Наименование дисциплины
Микропроцессорные системы

Для курса 3 _____ групп ИВБО-1, 2_____

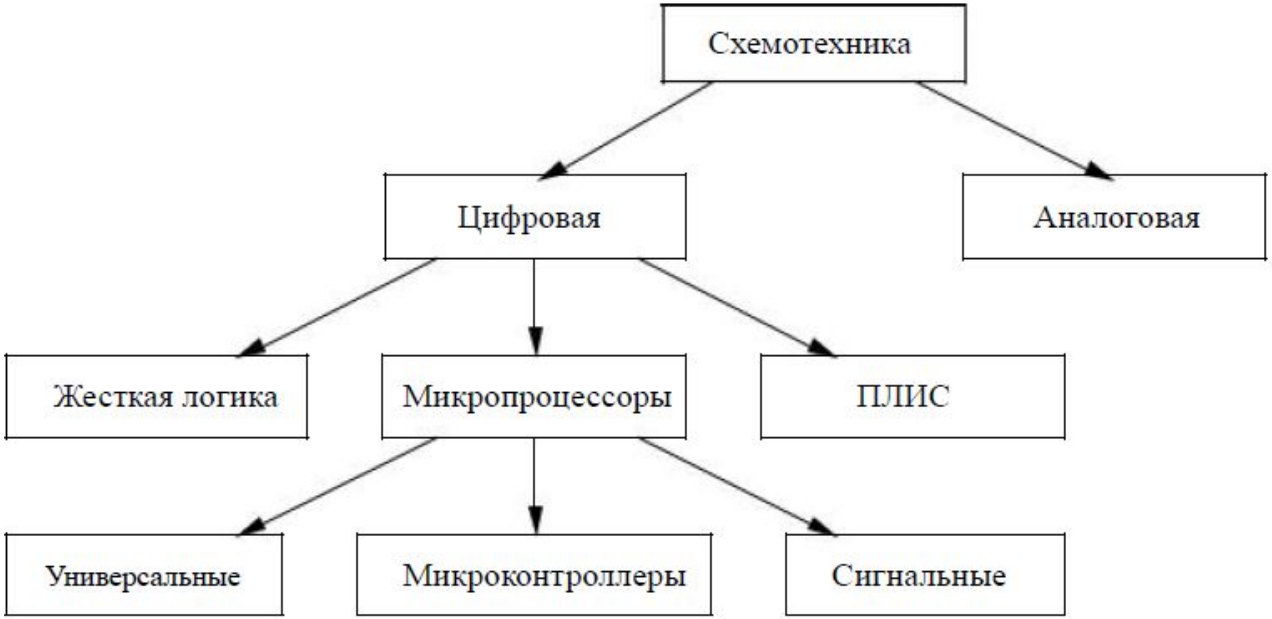
По направлению подготовки
бакалавры

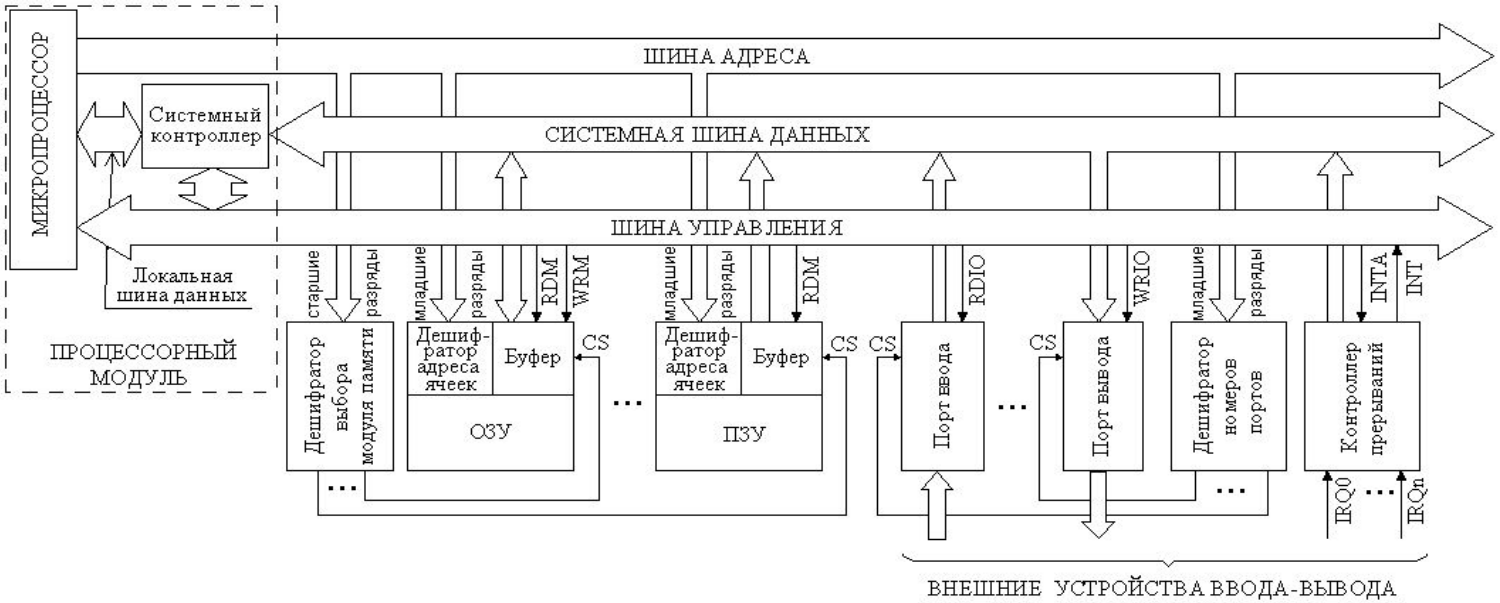
Москва, 2020

Содержание:

Лекция 9. Интерфейс SPI
Лекция 10. Интерфейс I2C
Лекция 11. CAN-шина и протокол (Control Area Network)
Лекция 12. Микроконтроллеры семейств PIC (Peripheral Interface Controller) компании Microchip
Лекция 13. Микроконтроллеры семейств PIC (продолжение)
Лекция 14. Микроконтроллеры семейств PIC (продолжение)
Лекция 15. 8-разрядные микроконтроллеры AVR фирмы Atmel.
Лекция 16. 8-разрядные микроконтроллеры AVR (продолжение)
Лекция 17. Основные этапы разработки микропроцессорной системы на основе микроконтроллеров

МП-системы, основные понятия. Процессор: характеристики, области применения.


Схемотехника
Цифровая
Аналоговая
Жесткая логика
Микропроцессоры
ПЛИС
Универсальные
Микроконтроллеры
Сигнальные



Типичная структура МП-системы с общей шиной

Основные типы МП-систем следующие (от более простых к более сложным):

микроконтроллеры — наиболее простой тип микропроцессорных систем, в которых все или большинство узлов системы выполнены в виде одной микросхемы;

контроллеры — управляющие микропроцессорные системы, выполненные в виде отдельных модулей;

микрокомпьютеры — более мощные микропроцессорные системы с развитыми средствами сопряжения с внешними устройствами.

компьютеры (в том числе персональные) — самые мощные и наиболее универсальные микропроцессорные системы.

Процессор – это цифровое устройство, осуществляющее обработку информации и программное управление этим процессом.

Процессор является основным узлом системы. Остальные узлы выполняют вспомогательные функции: хранение информации, связь с внешними устройствами, связь с пользователем и т.д.

Алгоритм работы системы задается программой, которая хранится в памяти. Программа представляет собой последовательность команд (инструкций), каждая из которых является требованием на выполнение некоторого действия.

Процессор выполняет команды последовательно одну за другой. Для выполнения каждой команды процессор осуществляет следующие действия:

1. Выборка команды
2. Дешифрация кода операции
3. Выборка операндов
4. Выполнение операции
5. Запись результатов

Основные характеристики процессора

1. Тактовая частота

Все процессоры являются синхронными устройствами, т.е. любые операции, выполняемые ими, синхронизируются по времени тактовым сигналом. В зависимости от сложности операции, выполняемой процессором, она может длиться несколько тактов.

Быстродействие процессора может быть намного выше быстродействия остальных компонентов системы. Поэтому используются внешняя и внутренняя тактовые частоты.

Внешняя тактовая частота процессора, равная частоте системной шины, синхронизирует операции обмена данными по системной шине между процессором и другими устройствами.

Внутренняя тактовая частота процессора используется для синхронизации работы самого процессора.

2. Разрядность данных

Различают внутреннюю и внешнюю разрядность данных.

Внутренняя – это число бит данных, которое процессор может обрабатывать одновременно.

Внешняя – это число бит данных, которое можно за один раз передать между процессором и другими устройствами системы.

В зависимости от типа процессора разрядность данных может составлять 8, 16, 32, 64, 128 бит и т.д..

3. Адресное пространство

Это совокупность ячеек оперативной и постоянной памяти и адресов устройств ввода-вывода, к которым может обращаться процессор.

Максимальный объем адресного пространства составляет 2^n слов, где n – разрядность шины адреса.

4. Система команд и способы адресации (см. ЭВМ и ПУ)

5. Система прерываний

Обеспечивает возможность процессора реагировать на возникающие внешние и внутренние ситуации.

Виды прерываний:

а).Внешние (аппаратные)

Источник прерывания – сигнал на входе процессора.

Для внешних прерываний выделяется как минимум одна управляющая линия шины, называемая линией запроса прерывания.

б) Внутренние (исключения)

Источник прерывания – исключительные ситуации, возникающие в процессоре.

в) Программные

Источник прерывания – вызов команды прерывания из исполняемой программы.

Микропроцессор – это процессор, реализованный в виде одной микросхемы или комплекта из нескольких специализированных микросхем

Классификация МП

1. *По архитектуре*

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 обрабатываются 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 процессора для выполнения операций распределяются динамически.

г) VLIW (Very Long Instruction Word, очень длинное командное слово)

2. *По областям применения*:

а) Универсальные процессоры

Предназначены для решения широкого круга задач. Применяются в ПК, рабочих станциях, серверах и других ВС.

б) Микроконтроллеры

Предназначены для построения устройств управления различной аппаратурой.

в) Цифровые процессоры обработки сигналов

Предназначены для цифровой обработки аналоговых сигналов.

г) Медийные процессоры

Предназначены для преобразования графической и звуковой информации.

3. *По типу управления*

а) Асинхронные

б) Синхронные.

4. *По производительности*:

а) Низкая (для встроенных систем);

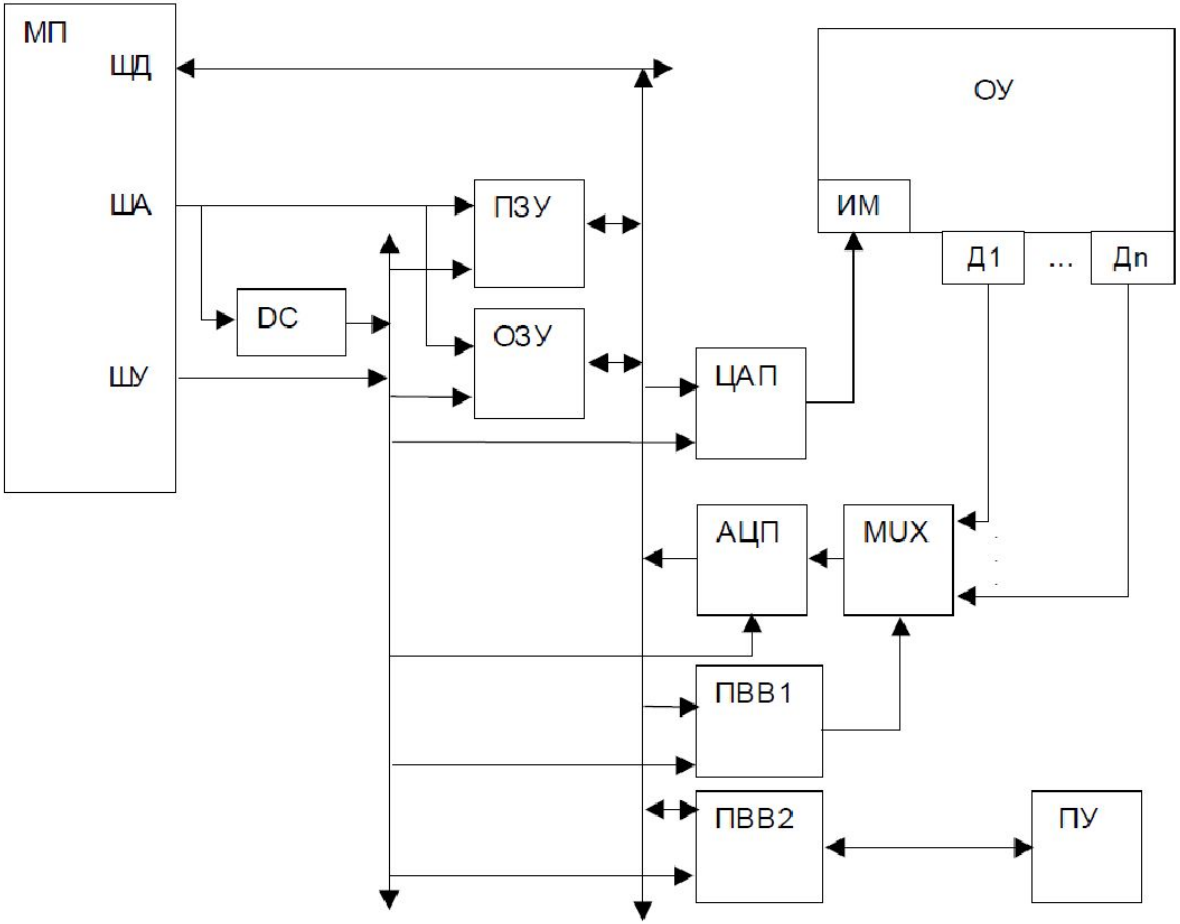
б) Средняя (для ПК);

