активным при высоком уровне напряжения и пассивным при низком уровне напряжения.

2. CANL (CAN Low): Низкоскоростная линия CAN используется для передачи отрицательного дифференциального сигнала. Этот сигнал является активным при низком уровне напряжения и пассивным при высоком уровне напряжения.

3. CAN Bus: Шина CAN представляет собой физическую среду, по которой передаются дифференциальные сигналы CANH и CANL. Все устройства на шине CAN должны быть подключены в соответствии с определенными электрическими требованиями и иметь подходящую терминированную схему.

Протокол CAN включает в себя механизмы для управления доступом к шине, определения приоритетов сообщений и обнаружения ошибок. Он позволяет устройствам обмениваться сообщениями в реальном времени и обеспечивает высокую надежность передачи данных.

Принципы передачи сообщений

Протокол CAN (Controller Area Network) использует принцип передачи сообщений по шине с общим доступом, известный как "шина с приоритетом и арбитражем по биту". Этот принцип обеспечивает эффективную и надежную передачу данных в сети CAN.

Основные принципы передачи сообщений:
1. Шина с приоритетом: В сети CAN каждое устройство имеет свой уникальный идентификатор сообщения. Более низкие значения идентификаторов сообщений имеют более высокий приоритет. При передаче сообщения на шину CAN, устройства сравнивают идентификаторы сообщений и устраивают арбитраж, чтобы определить, какое устройство имеет право передать сообщение.

2. Арбитраж по биту: Арбитраж в протоколе CAN основан на сравнении битов данных, передаваемых на шину. Каждое устройство передает биты данных на шину одновременно. Во время передачи каждое устройство сравнивает свой передаваемый бит данных с битом, который фактически наблюдается на шине. Если наблюдаемый бит отличается от отправляемого, устройство понимает, что есть другое устройство с более высоким приоритетом, и прекращает передачу. Это позволяет устройству с более высоким приоритетом успешно передать свои данные.

3. Нон-доминирующие и доминирующие биты: В протоколе CAN используется метод доминирования/недоминирования. Если устройство наблюдает логический ноль (0) на шине, оно выводит недоминирующий бит. Если же устройство наблюдает логическую единицу (1), оно выводит доминирующий бит. В случае конфликта передачи, доминирующий бит имеет приоритет над недоминирующим битом. Это позволяет арбитражу правильно определить устройство с более высоким приоритетом и предоставить ему возможность передать сообщение.

4. Приоритетное включение: Если два или более устройств начинают передачу одновременно, то после определения победителя арбитража другие устройства переходят в режим приема. Приоритетное включение

позволяет устройствам с более низким приоритетом ожидать, пока шина станет свободной, и повторить попытку передачи данных.

Эти принципы передачи сообщений в протоколе CAN обеспечивают эффективную и надежную коммуникацию между устройствами в сети, позволяя передавать данные согласованно и с учетом приоритетов.

Типы сообщений (фреймов)

1. Data Frame (DF) - Фрейм данных: Data Frame используется для передачи фактических данных между устройствами в сети CAN. Он содержит идентификатор сообщения (ID), данные (пакет данных) и другую информацию, такую как длина данных, флаги ошибок и контрольная сумма. Data Frame может быть отправлен от одного устройства (источника) к одному или нескольким устройствам-получателям.

2. Remote Frame (RF) - Фрейм запроса: Remote Frame используется для запроса данных от устройства-источника к устройству-получателю. Он содержит идентификатор сообщения (ID) и флаг, указывающий, что это запрос на данные. Устройство-получатель может затем отправить Data Frame в ответ на запрос Remote Frame с необходимыми данными.

3. Error Frame (EF) - Фрейм ошибки: Error Frame используется для передачи информации об ошибках в сети CAN. Он указывает на наличие ошибок в передаче данных или в работе устройства. Error Frame содержит специальный идентификатор (ID), который обозначает, что это фрейм ошибки, и код ошибки, который указывает на тип и характер ошибки.

4. Overload Frame (OF) - Фрейм перегрузки: Overload Frame используется для указания на перегрузку сети CAN. Он

отправляется, когда устройство-источник не может обработать все сообщения, поступающие на шину, из-за ограниченных ресурсов. Overload Frame содержит специальный идентификатор (ID), который обозначает, что это фрейм перегрузки.

Эти четыре типа фреймов позволяют эффективно передавать данные, запрашивать данные, обрабатывать ошибки и обрабатывать перегрузки в сети CAN, обеспечивая надежную коммуникацию между устройствами.

Основные поля фреймов

1. Start-of-Frame (SOF): Поле Start-of-Frame указывает начало фрейма и помогает синхронизировать устройства в сети CAN. Он состоит из единственного бита, имеющего значение 0.

2. Arbitration Field (ID): Arbitration Field содержит идентификатор сообщения (ID), который определяет приоритет сообщения на шине CAN. ID может быть стандартным (11 бит) или расширенным (29 бит). Более низкий ID обладает более высоким приоритетом. Это поле также включает бит RTR (Remote Transmission Request), который указывает, является ли фрейм запросом на передачу данных.