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

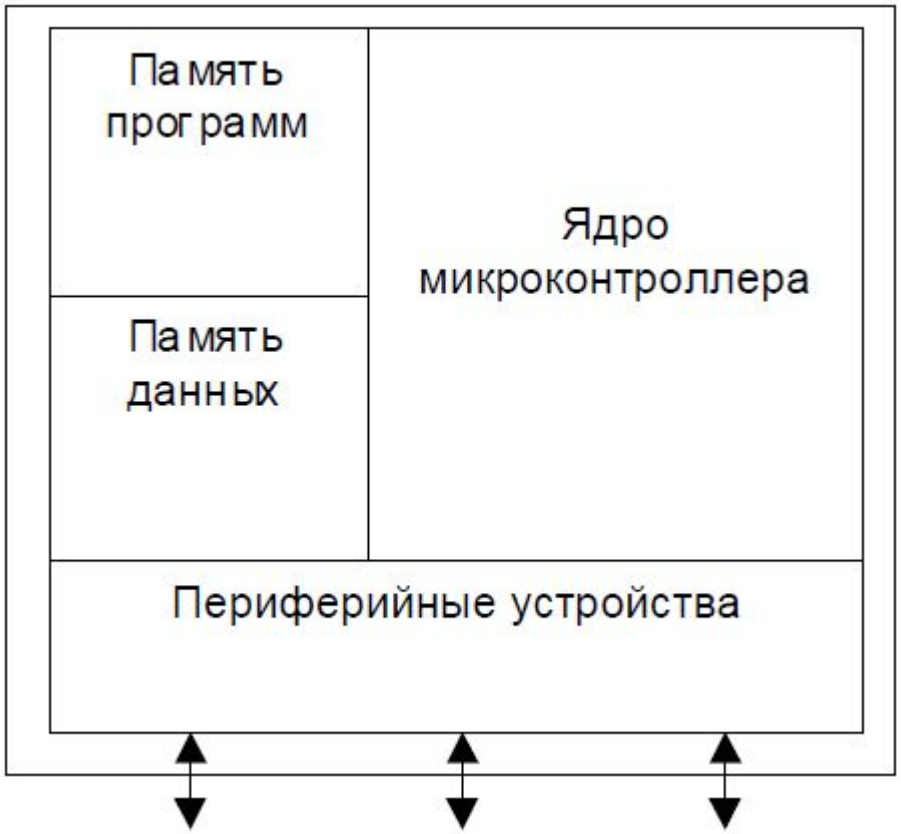
5. *По стоимости*

Аналогично классификации по производительности

Микроконтроллеры, назначение, структура

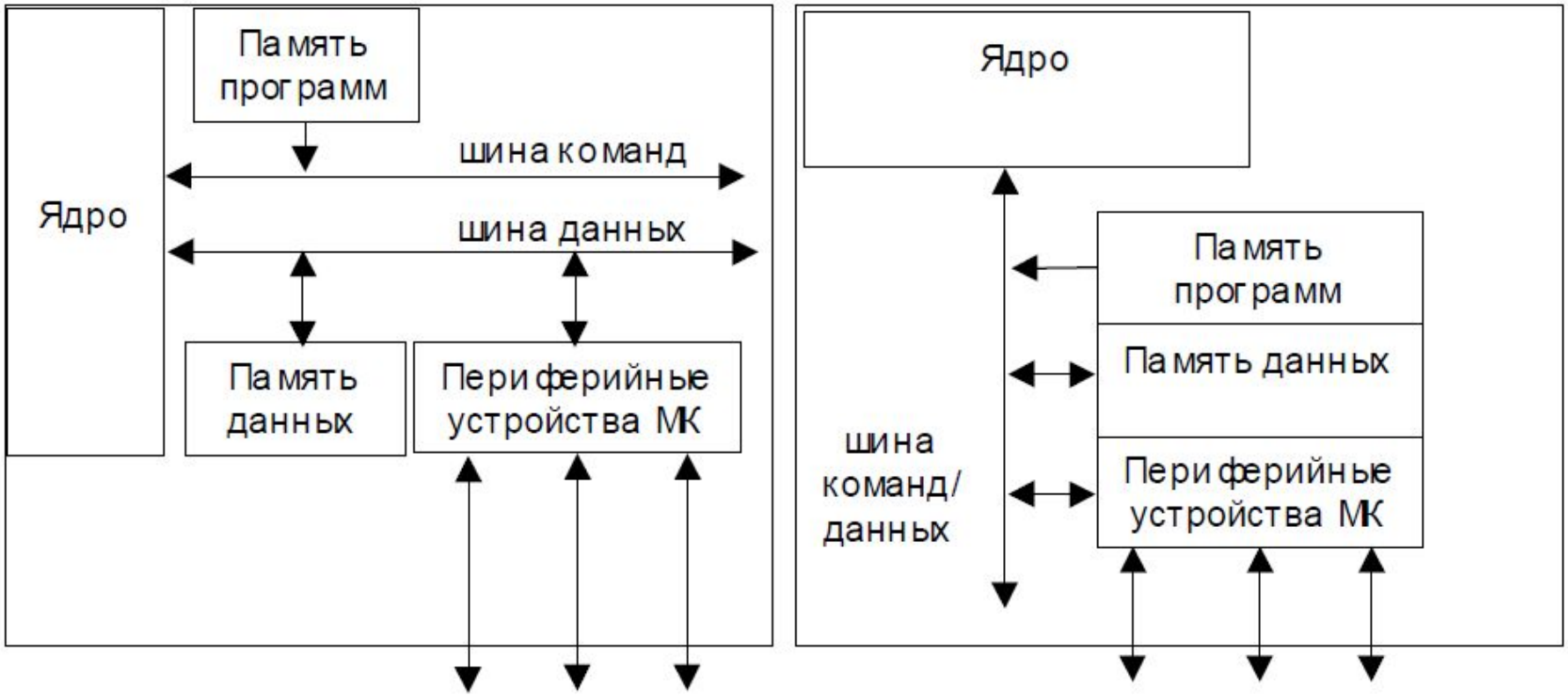


Система управления на базе микропроцессора



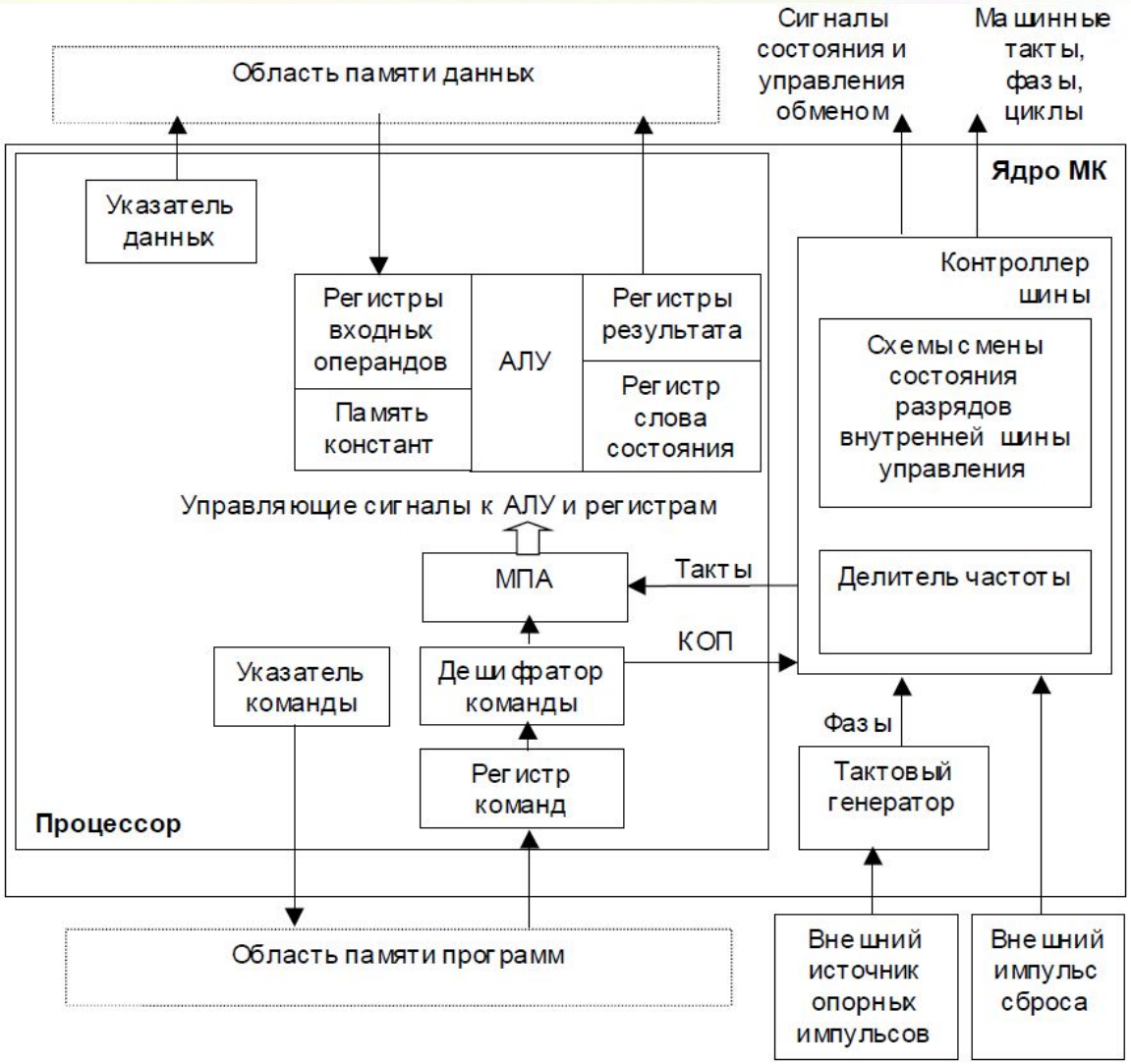
Обобщенная структурная схема микроконтроллера

Гарвардская (а) и принстонская (б) архитектуры микроконтроллеров



а) б)

Структурная схема микроконтроллера



Микроконтроллеры

Микроконтроллер (МК) –это разновидность микропроцессора, ориентированная на реализацию алгоритмов управления техническими устройствами и технологическими процессами.

МК намного более распространены, чем МП. Области применения МК:

промышленная автоматика;

автомобильная электроника;

измерительная техника;

аппаратура связи;

теле- , видео- , аудиоаппаратура;

бытовая техника и др.

К МК предъявляются повышенные требования по стоимости, габаритам и устойчивости к внешним воздействиям (вибрации, температура).

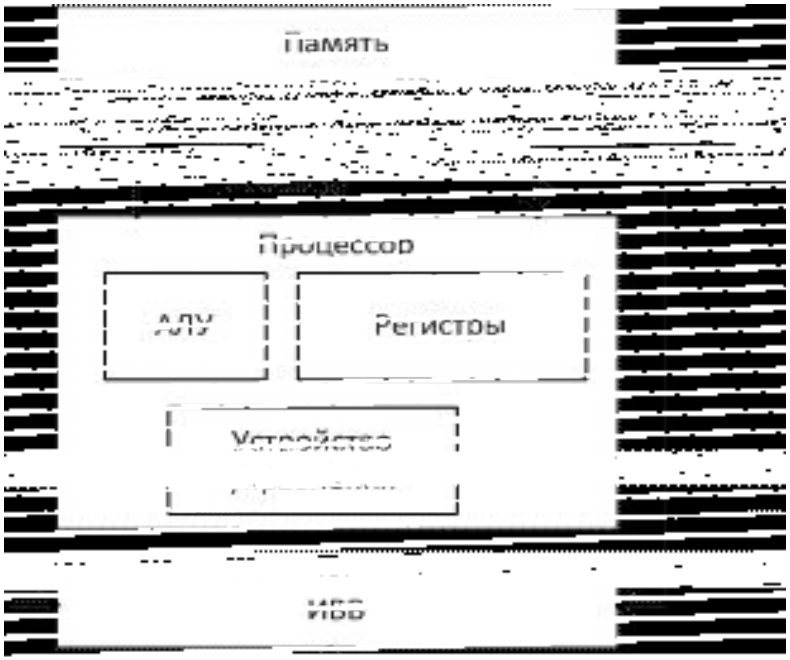
МК объединяет в одном кристалле: центральный процессор (ЦП), внутреннюю память, периферийные устройства для ввода и вывода информации (УВВ).

Типы МК:

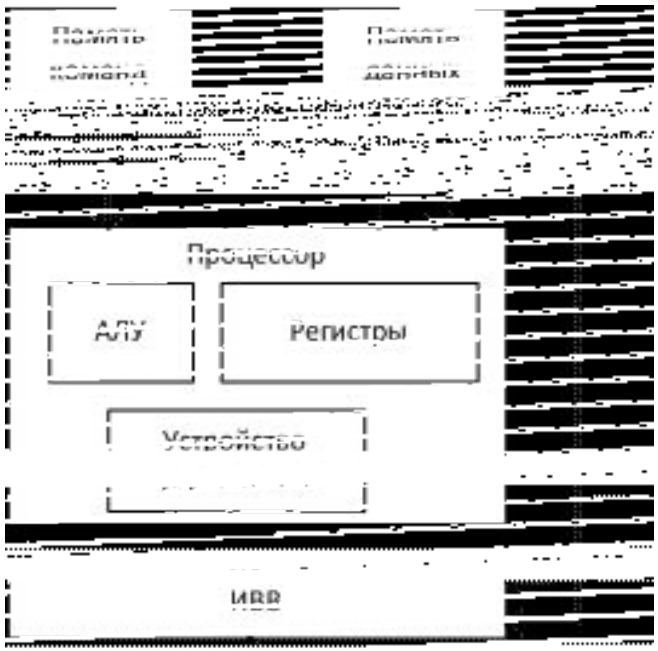
-8-ми разрядные МК. Наиболее многочисленная группа. Это простые и дешевые МК с относительно низкой производительностью.