3. Control Field: Control Field содержит различные управляющие биты, которые контролируют передачу

данных. Он включает длину данных (DLC - Data Length Code), указывающую количество байтов данных, которые содержатся в поле Data Field. Также в Control Field может включаться бит, указывающий, что фрейм является фреймом перегрузки (Overload Frame).

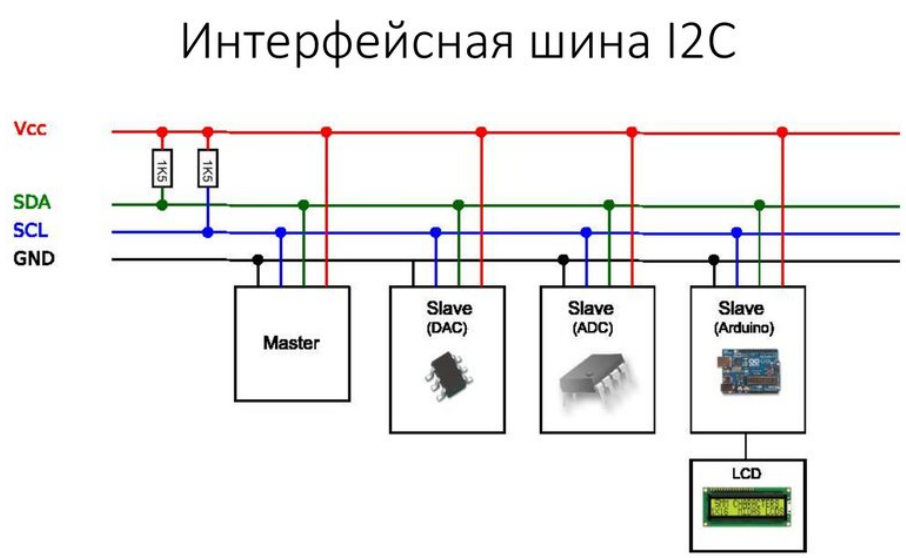
4. Data Field: Data Field содержит фактические данные, которые передаются между устройствами в сети CAN. Размер Data Field может варьироваться в зависимости от значения DLC в Control Field.

5. CRC Field: CRC (Cyclic Redundancy Check) Field содержит контрольную сумму, которая используется для обнаружения и исправления ошибок в данных. CRC вычисляется на основе содержимого фрейма и проверяется устройством-получателем для проверки целостности данных.

6. Acknowledgement Field (ACK): Acknowledgement Field состоит из бита ACK Delimiter, который указывает на успешное или неуспешное принятие сообщения. Успешное принятие сообщения подтверждается устройством-получателем путем вывода доминирующего (0) бита ACK.

7. End-of-Frame (EOF): Поле End-of-Frame обозначает конец фрейма и служит для завершения передачи данных. Оно состоит из нескольких последовательных битов с высоким уровнем (1).

короткие вопросы
13. Структурная схема интерфейса I2C.



14. Структурная схема CAN- интерфейса.