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

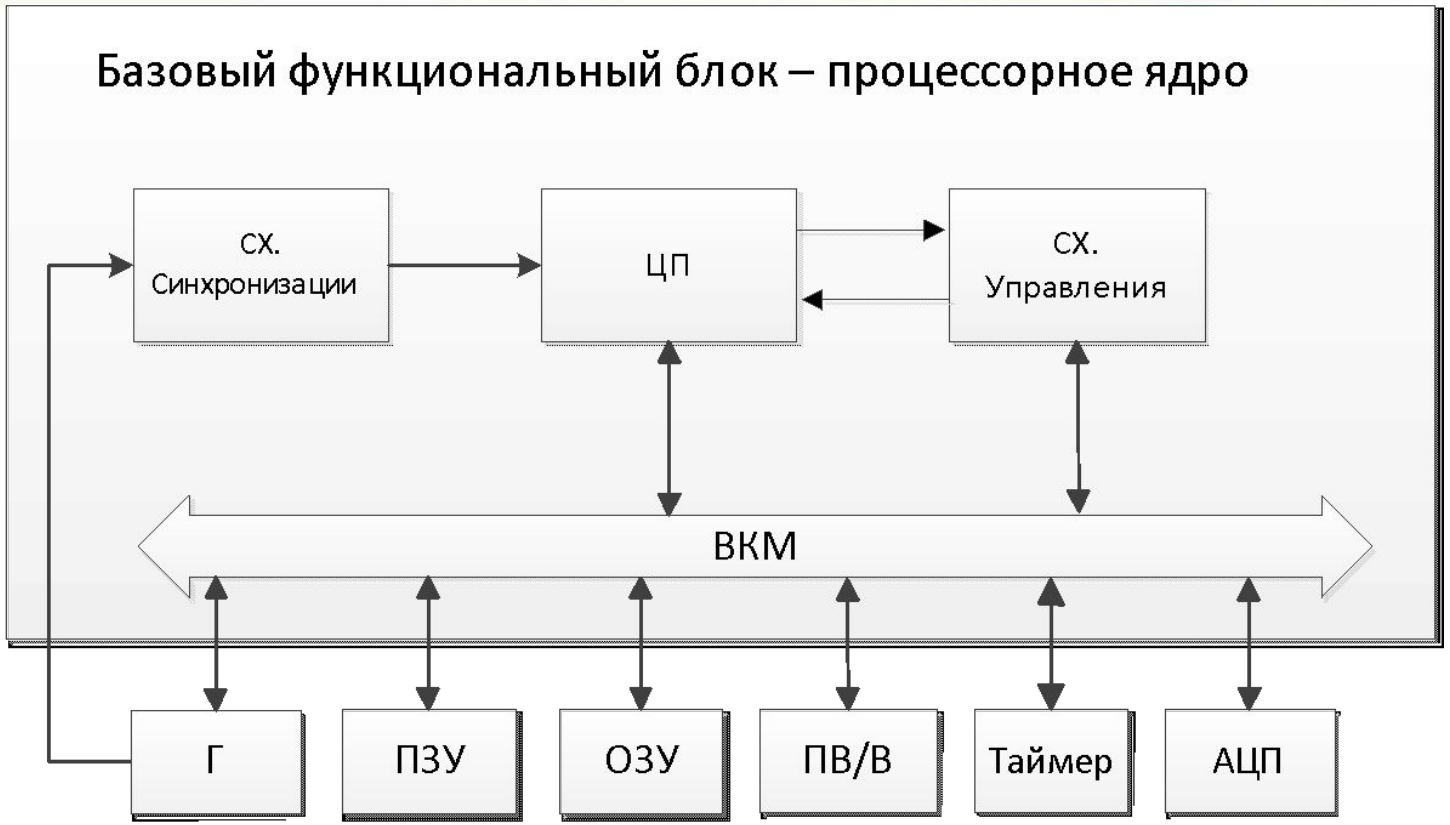
- 32-х разрядные МК. Содержат высокопроизводительный процессор, соответствующий младшим моделям процессоров общего назначений. Применение этих процессоров позволяет использовать разработанное ранее ПО для ПК на базе этих процессоров.



Фон Неймановская (Принстонская) архитектура



Гарвардская архитектура



Обобщенная структурная схема микроконтроллера

Отличительные признаки микроконтроллеров:

•*Модульный способ организации.* На основе одного ядра проектируется целое семейство микроконтроллеров и отличается семейство другими блоками: объемом и видом памяти, частотой синхронизации, набором периферийных устройств.

•*Закрытая архитектура.* Отсутствуют внешние шины данных и адреса на выводах корпуса микроконтроллера. Законченное устройство не предполагает расширений.

•*Наличие типовых периферийных модулей* (таймер, процессор событий, ЦАП, АЦП и др.)

Процессорное ядро микроконтроллера 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.

Каждый модуль имеет выводы для подключения к магистралям процессорного ядра. Это позволяет создавать разнообразные по структуре МК в пределах одного семейства.

Память МК

В МК используется три основных вида памяти.

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

2.память данных – ОЗУ, предназначена для хранения переменных в процессе выполнения программы.

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

Классификация ПЗУ

Память данных

Как правило, выполняется на основе статического ОЗУ.

Содержимое ячеек ОЗУ сохраняется при снижении тактовой частоты МК до сколь угодно малых значений. Это 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. Это позволяет в случае необходимости перевести МК на питание от автономного источника и сохранить данные ОЗУ.

Внешняя память

В некоторых случаях возникает необходимость подключения дополнительной внешней памяти (как памяти программ, так и данных).

Если МК содержит необходимые для этого аппаратные средства (выводы ША, ШД, ШУ), то подключение производится аналогично подключению ОЗУ к МП.

Второй способ – использовать порты ввода/вывода и реализовать обращение к памяти программными средствами. Недостаток: снижение быстродействия системы при обращении к внешней памяти.

Таймеры-счетчики микроконтроллеров

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 формирование 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.
- контроль за изменением состояния линий ввода микроконтроллера

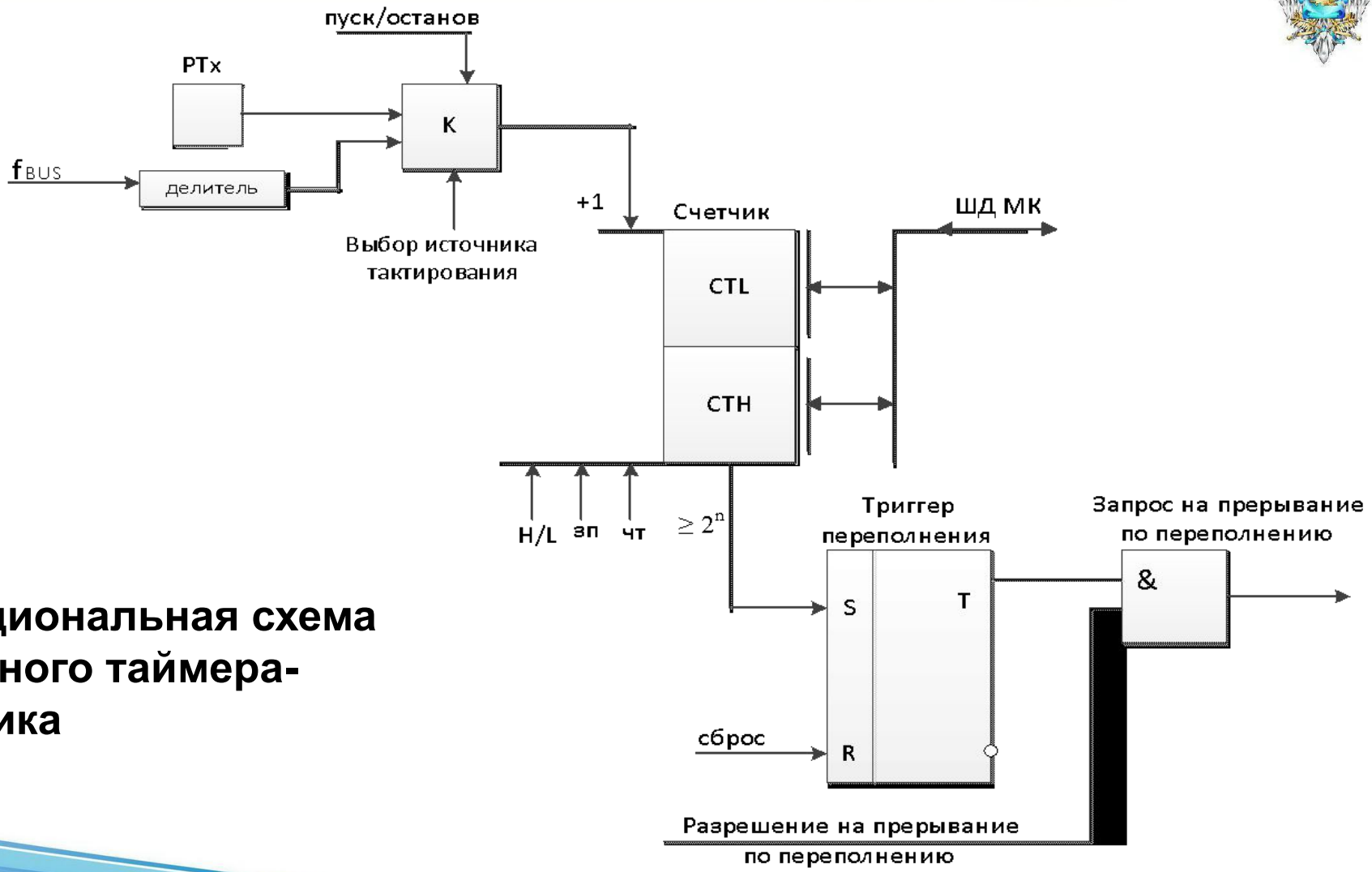
Решение перечисленных задач программным путем без использования аппаратных средств является неэффективным, так как занимает ресурсы, необходимые для вычислений. Поэтому для решения задач управления в реальном времени используют специальные аппаратные средства, которые называют таймерами.

Режимы работы счетчика

1.Режим таймера – отсчет времени через подсчет внутренних импульсов синхронизации с выхода управляемого делителя частоты.

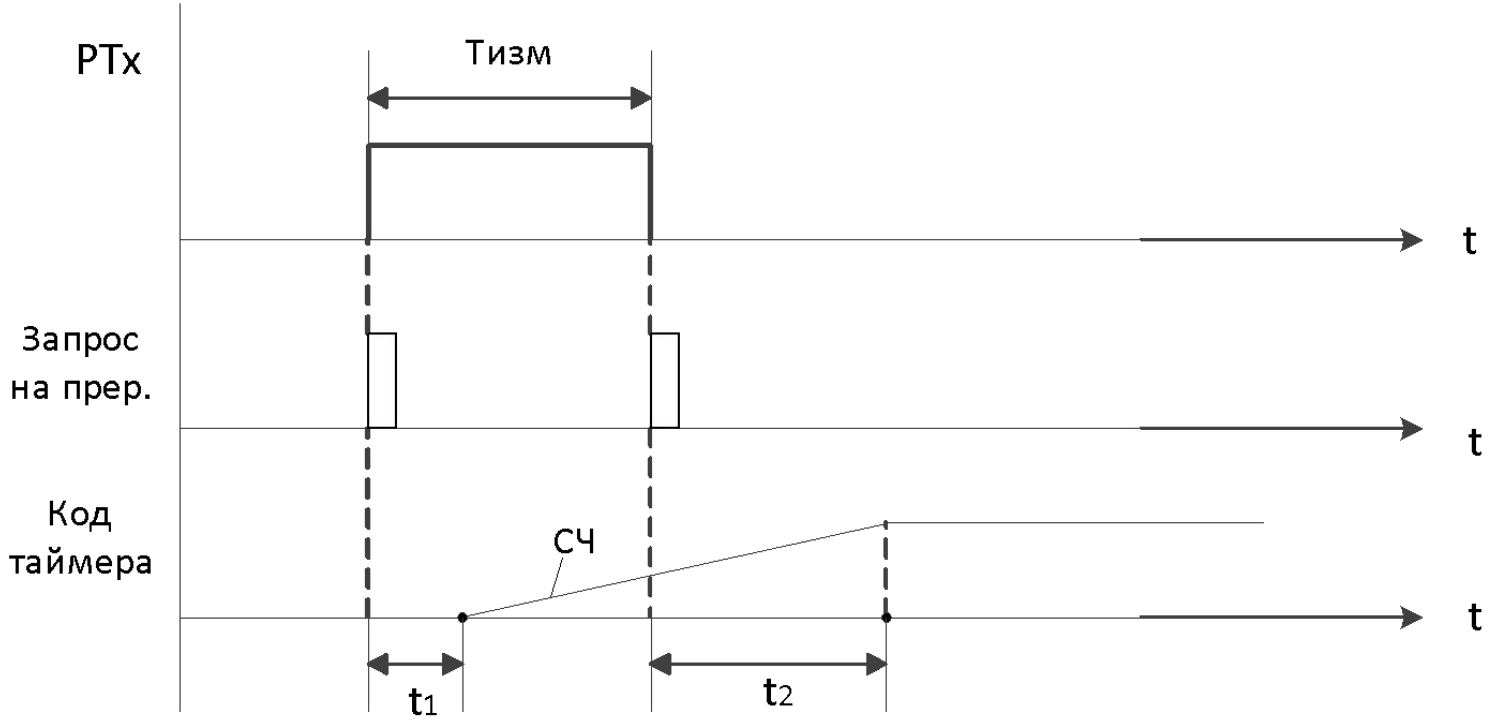
2. Режим счетчика – подсчет событий на внешнем входе МК.

И в 1-м, и во 2-м случае переход через заранее установленное значение приводит к формированию запроса на прерывание.



Функциональная схема типичного таймера-счетчика

Измерение временного интервала с помощью обычного таймера



t1 – разрешение счета; t2 – остановка счета

Для измерения временного интервала выполняется следующая последовательность действий:

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}

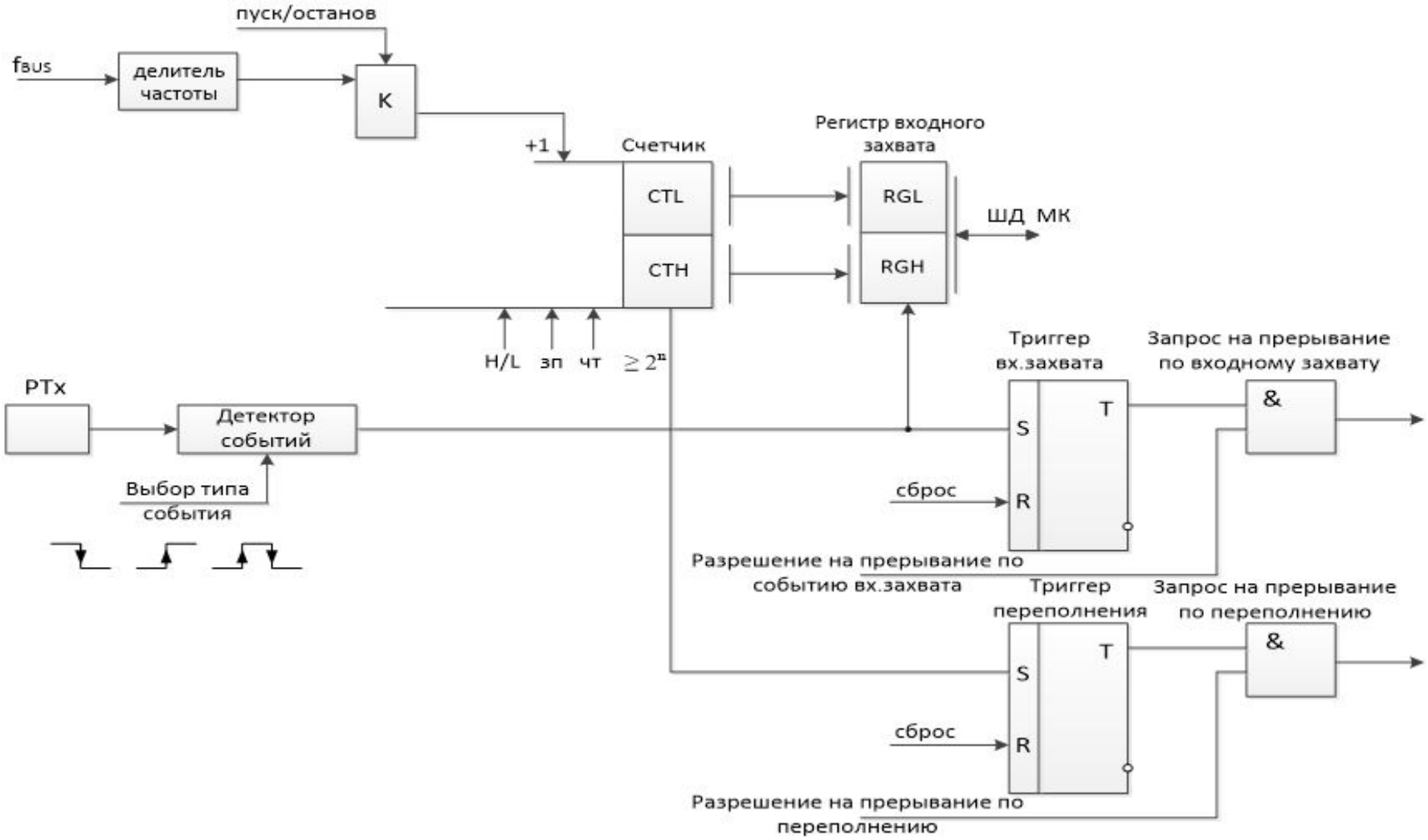
Усовершенствование таймеров-счетчиков

Усовершенствование модуля таймера-счетчика

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

Одним из недостатков модуля типичного таймера-счетчика является невозможность одновременного обслуживания нескольких каналов.

Увеличение числа каналов возможно увеличением числа модулей таймеров-счетчиков в составе микроконтроллера



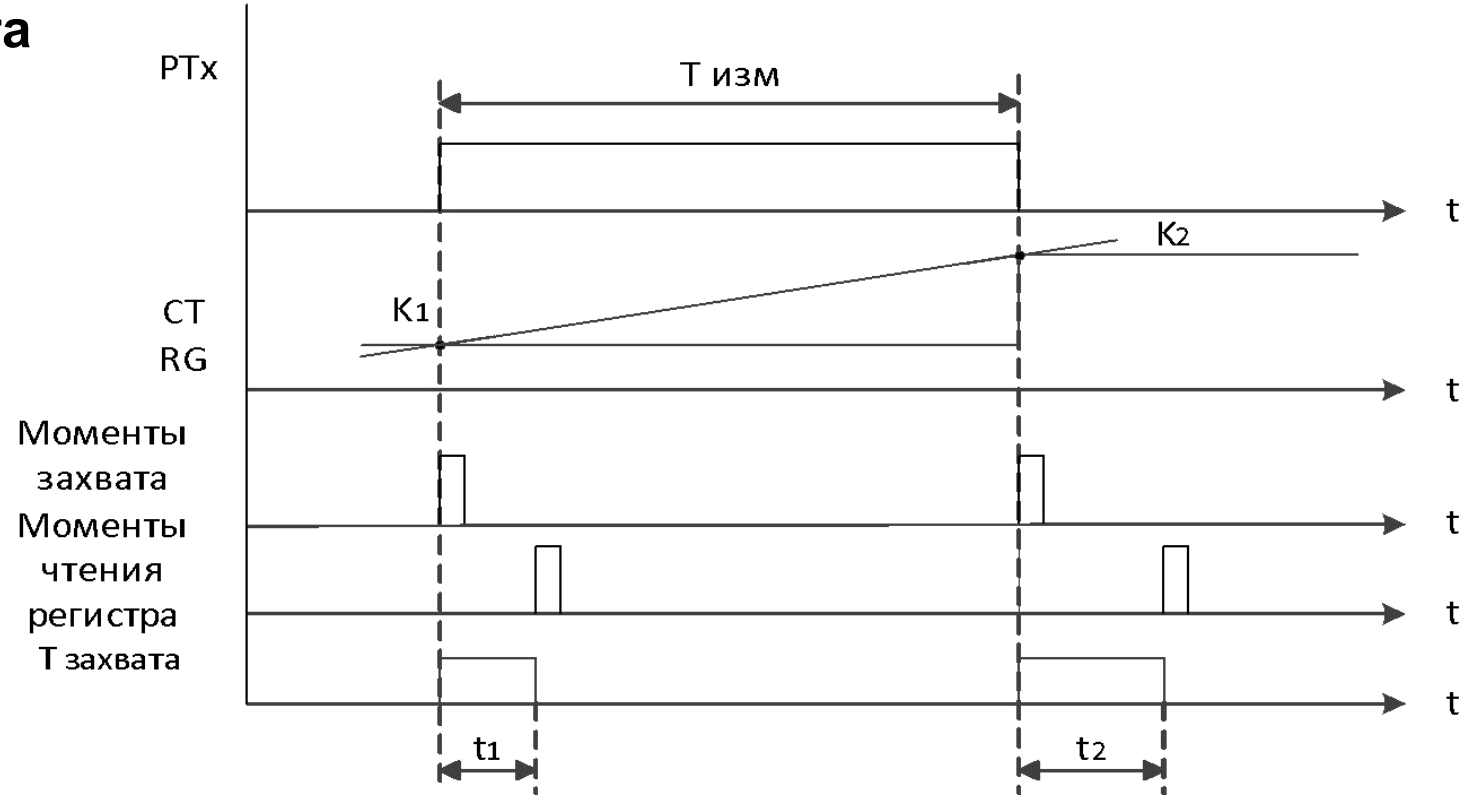
Канал входного захвата

Детектор событий следит за уровнем напряжения на одном из входов МК. 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. 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.

Выбор типа события захвата устанавливается в процессе инициализации таймера и может изменяться в ходе выполнения программы.
При возникновении события захвата устанавливается в 1 триггер входного захвата. Состояние триггера входного захвата может быть считано программно. Если разрешены прерывания по событию захвата, то формируется запрос на прерывание.

Измерение временного интервала с помощью канала входного захвата



t_1, t_2 – время перехода на подпрограмму прерывания

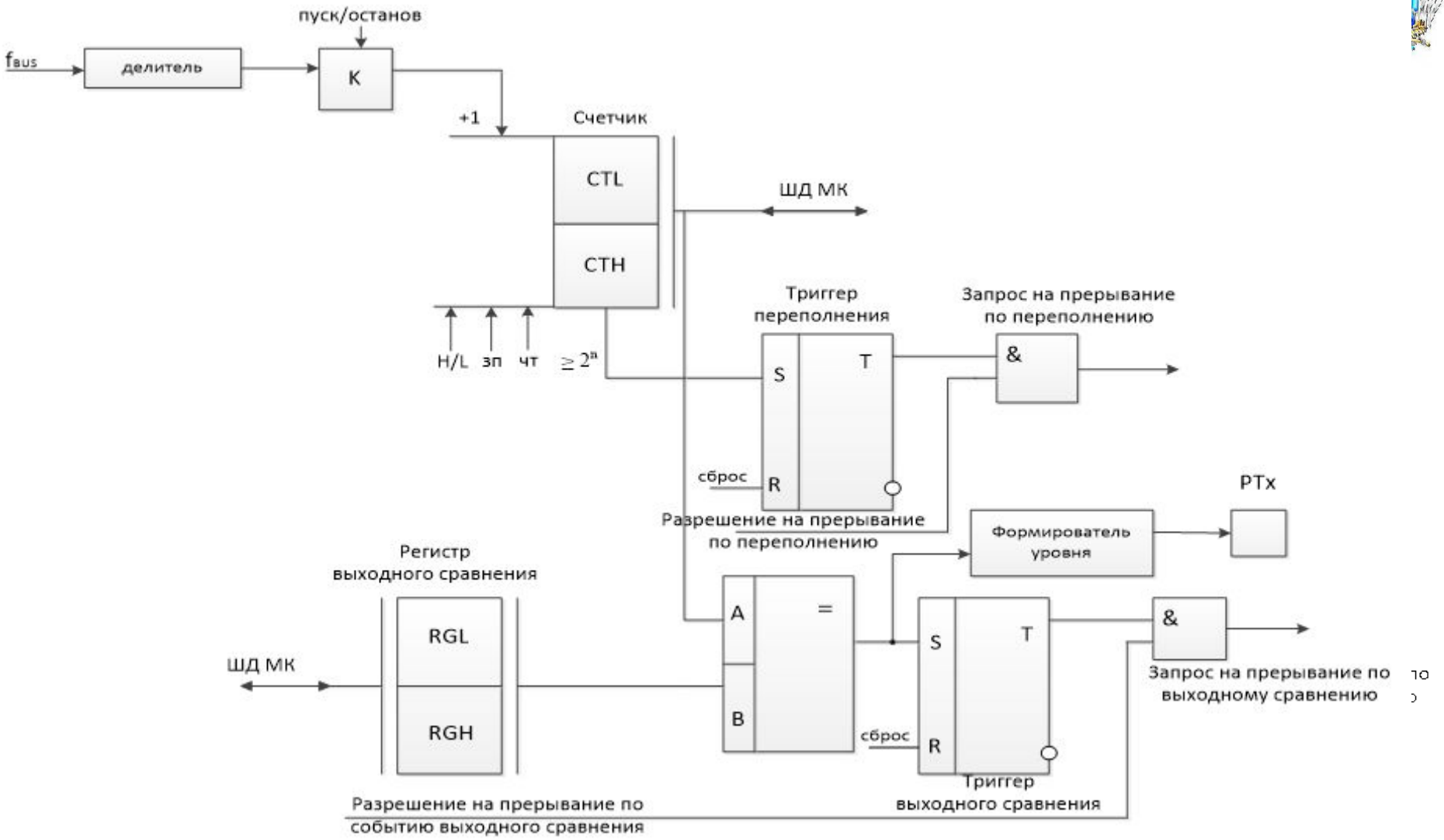
$\Delta t = K_2 - K_1$ - длительность измерительного интервала

$T_{изм} > t_1$ – ограничение на длительность измеряемого интервала

При изменении уровня входного сигнала с 0 на 1 код счетчика К1 копируется в регистр захвата. Триггер захвата устанавливается в 1. Формируется запрос на прерывание (таймер сообщает МК, что интервал начался). С задержкой t1 МК считывает код К1 из регистра захвата, сбрасывает триггер захвата 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 захвата. Снова выставляется запрос на прерывание. С задержкой t2 код считывается в память МК. Разность кодов К2-К1 и есть длительность измеряемого интервала, выраженная числом периодов тактовой частоты.

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



Канал выходного сравнения

Канал выходного сравнения

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

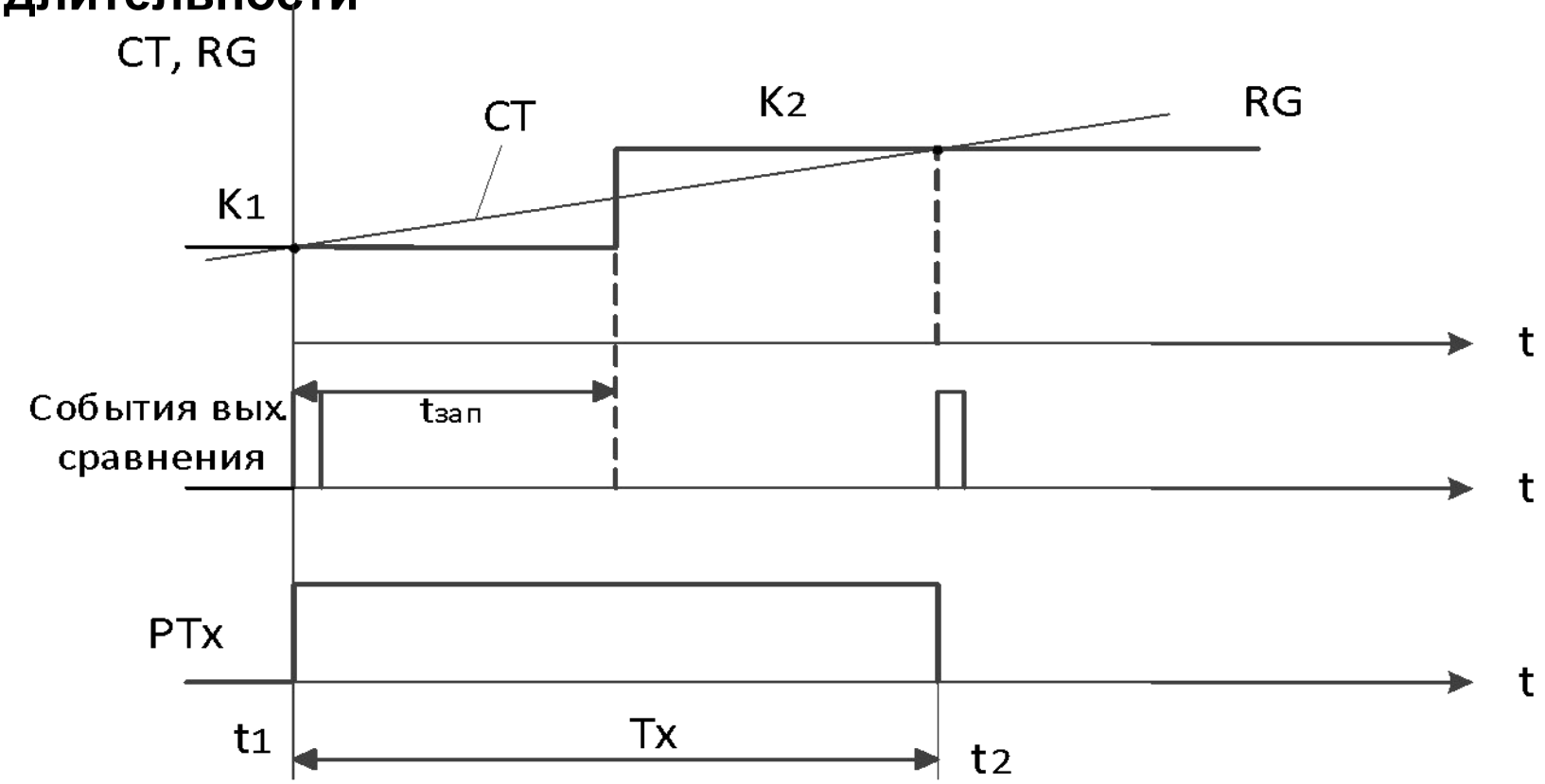
Возможны три типа изменения сигнала на выходе PTx2 в момент события выходного сравнения:

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

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

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

Формирование временного интервала заданной длительности



$T_x = K_2 - K_1$

$T_x > t_{зап}$ - ограничение на длительность формируемого интервала

Первое событие сравнения в момент t1 формирует нарастающий фронт сигнала РТх2. Одновременно генерируется запрос на прерывание. В п/п прерывания происходит загрузка нового кода сравнения К2. Время, необходимое для записи нового значения в регистр сравнения , ограничивает минимальную длительность формируемого интервала.

В момент t2 наступает второе событие, и выход РТх2 устанавливается в 0. Длительность сформированного интервала tx определяется только разностью кодов К2 и К1.

Процессор событий, WDT, АЦП

Процессоры событий -программируемый счетный массив (Programmable Counter Array, PCA).

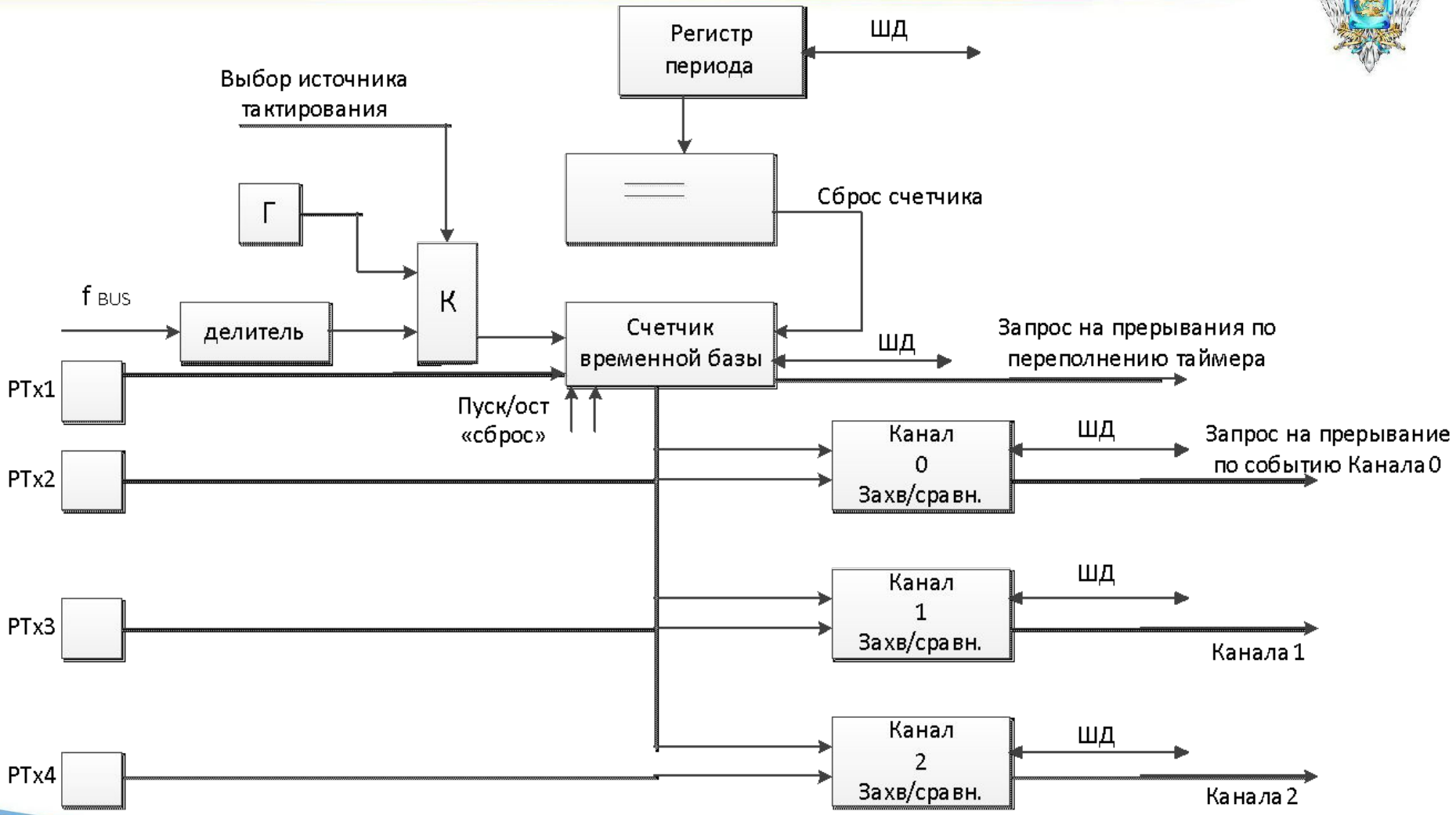
Усовершенствованный таймер позволяет решить многие задачи управления в реальном времени, но имеет следующие ограничения:

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

В процессорах событий устранены ограничения усовершенствованных таймеров. Процессор событий состоит из 16-разрядного счетчика временной базы и нескольких универсальных каналов захвата/сравнения.

Режимы работы универсальных каналов:

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

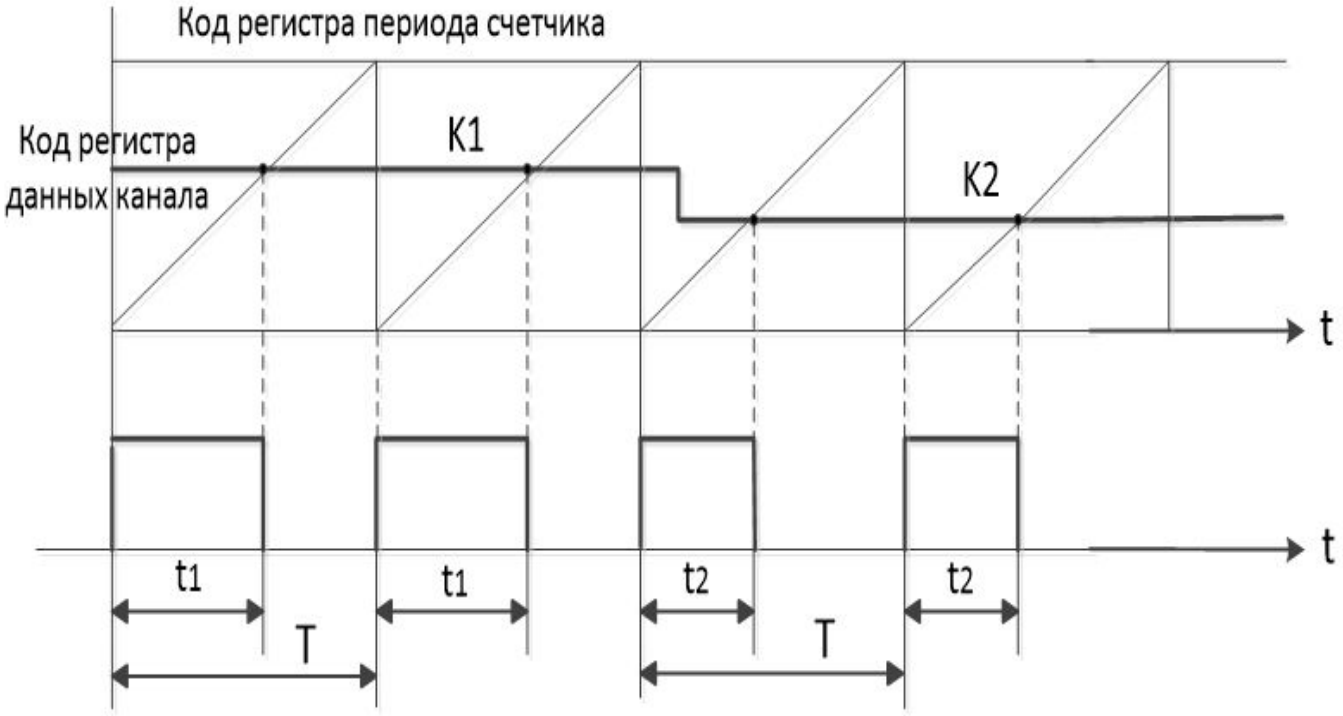


Процессор событий

Режим ШИМ

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.

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



r1= t1/T r2= t2/T

Временная диаграмма функционирования канала в режиме ШИМ

Сторожевой таймер (WDT – Watchdog Timer)

Обеспечивает перезагрузку процессора при зависании

Основа таймера – счетчик, который обнуляется при сбросе микроконтроллера. При переходе МК в активный режим значение счетчика начинает увеличиваться, независимо от выполняемой программы. При достижении счетчиком максимального кода генерируется сигнал внутреннего сброса, и МК начинает выполнять рабочую программу сначала.

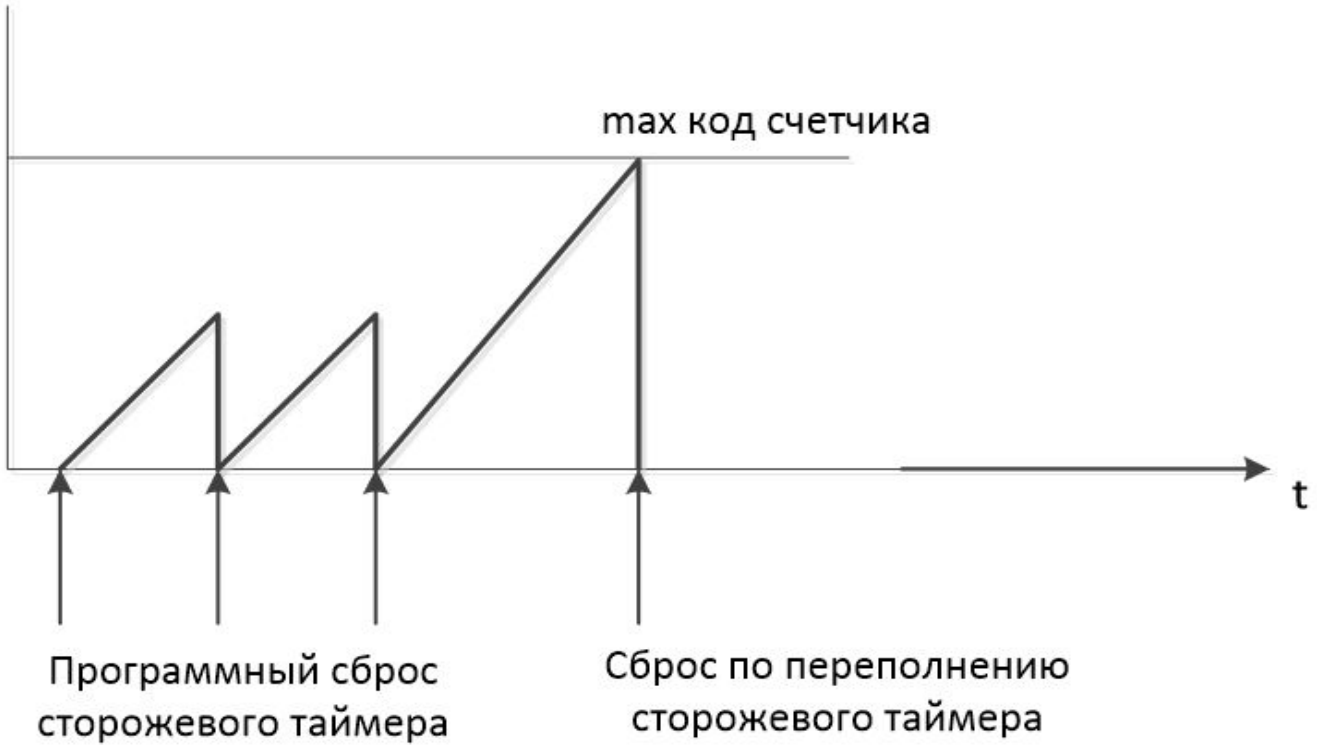
Чтобы исключить сброс по переполнению, рабочая программа должна периодически сбрасывать СЧ. Тогда переполнения не будет, и СЧ не окажет влияния на работу МК.

Если рабочая программа «зависла», то СЧ вовремя сброшен не будет, произойдет сброс по переполнению сторожевого таймера, и нормальный ход программы будет восстановлен.