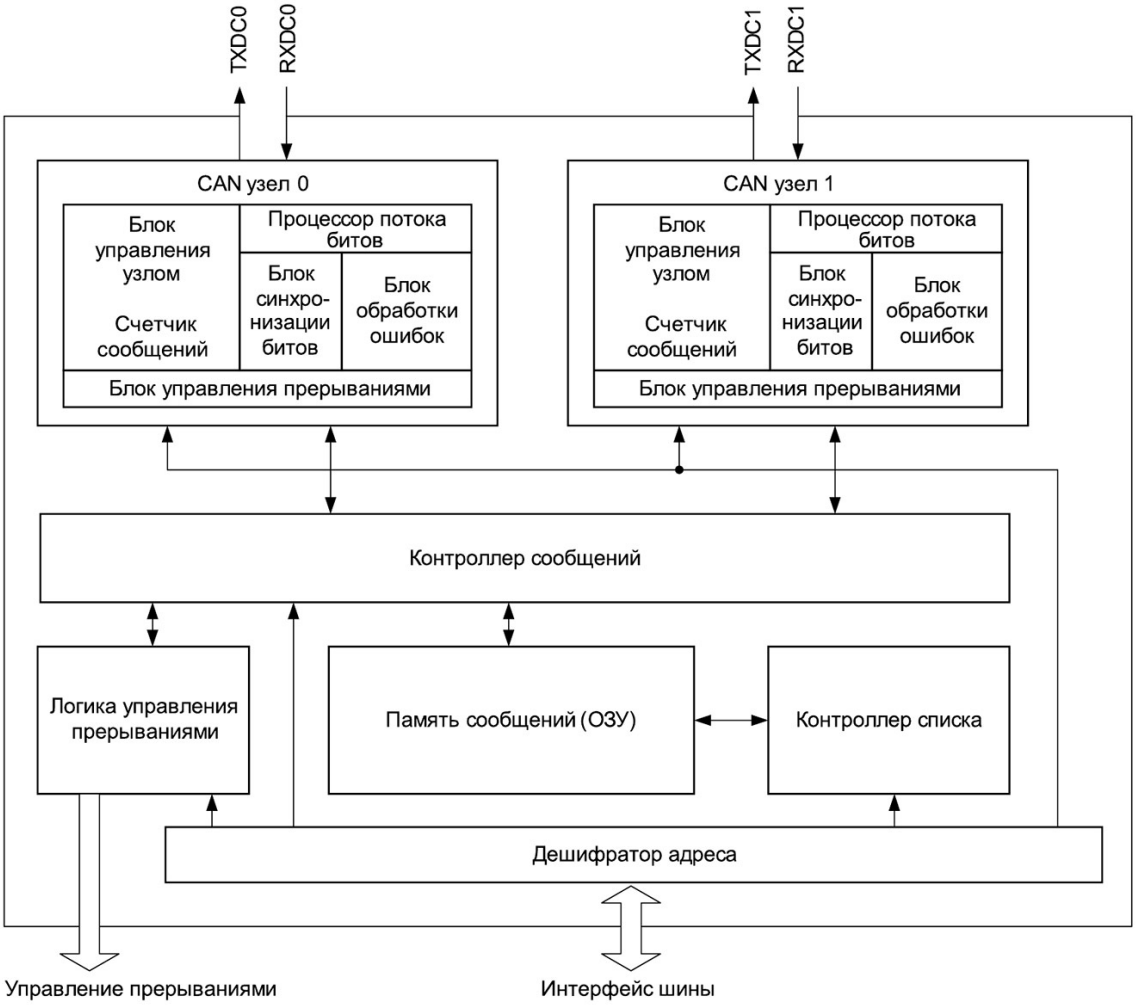


Рис. 3. **Структурная схема блока CAN**

16. Какие виды памяти существуют в микроконтроллерах семейства PIC?

ПЕРВЫЕ ТРИ ТОЧНО ЕСТЬ

1. Программная память (Program Memory): Это память, в которой хранится программа микроконтроллера. Обычно представлена флэш-памятью, которая позволяет записывать и стирать программу в микроконтроллере.

2. Оперативная память (RAM - Random Access Memory): Оперативная память используется для временного хранения данных и переменных во время выполнения программы. Обычно представлена в виде статической RAM (SRAM) или энергонезависимой RAM (EEPROM) для сохранения данных при отключении питания.

3. EEPROM (Electrically Erasable Programmable Read-Only Memory): EEPROM представляет собой тип памяти, который можно электрически стирать и записывать. EEPROM используется для хранения постоянных данных, которые нужно сохранить даже при отключении питания, например, конфигурационные настройки и калибровочные данные.

4. Буферные регистры (Buffer Registers): Некоторые микроконтроллеры PIC имеют специальные буферные регистры, которые используются для временного хранения данных перед их передачей или обработкой. Это может быть регистр данных, регистр статуса и другие специализированные регистры.

5. Регистры периферийных устройств (Peripheral Registers): Микроконтроллеры PIC имеют различные периферийные устройства, такие как таймеры, счетчики, преобразователи АЦП/ЦАП, порты ввода/вывода и другие. Каждое периферийное устройство обычно имеет свои регистры, которые используются для управления и обмена данными с этими устройствами.

15. Какие бывают порты ввода-вывода в микроконтроллерах?

1. GPIO (General Purpose Input/Output): Это самый распространенный тип порта ввода/вывода. GPIO-порты могут быть настроены как входы или выходы, и их состояние может быть прочитано или изменено программно. Они используются для подключения и управления внешними устройствами, такими как кнопки, светодиоды, датчики и т.д.

2. Аналоговые порты (Analog Ports): Некоторые микроконтроллеры имеют встроенные аналоговые порты, которые позволяют считывать аналоговые значения напряжения от внешних источников, таких как датчики температуры, давления и т.д. Эти порты обычно связаны с встроенными преобразователями аналогового-цифрового (ADC) и цифро-аналогового (DAC).

3. Серийные порты (Serial Ports): Микроконтроллеры могут также иметь специализированные порты для серийной коммуникации, такие как UART (Universal Asynchronous Receiver-Transmitter), SPI (Serial Peripheral Interface) и I2C (Inter-Integrated Circuit). Эти порты предназначены для обмена данными с другими устройствами посредством последовательного протокола передачи данных.

4. Порты с прерываниями (Interrupt Ports): Некоторые микроконтроллеры имеют порты, которые могут генерировать прерывания, когда происходит определенное событие, такое как изменение состояния входа или прием данных по серийному порту. Это позволяет микроконтроллеру немедленно отреагировать на важные события, не тратя время на постоянное опрос состояния портов.

На всякий выпишу понятие прерывания из видео

Прерывание - это остановка основной логики программы по какому-либо событию, запоминание всех состояний областей памяти на момент остановки, обработка подпрограммы прерывания, которую мы вызываем, возврат в точку выполнения основной логики программы и продолжение работы контроллера.