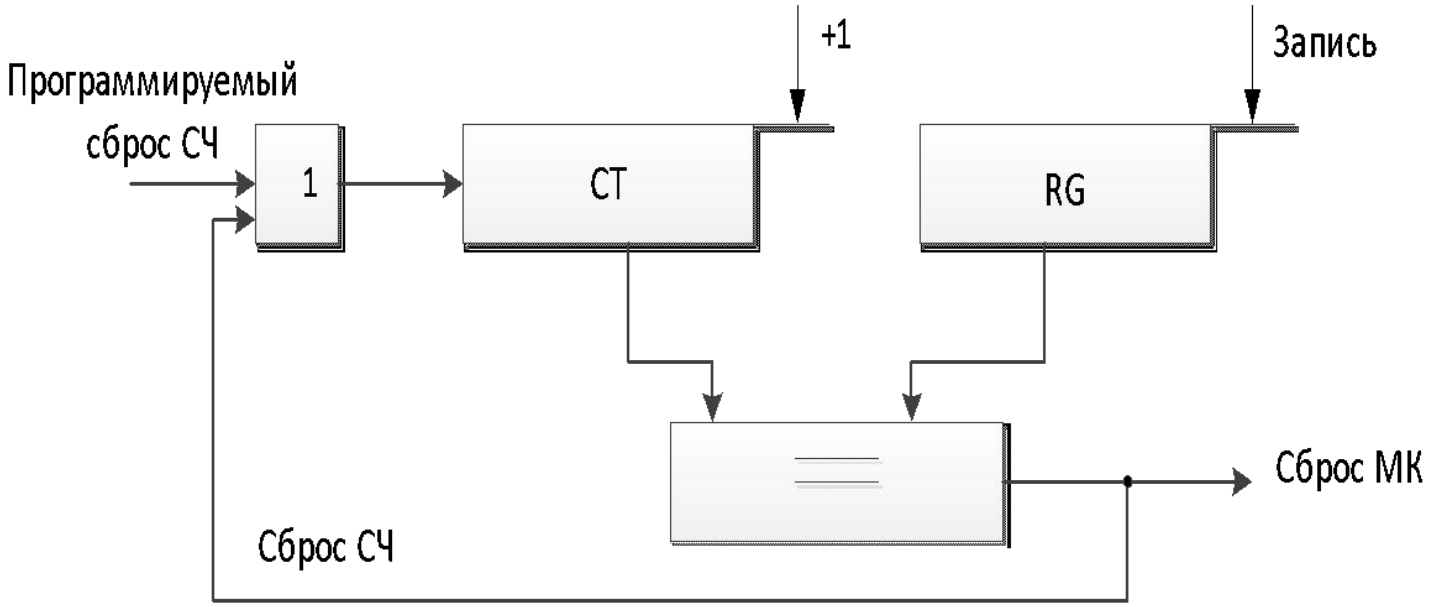
Сброс СЧ осуществляется

1)Путем исполнения спец. команды;

2)Периодически меняя значение регистра СЧ.



Временная диаграмма функционирования сторожевого таймера



Структурная схема сторожевого таймера

Синхронизация МК, схема формирования сигнала сброса

Аппаратные средства обеспечения надежной работы МК

Прикладная программа, записанная в память программ МК, должна обеспечивать его надежную работу при любых комбинациях входных сигналов. Однако в результате электромагнитных помех, колебаний напряжения питания и других внешних факторов предусмотренный разработчиком ход выполнения программы может быть нарушен.

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.

Схема формирования сигнала сброса МК

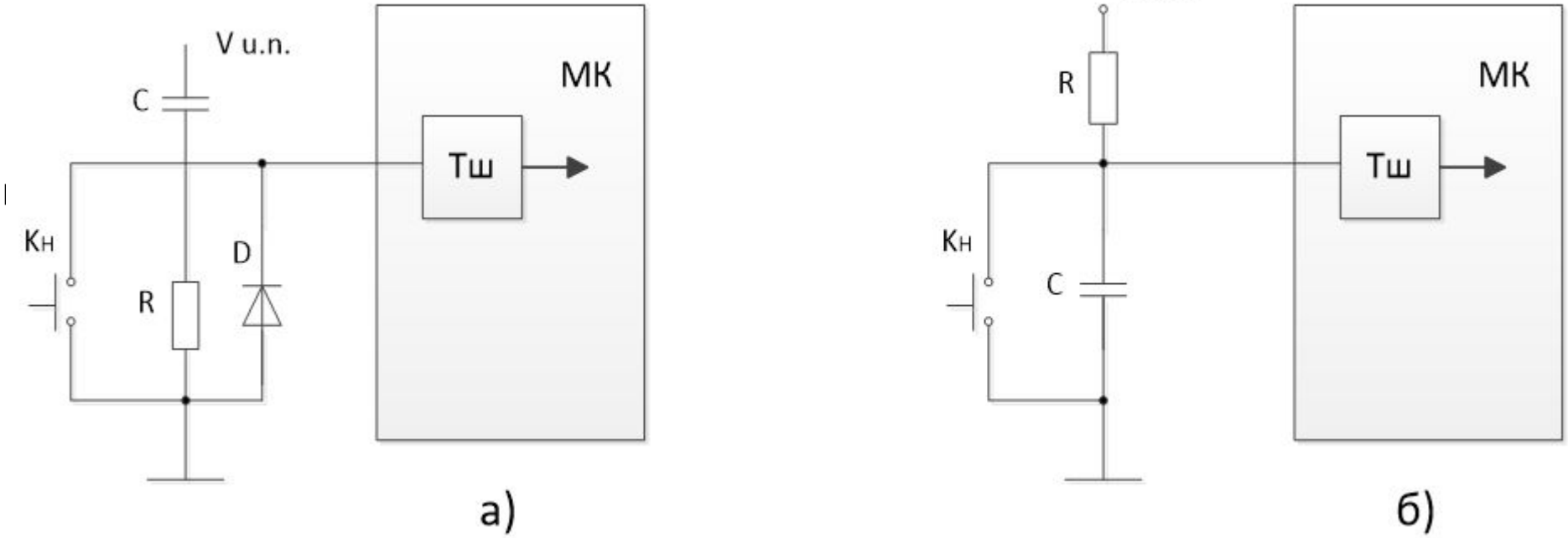
При включении напряжения питания МК должен начать выполнять записанную в памяти программу работы. На этапе нарастания напряжения питания МК принудительно переводится в начальное состояние, которое называют состоянием сброса. При этом устанавливаются в исходное состояние внутренние магистрали МК, сигналы управления и регистры специальных функций. Последние определяют начальное состояние периферийных модулей МК, которое чаще всего по умолчанию неактивно.

С целью обеспечения надежного запуска от любых источников питания с различной динамикой нарастания напряжения большинство современных МК содержат встроенный детектор напряжения питания (схема Power-On-Reset — POR), который формирует сигнал сброса при нарастании напряжения питания. В частности, входящий в состав МК семейства PIC16 таймер установления питания (PWRT) начинает отсчет времени после того, как напряжение питания пересекло уровень около 1,2…1,8 В. По истечении выдержки около 72 мс считается, что напряжение достигло номинала.

Сразу после выхода из состояния сброса МК выполняет следующие действия:

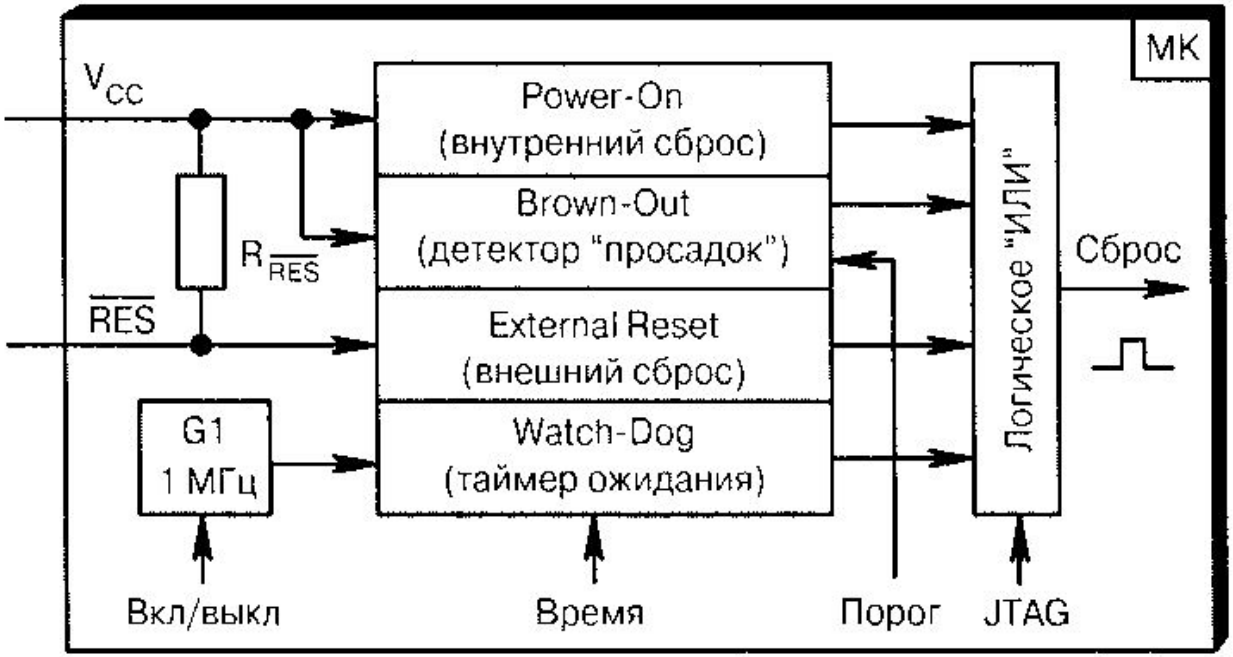
- запускает генератор синхронизации МК. Для стабилизации частоты тактирования внутренними средствами формируется задержка времени;
- считывает энергонезависимые регистры конфигурации в соответствующие регистры ОЗУ (если необходимо);
- загружает в счетчик команд адрес начала рабочей программы;
- производит выборку первой программы из памяти программ и приступает к выполнению программы.

Адрес ячейки памяти, в которой хранится код первой исполняемой команды, называют вектором начального запуска или вектором сброса. В некоторых МК этот адрес однозначно определен и приведен в техническом описании. Про такие МК говорят, что они имеют **фиксированный вектор сброса**. В других МК вектор сброса может быть произвольно определен пользователем. На этапе программирования МК необходимый вектор начального запуска записывается в ячейки с фиксированными адресами, и при выходе МК из сброса автоматически загружается в счетчик команд. О таких МК говорят, что они имеют **загружаемый вектор сброса**



Схемы формирования сигнала внешнего сброса с высоким активным уровнем (а) и с низким активным уровнем (б)

Подсистема начального сброса современных микроконтроллеров



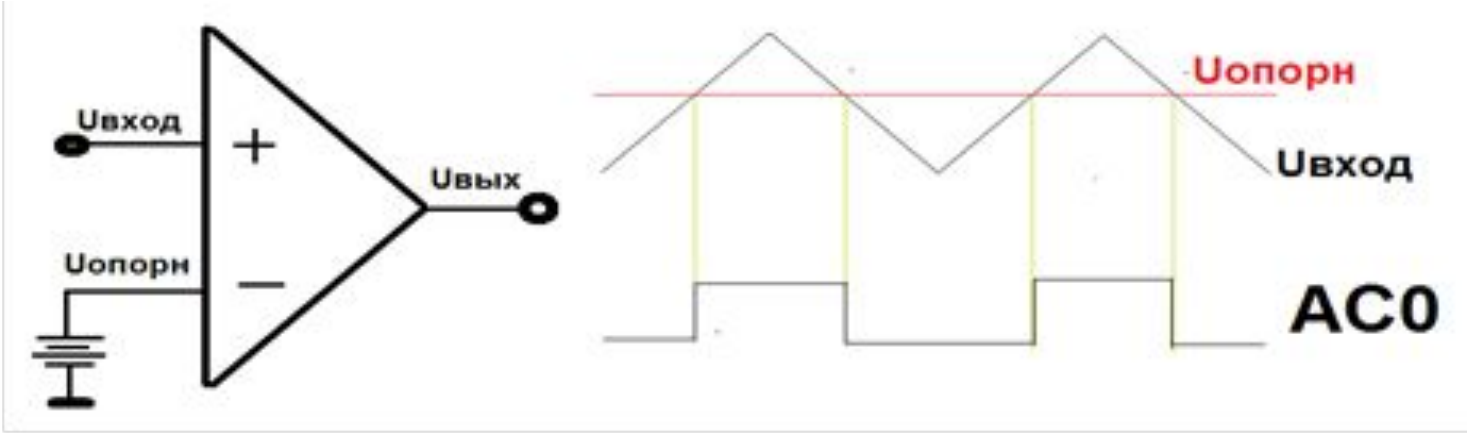
Power-On — внутренний автоматический сброс, который активизируется сразу после подачи питания;
Brown-Out — сброс от внутреннего детектора «просадок» питающего напряжения;
External Reset — внешний сброс НИЗКИМ уровнем на выводе RES;
Watch-Dog — сброс от внутреннего «сторожевого» таймера при случайной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