26. CAN-протокол. Основные характеристики. Проблемы коллизий. Доминантный и рецессивный логические уровни на шине CAN

Свойства CAN протокола

1. На рынке CAN присутствует в двух версиях: версия А задает 11- битную идентификацию сообщений (т. е. в системе может быть 2048 сообщений), версия В — 29-битную (536 млн. сообщений). Отметим, что версия В, часто именуемая FullCAN, все больше вытесняет версию А, которую называют также BasicCAN. Сеть CAN состоит из узлов с собственными тактовыми генераторами.
2. Протокол CAN использует оригинальную систему адресации сообщений. Любой узел сети CAN посылает сообщение всем системам, подсоединенным к шине.
3. Протокол CAN обладает развитой системой обнаружения и сигнализации ошибок. Для этих целей используется поразрядный контроль, прямое заполнение битового потока, проверка пакета сообщения CRCполиномом, контроль формы пакета сообщений, подтверждение правильного приема пакета данных. Хеммингoвый интервал d=6. Общая вероятность необнаруженной ошибки 4.7x10-11 .
4. Система арбитража протокола CAN исключает потерю информации и времени при "столкновениях" на шине.
5. Интерфейс с применением протокола CAN легко адаптируется к физической среде передачи информации. Это может быть радиоканал, оптоволокно, просто открытый коллектор и т.п. Но на практике под

CANсетью обычно подразумевается сеть топологии «шина» с физическим уровнем в виде дифференциальной пары, определённым в стандарте ISO 11898.
Коллизии: Поле арбитража CAN-кадра используется для разрешения коллизий доступа к шине методом недеструктивного арбитража. Суть метода заключается в следующем. В случае, когда несколько контроллеров начинают одновременную передачу CAN кадра в сеть, каждый из них сравнивает, бит, который собирается передать на шину с битом, который пытается передать на шину конкурирующий контроллер. Если значения этих битов равны, оба контроллера передают следующий бит. И так происходит до тех пор, пока значения передаваемых битов не окажутся различными.
Доминант: Логический ноль - называется доминантным битом, а логическая единица - рецессивным. Эти названия отражают приоритет логической единицы и нуля на шине CAN. При одновременной передаче в шину лог. нуля и единицы, на шине будет зарегистрирован только логический ноль (доминантный сигнал), а логическая единица будет подавлена (рецессивный сигнал).
27. Методы обнаружения ошибок в CAN-протоколе. Что такое bit staffing?
CAN протокол определяет пять способов обнаружения ошибок в сети:

- Bit monitoring
- ACKnowledgement Check
- Bit stuffing
- CRC Check
- Frame check

Разрядная ошибка Bit monitoring появляется, когда передатчик сравнивает уровень на шине с уровнем, который должен передаваться, и 5 обнаруживает их неравенство. При этом обнаружение активного бита, когда передается пассивный бит, не выдает ошибку в

течение передачи поля арбитража, поля ACK Slot или флажка пассивной ошибки.

Ошибка подтверждения ACKnowledgement Check - возникает, когда передатчик определяет, что сообщение не было подтверждено. Слот подтверждения существует внутри фреймов данных и удаленных фреймов. Внутри этого слота все приемные узлы, независимо от того, являются они пунктом назначения или нет, должны подтвердить получение сообщения.

Ошибка заполнения Bit stuffing появляется, когда узел обнаруживает шесть (6) последовательных битов одного и того же значения. В процессе нормальной работы, когда передатчик обнаруживает, что он послал пять (5) последовательных битов одного и того же значения, он заполняет следующий бит противоположным значением (это называется заполнением бита). Все приемники удаляют заполненные биты до вычисления CRC (контрольного кода). Таким образом, когда узел обнаруживает шесть (6) последовательных битов того же значения, возникает ошибка заполнения.

CRC-ошибка CRC Check ошибка контр. суммы появляется, когда CRC-значение (контрольный код) не соответствует значению, сгенерированному передатчиком. Каждый фрейм содержит поле контрольного кода, которое инициализировано передатчиком. Приемники вычисляют CRC и сравнивают его со значением, сгенерированным передатчиком. Если эти два значения не тождественны, то имеет место CRCошибка.

Ошибка формы Frame Check возникает, когда недопустимое разрядное значение обнаружено в области, в которую должно быть передано предопределенное значение. В CAN-протоколе существуют некоторые предопределенные разрядные значения, которые должны быть переданы в определенных местах. Если недопустимое разрядное значение обнаружено в одной из этих областей, имеет место ошибка формы

28. CAN-протокол: механизм ограничения ошибок, счетчики ошибок. Фильтрация сообщений в CAN-протоколе.

Механизм ограничения ошибок (Error confinement). Каждый узел сети CAN, во время работы пытается обнаружить одну из пяти возможных ошибок. Если ошибка обнаружена, узел передает в сеть Error Frame, разрушая тем самым весь текущий трафик сети (передачу и прием текущего сообщения). Все остальные узлы обнаруживают Error Frame и принимают соответствующие действия (сбрасывают принятое сообщение). Кроме того, каждый узел ведет два счетчика ошибок: **Transmit Error Counter** (счетчик ошибок передачи) и **Receive Error Counter** (счетчик ошибок приема). Эти счетчики увеличиваются или уменьшаются в соответствие с несколькими правилами. Сами правила управления счетчиками ошибок достаточно сложны, но сводятся к простому принципу. Ошибка передачи приводит к увеличению **Transmit Error** счетчика на 8, ошибка приема увеличивает счетчик **Receive Error** на 1, любая корректная передача/прием сообщения уменьшают соответствующий счетчик на Эти правила приводят к тому, что счетчик ошибок передачи передающего узла увеличивается быстрее, чем счетчик ошибок приема принимающих узлов. Это правило соответствует предположению о большой вероятности того, что источником ошибок является передающий узел.