Модули аналогового ввода/вывода

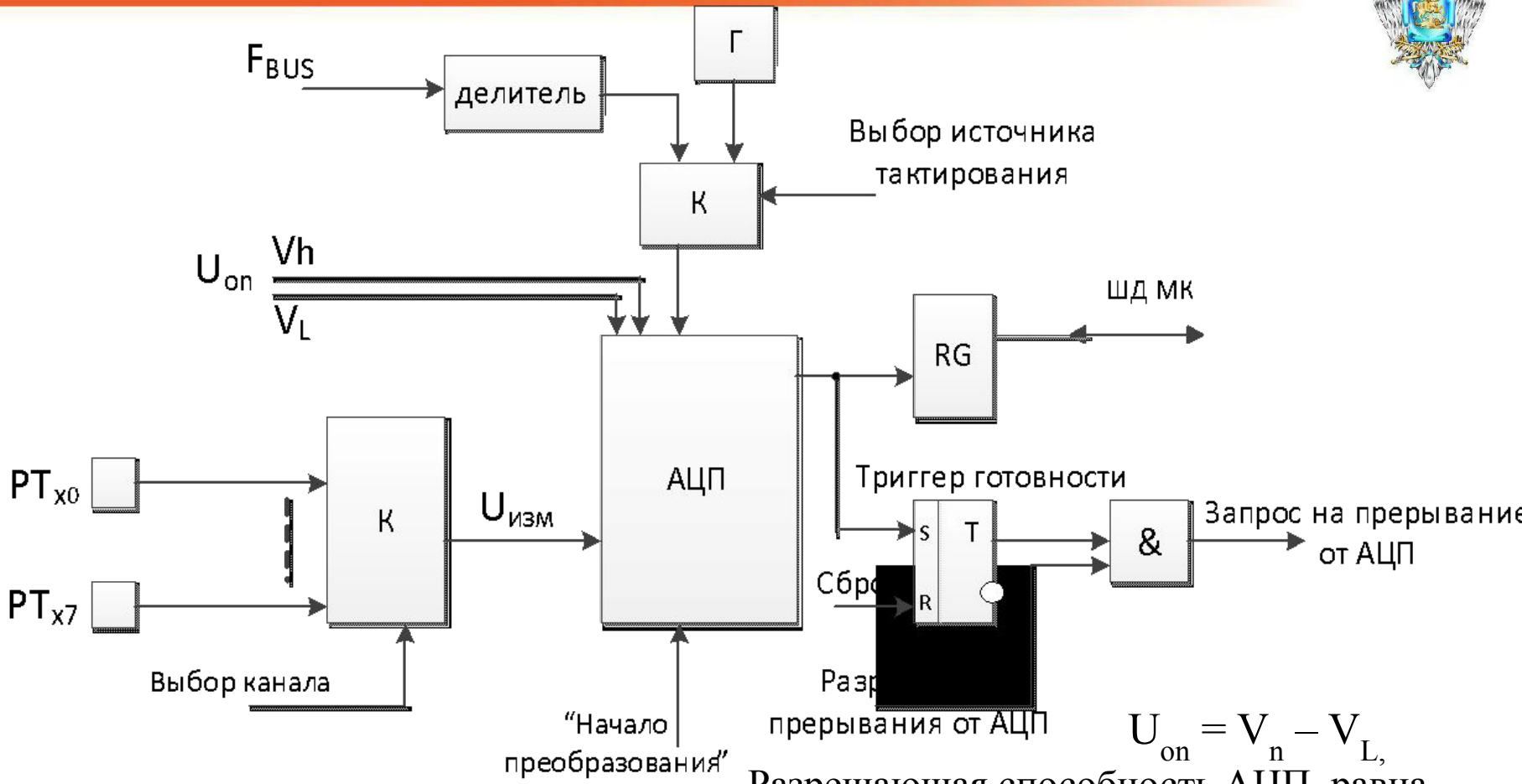
Необходимость приема и формирования аналоговых сигналов требует наличия в МК модулей аналогового ввода/вывода.

Простейшим устройством аналогового ввода в МК является встроенный компаратор напряжения.

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. 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.



Встроенный аналоговый компаратор



Функциональная схема АЦП

$$U_{on} = V_n - V_{L,}$$

Разрешающая способность АЦП равна $U_{оп}/2^n$, где n – число разрядов в результате

При $U_{изм} > V_н$ результат FF

При $U_{изм} < V_L$ результат 00

Минимизация энергопотребления, синхронизация

Минимизация энергопотребления в системах на основе МК

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 основные режимы работы

Основные режимы работы микроконтроллеров

-Активный режим (Run mode) — основной режим работы МК

-Режим ожидания (Wait mode, Idle mode или Halt mode). В этом режиме прекращает работу центральный процессор, но продолжают функционировать периферийные модули.
$P_{RAN} \sim (5\text{-}10) P_{WAIT}$;
Задержка выхода из режима Wait в режим Run составляет ~3…5 периодов синхронизации МК

-Режим останова (Stop mode, Sleep mode или Power Down mode). В этом режиме прекращает работу как центральный процессор, так и большинство периферийных модулей.
$P_{RAN} \sim 1000\ P_{STOP}$
Задержка выхода из режима Stop в режим Run составляет ~1000 периодов синхронизации МК

-Экономичный режим (Power save) – в AVR. Продолжает работать только генератор таймера, временная база сохраняется, все остальные функции выключены.

Основные группы МК в зависимости от диапазона питающих напряжений

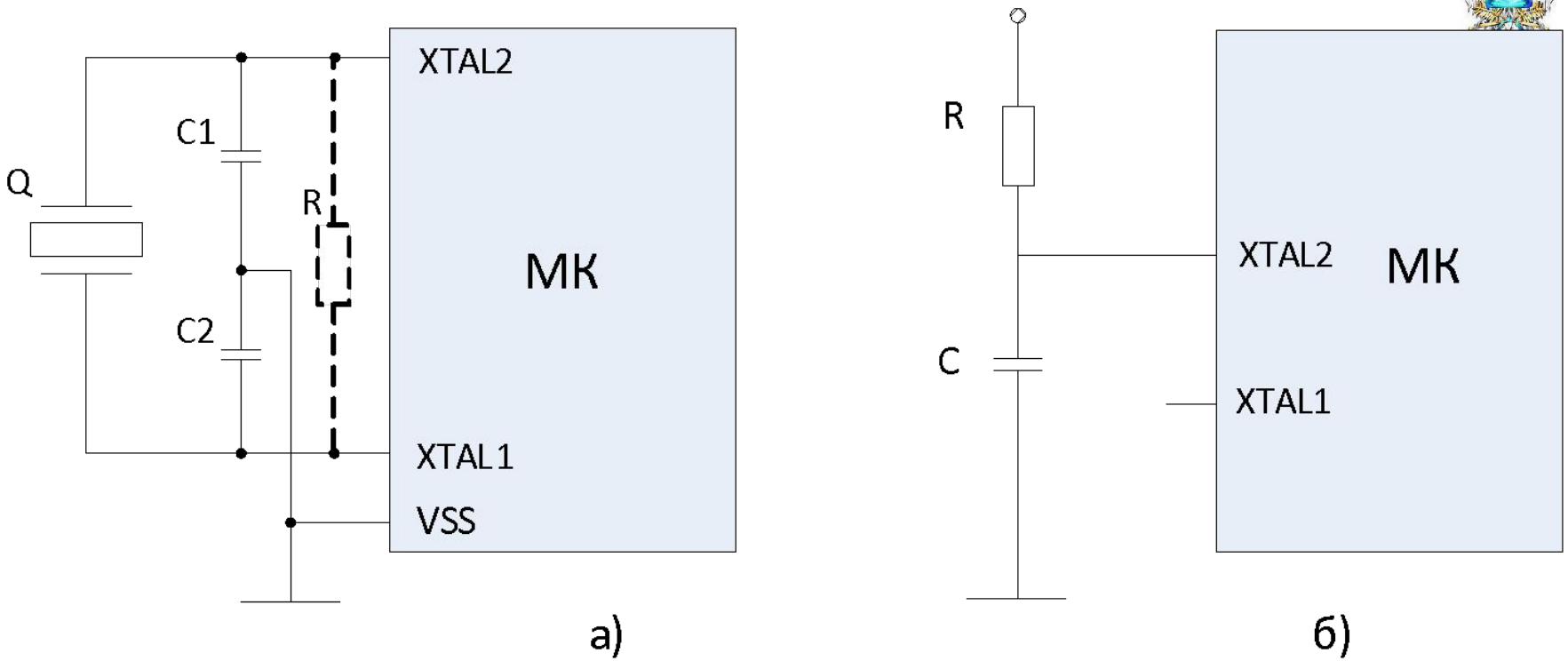
-МК с напряжением питания 5,0 В±10%. Эти МК предназначены, как правило, для работы в составе устройств с питанием от промышленной или бытовой сети, имеют развитые функциональные возможности и высокий уровень энергопотребления

-МК 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.

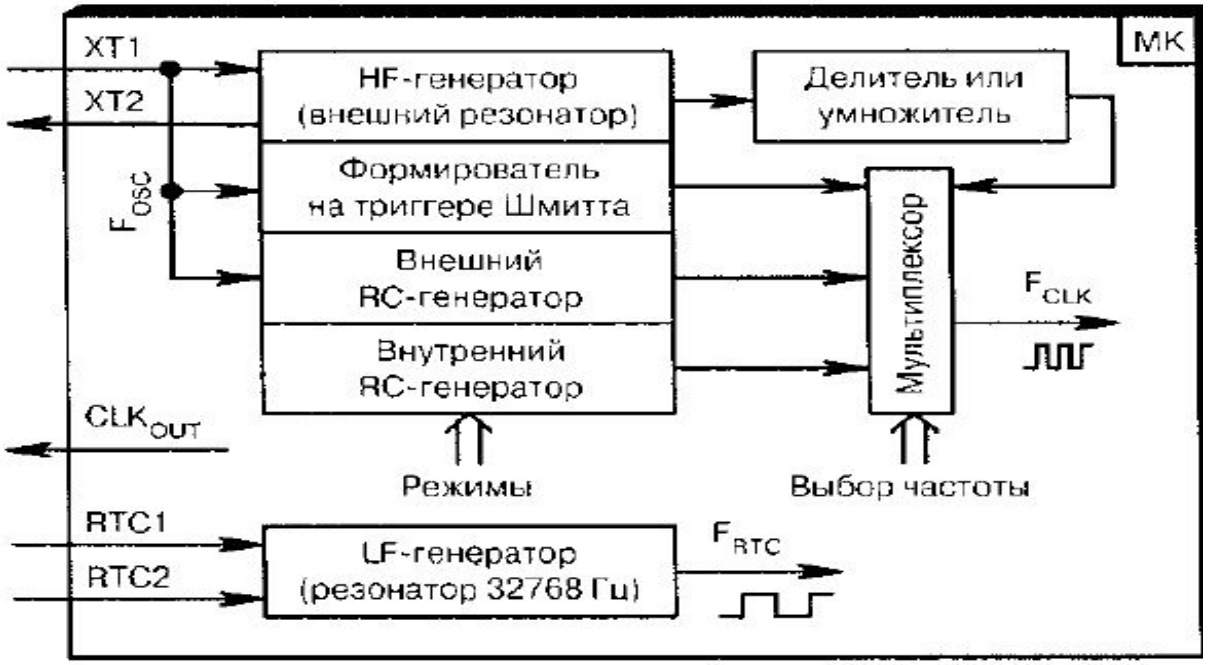
Снижение напряжения питания понижает мощность потребления МК. Выигрыш в потребляемой мощности сопровождается снижением производительности системы.

(-)Произв.↓ ← f такт ↓ ← U ип ↓ → I потр ↓ → P потр ↓ (+)



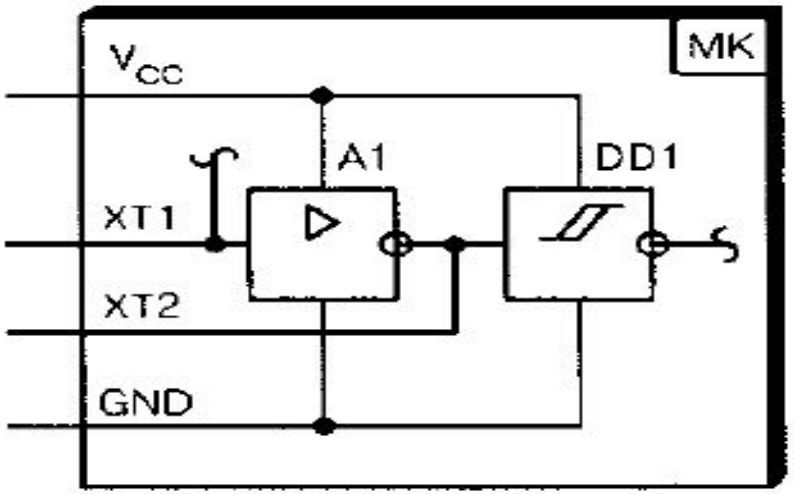
Тактирование с использованием кварцевого или керамического резонатора (а) и с использованием RC-цепи (б)

подсистема синхронизации



На структурной схеме имеется несколько встроенных генераторных узлов.
HF (High Frequency) — высокочастотный, LF (Low Frequency) — низкочастотный, CLK — тактирование.

внутреннее устройство генератора



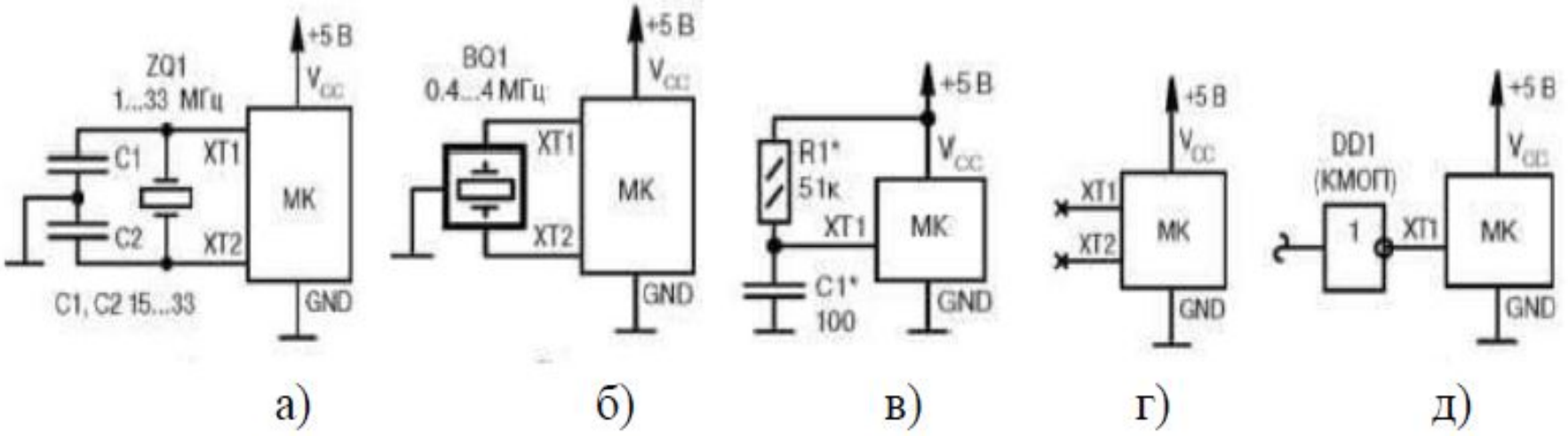
инвертирующий усилитель А1;
буферный логический формирователь на триггере Шмитта DD1.
Если к выводам XT1, XT2 подключить времязадающий элемент, то в системе возникнут условия для автогенерации

В МК произошло функциональное разделение генератора синхронизации, который выделился в отдельный модуль, и схемы формирования последовательности импульсов для тактирования ЦП и межмодульных магистралей, которая входит в состав ядра. Можно выбирать внешний времязадающий элемент.