Каждый узел CAN сети может находиться в одном из трех состояний. Когда узел стартует, он находится в состоянии **Error Active.** Когда, значение хотя бы одного из двух счетчиков ошибок превышает предел 127, узел переходит в состояние **Error Passive**. Когда значение хотя бы одного из двух счетчиков превышает предел 255, узел переходит в состояние **Bus Off.**

Узел, находящийся в состоянии Error Active, в случае обнаружения ошибки на шине передает в сеть Active Error Flags. Active Error Flags состоит из 6 доминантных

бит, поэтому все узлы его регистрируют. Узел в состоянии Passive Error передает в сеть Passive Error Flags при обнаружении ошибки в сети. Passive Error Flags состоит из 6 рецессивных бит, поэтому остальные узлы сети его не замечают, и Passive Error Flags лишь приводит к увеличению Error счетчика узла. Узел в состоянии Bus Off ничего не передает в сеть (не только Error кадры, но вообще никакие другие).

Фильтрация сообщений в CAN-протоколе. CAN — это протокол, ориентированный на использование в условиях помех. Различные сообщения, передающиеся по сети, имеют идентификатор, и каждая станция решает, основываясь на этом идентификаторе, получать или нет это сообщение. Этот идентификатор определен в поле идентификатора CAN-фрейма.

В CAN имеется аппаратная реализация фильтрации сообщений. У каждого приемника имеется свой адрес, который устанавливается в самом приемнике путем настройки входных фильтров.

Входные фильтры представляют собой решета, или идентификационные экраны. Любое сообщение, которое проходит через входные фильтры, должно быть обработано процессором обслуживания CAN контроллера. Существуют следующие два типа входных фильтров:

- **фиксированные** — фильтры, которые требуют, чтобы биты соответствовали точно один к одному (one-for-one).

- **Mask-and-Match (маскируемые)** — фильтры, которые применяют маску к полю идентификатора, прежде чем он сравнивается с приемным регистром кода. Например, на рисунке ниже регистр маски сконфигурирован так, что полученные биты 10-6 идентификатора должны соответствовать битам 10-6 в приемном регистре кода. В этом примере биты 10-6 идентификатора должны быть установлены в

11110, а остальные не имеют значения. Если биты 10-6 установлены в 11110, то эти сообщения принимаются независимо от значений битов 5-0.

Фильтрация типа Mask-and-Match											
10	9	8	7	6	5	4	3	2	1	0	Поразрядное значение идентификатора
1	1	1	1	0	0	0	0	1	1	1	Принятое значение идентификатора
mm	mm	mm	mm	mm	X	X	X	X	X	X	Регистр маски
1	1	1	1	0	0	0	0	0	0	0	Регистр кода после фильтрации
Принятое сообщение											

mm – код маски
x – произвольный код («1» или «0»)

29. PIC-микроконтроллеры фирмы MICROCHIP. Семейства PIC микроконтроллеров; характеристики процессоров.

Микроконтроллер спроектирован по Гарвардской архитектуре. В своей структуре содержит 64 кБ памяти программ и почти 4 кБ памяти данных, содержит цифро-аналоговый преобразователь ЦАП, три 8-битных таймера и четыре 16-битных таймера. Имеются компараторы, модули связи SPI, I2C. Имеется 36 модулей общего назначения и 30 аналоговых входов.

Высокая скорость выполнения команд в PIC -контроллерах достигается за счет использования двухшинной гарвардской архитектуры вместо традиционной одношинной фон-неймановской.

Гарвардская архитектура основывается на наборе регистров с разделенными шинами и адресными пространствами для команд и данных. Все ресурсы микроконтроллера, такие как порты ввода/вывода, ячейки памяти и таймер, представляют собой физически реализованные аппаратные **регистры** .

Микроконтроллеры PIC содержат RISC-процессор с симметричной системой команд, позволяющей выполнять операции с любым регистром, используя произвольный метод адресации. Пользователь может сохранять результат операции в самом регистре -аккумуляторе или во втором регистре, используемом для операции.

30. Семейства микроконтроллеров AVR фирмы Atmel. Основные характеристики. Структурная схема. Блок АЛУ, организация памяти программ и данных; карта памяти.

В рамках единой базовой архитектуры микроконтроллеры AVR подразделяются на три семейства:

- Classic AVR;
- Mega AVR;
- Tiny AVR.

Области применения AVR **Для семейства "tiny"** - это интеллектуальные автомобильные датчики различного назначения, игрушки, игровые приставки, материнские платы персональных компьютеров, контроллеры защиты доступа в мобильных телефонах, зарядные устройства, детекторы дыма и пламени, бытовая техника, разнообразные инфракрасные пульты дистанционного управления.

Для семейства "classic" - это модемы различных типов, современные зарядные устройства, изделия класса Smart Cards и устройства чтения для них, спутниковые навигационные системы для определения местоположения автомобилей на трассе, сложная бытовая техника, пульты дистанционного управления, сетевые карты, материнские платы компьютеров, сотовые телефоны нового поколения а также различные и разнообразные промышленные системы контроля и управления.

Для "mega" AVR - это аналоговые (NMT, ETACS, AMPS) и цифровые (GSM, CDMA) мобильные телефоны, принтеры и ключевые контроллеры для них, контроллеры аппаратов факсимильной связи и ксероксов, контроллеры современных дисковых накопителей, CD-ROM и т.д.

Микроконтроллер AVR изготовлен по КМОП технологии, содержит: быстрый RISC-процессор, два типа энергонезависимой памяти (Flash-память программ и память данных EEPROM), оперативную память RAM, порты ввода/вывода и различные

периферийные интерфейсные схемы. Микроконтроллеры имеют Гарвардскую архитектуру - с раздельной памятью программ и данных и раздельными шинами для памяти программ и данных.

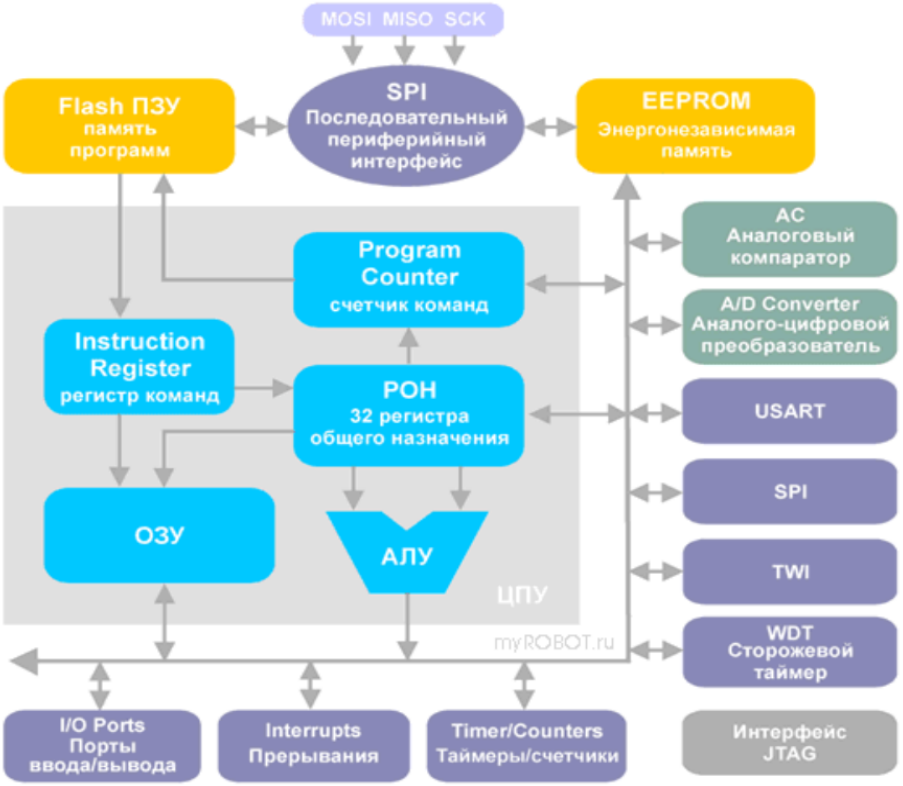
Центральный процессор работает одновременно как с памятью программ, так и с памятью данных; разрядность шины памяти программ расширена до 16 бит. Увеличение быстродействия AVR является также следствием использования технологии конвейеризации, вследствие чего цикл "выборка - исполнение" команды заметно сокращен.

AVR ядро объединяет мощную систему команд с 32 регистрами общего назначения и конвейером (в течение процессорного цикла одна команда выполняется а другая выбирается из памяти программ). Все 32 регистра напрямую связаны с АЛУ, что позволяет выполнять обращение к двум независимым регистрам и возвращать результат одной командой, выполняемой в одном тактовом цикле. Выполняя команды за один тактовый цикл, прибор обеспечивает производительность, приближающуюся к 1 MIPS 11 на МГц, что на порядок больше, чем у CISC микроконтроллеров. Архитектура эффективно поддерживает как языки высокого уровня, так и программы, написанные на языках ассемблера.

Основой ЦПУ является высокопроизводительное **арифметикологическое устройство** (АЛУ). АЛУ подключено к регистрам общего назначения РОН. За один тактовый цикл АЛУ выполняет операцию между регистрами этого регистрового файла. Операции АЛУ подразделяются на три основные категории: арифметические, логические и операции над битами. АЛУ поддерживает арифметические и логические операции между регистрами или между константой и регистром.

Выполняются в АЛУ и операции с отдельными регистрами. Регистров общего назначения всего 32, они имеют байтовый формат, то есть каждый из них состоит из восьми бит. РОН находятся в начале

адресного пространства оперативной памяти, но физически не являются ее частью. Поэтому к ним можно обращаться двумя способами (как к регистрам и как к памяти). Такое решение является особенностью AVR и повышает эффективность работы и производительность микроконтроллера.



Организация памяти

Память программ (Flash ROM или Flash ПЗУ) Память программ предназначена для хранения последовательности команд, управляющих функционированием микроконтроллера, и имеет 16-ти битную организацию. В процессе обработки прерываний и вызовов подпрограмм адрес возврата счетчика команд (PC) сохраняется в стеке. Все пользовательские программы в подпрограммах возврата (прежде, чем подпрограммы или прерывания будут выполняться) должны инициализировать указатель стека (SP). 16-разрядный указатель стека, с возможностью чтения/записи располагается в пространстве I/O. Все AVR имеют Flash-память программ, которая может быть различного размера – от 1 до 256 КБайт. Ее главное достоинство в том, что она построена на принципе электрической перепрограммируемости, т. е. допускает многократное стирание и запись информации.

Все микроконтроллеры семейства Mega имеют возможность самопрограммирования, т. е. самостоятельного изменения содержимого своей памяти программ

Память данных: Память данных разделена на три части: **регистровая память**, **оперативная память** (ОЗУ – оперативное запоминающее устройство или RAM) и **энергонезависимая память** (ЭСППЗУ или EEPROM).

Энергонезависимая память данных (EEPROM) Для долговременного хранения различной информации, которая может изменяться в процессе функционирования микроконтроллерной системы, используется EEPROM-память. Этот тип памяти, доступный программе микроконтроллера непосредственно в ходе ее выполнения, удобен для хранения промежуточных данных, различных констант, коэффициентов, серийных номеров, ключей и т.п. EEPROM может быть загружена извне как через SPI интерфейс, так и с помощью обычного программатора. Число циклов стирание/запись – не менее 100 тыс.

Оперативная память (ОЗУ или RAM) Внутренняя оперативная статическая память Static RAM (SRAM) имеет байтовый формат и используется для оперативного хранения данных. Размер оперативной памяти может варьироваться у различных чипов от 64 Байт до 4 КБайт. Число циклов чтения и записи в RAM не ограничено, но при отключении питающего напряжения вся информация теряется. Для некоторых микроконтроллеров возможна организация подключения внешнего статического ОЗУ объемом до 64К

31. Периферия микроконтроллеров AVR фирмы Atmel.

Периферия микроконтроллеров AVR включает:

- **порты (от 3 до 48 линий ввода и вывода)**
  Архитектурная особенность построения портов ввода-вывода у AVR заключается в том, что для каждого физического вывода (пина) существует 3 бита контроля/управления, а не 2, как у распространенных 8-разрядных микроконтроллеров (Intel, Microchip, Motorola и т.д.). Это позволяет избежать необходимости иметь копию содержимого порта в памяти для безопасности и повышает скорость работы микроконтроллера при работе с внешними устройствами, особенно в условиях внешних электрических помех,
- **поддержку внешних прерываний**
  Система прерываний – одна из важнейших частей микроконтроллера. Все микроконтроллеры AVR имеют многоуровневую систему прерываний. Прерывание прекращает нормальный ход программы для выполнения приоритетной задачи, определяемой внутренним или внешним событием. Для каждого такого события разрабатывается отдельная программа, которую называют подпрограммой обработки запроса на прерывание (для краткости – подпрограммой прерывания,
- **таймеры-счетчики**
  Микроконтроллеры AVR имеют в своем составе от 1

до 4 таймеров/счетчиков с разрядностью 8 или 16 бит, которые могут работать и как таймеры от внутреннего источника тактовой частоты, и как счетчики внешних событий. Их можно использовать для точного формирования временных интервалов, подсчета импульсов на выводах микроконтроллера, формирования последовательности импульсов, тактирования приемопередатчика последовательного канала связи. В режиме ШИМ (PWM) таймер/счетчик может представлять собой широтно-импульсный модулятор и используется для генерирования сигнала с программируемыми частотой и скважностью,

- **сторожевой таймер**
  Идея использования сторожевого таймера предельно проста и состоит в регулярном его сбрасывании под управлением программы или внешнего воздействия до того, как закончится его выдержка времени и не произойдет сброс процессора. Если программа работает нормально, то команда сброса сторожевого таймера должна регулярно выполняться, предохраняя процессор от сброса,
- **аналоговые компараторы**
  сравнивает напряжения на двух выводах (пинах) микроконтроллера. Результатом сравнения будет логическое значение, которое может быть прочитано из программы,
- **10-разрядный 8-канальный АЦП**
  Аналого-цифровой преобразователь (АЦП) служит для получения числового значения напряжения, поданного на его вход. Этот результат сохраняется в регистре данных АЦП,
- **интерфейсы UART**
  удобный и простой последовательный интерфейс для организации информационного канала обмена микроконтроллера с внешним миром. Способен

работать в дуплексном режиме (одновременная передача и прием данных).,

- JTAG
  Четырехпроводной интерфейс JTAG используется для тестирования печатных плат, внутрисхемной отладки, программирования микроконтроллеров. Многие микроконтроллеры семейства Mega имеют совместимый с IEEE Std 1149.1 интерфейс JTAG или debugWIRE для встроенной отладки,
- TWI
  проводной последовательный интерфейс TWI (Two-wire Serial Interface) является полным аналогом базовой версии интерфейса I2C фирмы Philips. Этот интерфейс позволяет объединить вместе до 128 различных устройств с помощью двунаправленной шины, состоящей из линии тактового сигнала и линии данных,
- SPI
  С его помощью может осуществляться обмен данными между микроконтроллером и различными устройствами, такими, как цифровые потенциометры, ЦАП/АЦП, FLASH ПЗУ и др. С помощью этого интерфейса удобно производить обмен данными между несколькими микроконтроллерами AVR. Кроме того, 25 через интерфейс SPI **может осуществляться программирование микроконтроллера,**
- **устройство сброса по понижению питания**
  отслеживает напряжение источника питания. Если схема включена, то при снижении питания ниже некоторого значения она переводит микроконтроллер в состояние сброса. Когда напряжение питания вновь увеличится до порогового значения, запускается таймер задержки сброса. После формирования задержки внутренний сигнал сброса снимается и происходит запуск микроконтроллера,

Короткие вопросы

17. Какие основные области памяти данных имеют место у микроконтроллеров семейства PIC?

Память данных МК разбита на две области. Первые 12 адресов – это область **регистров специальных функций** (SFR), а вторая – область **регистров общего назначения** (GPR).

Область SFR предназначенна для управления периферийными устройствами и конфигурации микроконтроллера.

18. За сколько циклов выполняются команды в микроконтроллерах семейства PIC?

Цикл выполнения команды состоит из четырех тактов: Q1...Q4 . Выборка команды и ее выполнение совмещены по времени таким образом, что выборка команды занимает один цикл, а выполнение – следующий цикл.

Эффективное время выполнения команды составляет один цикл. Если команда изменяет счетчик команд (например, команда GOTO), то для ее выполнения потребуется два цикла.

рисунки Схема тактирования и выполнения команды, Выборка команд.

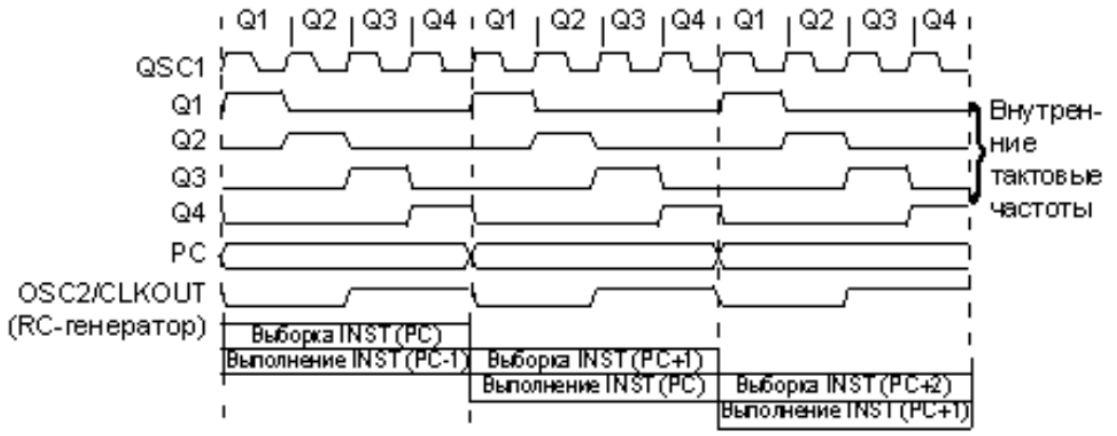


Рис. 5.2. Схема тактирования и выполнения команды.

Рис. 5.3. Выборка команд.

19. Семейства микроконтроллеров AVR фирмы Atmel

1. **ATmega**: Например, ATmega328P - это одна из самых распространенных моделей, используемых в Arduino-платформе. это аналоговые (NMT, ETACS, AMPS) и цифровые (GSM, CDMA) мобильные телефоны, принтеры и ключевые контроллеры для них, контроллеры аппаратов факсимильной связи и ксероксов, контроллеры современных дисковых накопителей, CD-ROM и т.д.
2. **ATtiny**: Семейство ATtiny представляет собой более компактные и экономичные микроконтроллеры AVR, которые обычно имеют меньший объем памяти и меньшее количество периферийных модулей. интеллектуальные автомобильные датчики различного назначения, игрушки, игровые приставки, материнские платы персональных компьютеров, контроллеры защиты доступа в мобильных телефонах, зарядные устройства,

детекторы дыма и пламени, бытовая техника, разнообразные инфракрасные пульты дистанционного управления.

3. **Classic: AT90S** это модемы различных типов, современные зарядные устройства, изделия класса Smart Cards и устройства чтения для них, спутниковые навигационные системы для определения местоположения автомобилей на трассе, сложная бытовая техника, пульты дистанционного управления, сетевые карты, материнские платы компьютеров, сотовые телефоны нового поколения а также различные и разнообразные промышленные системы контроля и управления.