Кварцевый резонатор задает высокую точность и стабильность тактовой частоты (разброс частот менее 0,01%), что обеспечивает точный ход часов реального времени или организацию интерфейса с другими устройствами. Недостатками кварцевого резонатора являются его низкая механическая прочность (высокая хрупкость) и относительно высокая стоимость.

Керамические резонаторы имеют разброс частот порядка нескольких десятых долей процента (обычно около 0,5 %), однако более стойки к ударной нагрузке Самым дешевым способом задания тактовой частоты МК является использование **внешней RC-цепи** (разброс частот может доходить до десятков процентов)

Типовые схемы формирования тактовой частоты



Режимы работы основного канала синхронизации

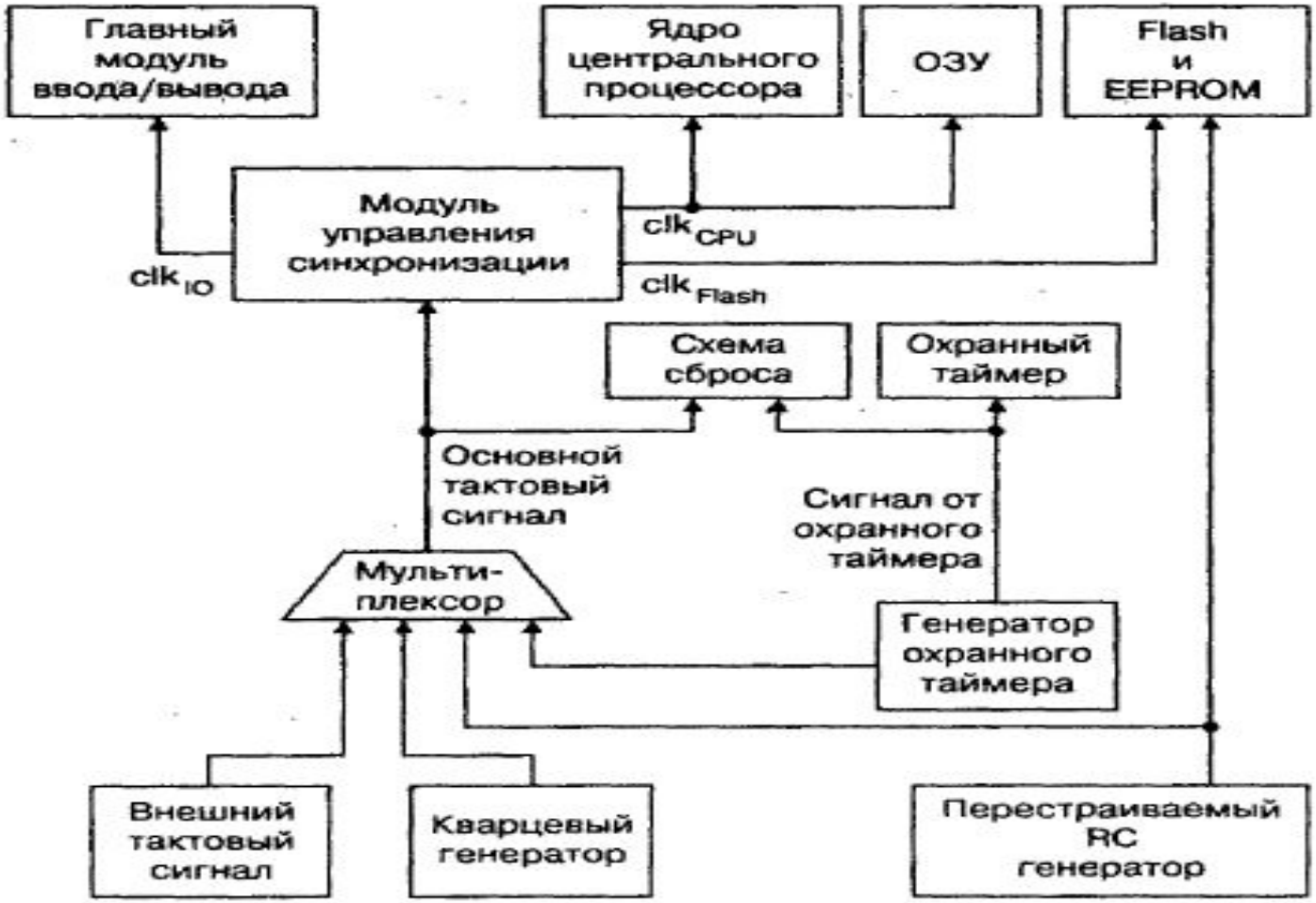
- а) от высокочастотного кварцевого резонатора 1...33 МГц ;
- б) от среднечастотного керамического резонатора 0.4...4 МГц ;
- в) от внешней RC-цепочки 0.4.. .12 МГц ;
- г) от внутреннего RC-генератора 1; 2; 4; 8 МГц ;
- д) от внешних синхроимпульсов 0...40 МГц.

Пример схемы синхронизации современных микроконтроллеров AVR

Источники тактового сигнала для микроконтроллеров AVR

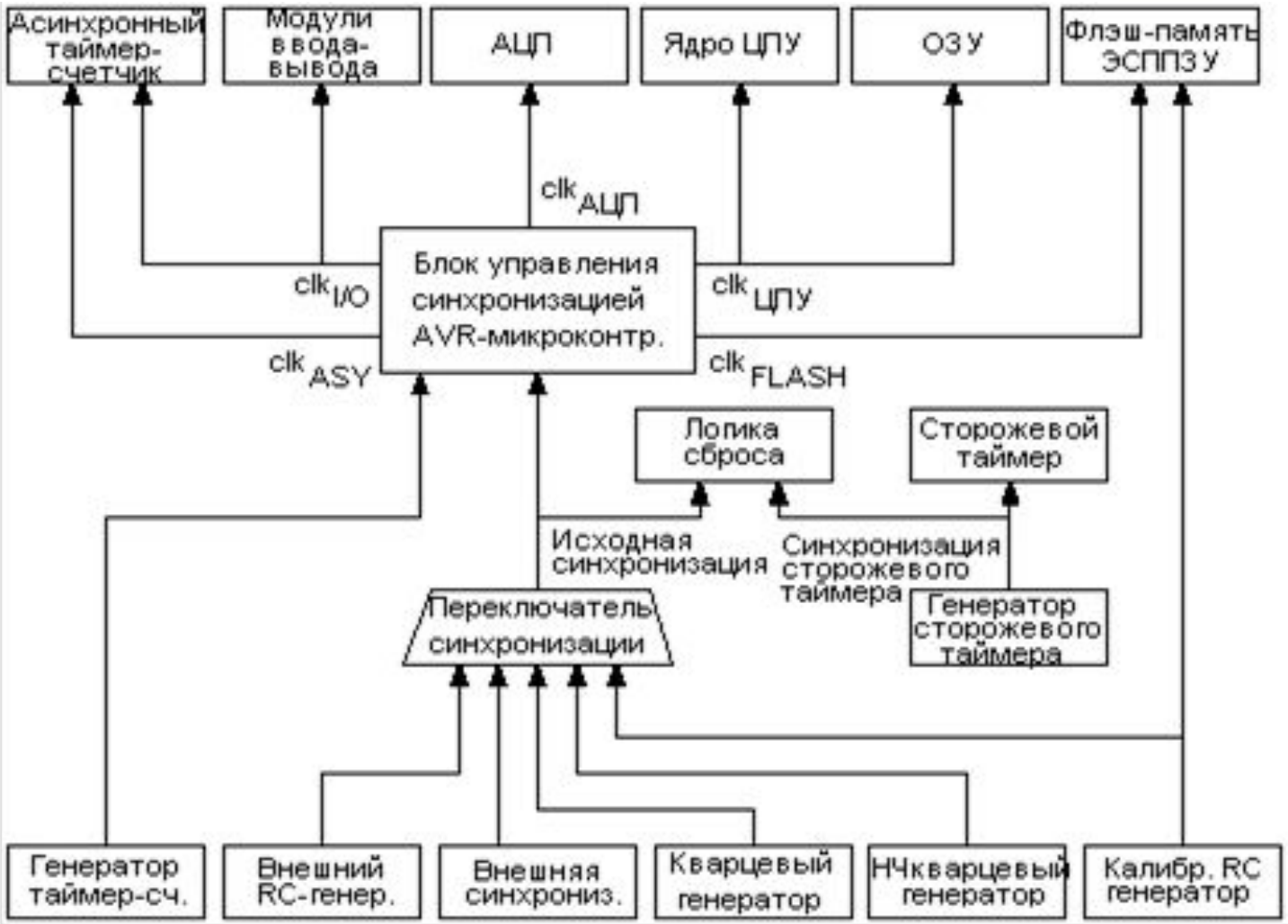
1)Настраиваемый внутренний RC-генератор
2)Внешний кварцевый резонатор
3)Генератор сторожевого таймера
4)Внешний тактовый сигнал

После выбора источника он является единственным в микроконтроллере AVR. Когда МК пробуждается после спящего режима, выбранный источник запускается в самом начале пробуждения

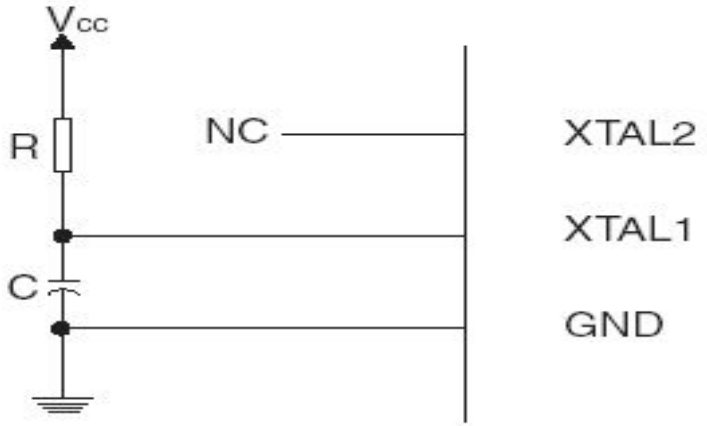


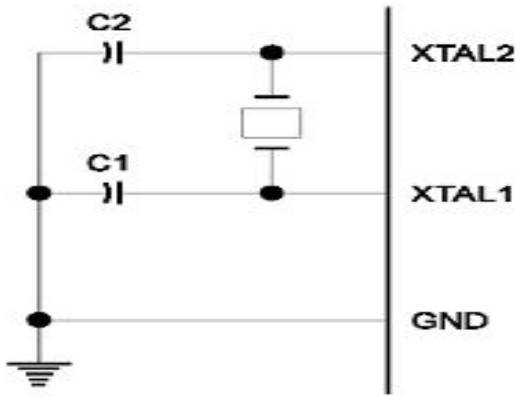
4; 8 Мгц

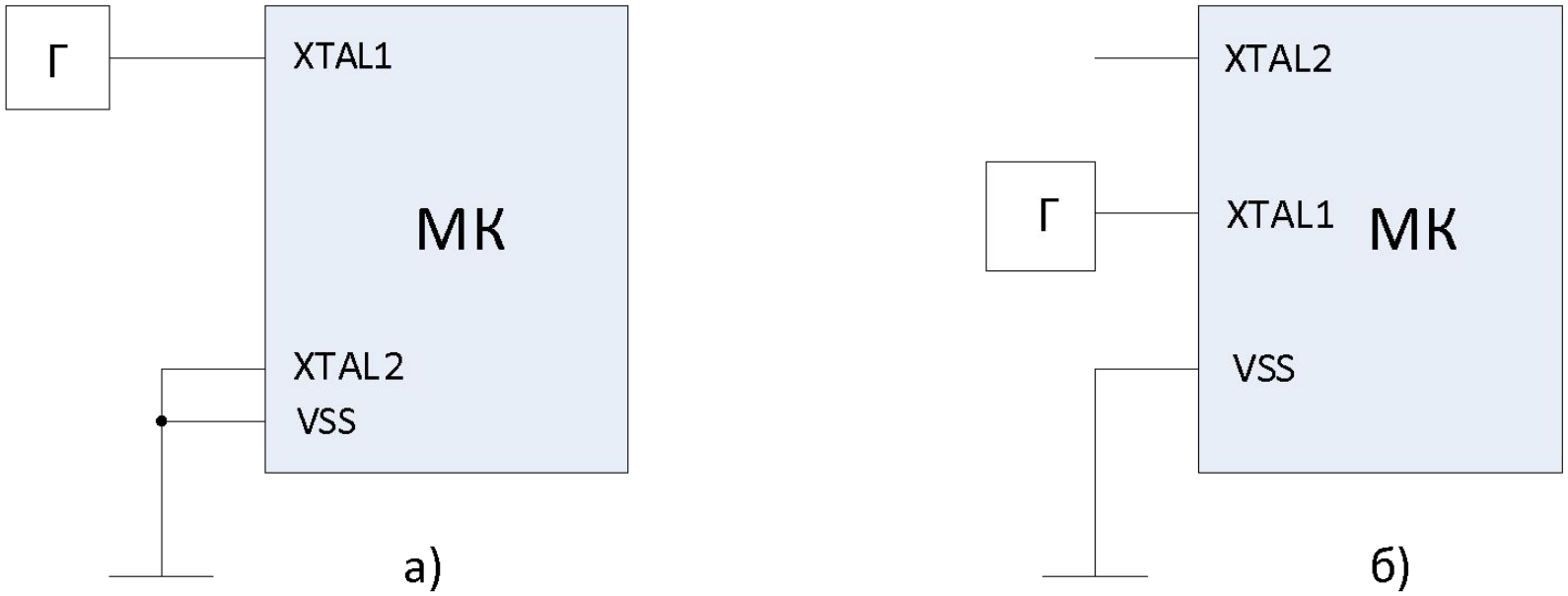
Схема синхронизации микроконтроллеров AVR



2-я схема синхронизации микроконтроллеров AVR


Vcc
R
C
NC
XTAL2
XTAL1
GND


C2
C1
XTAL2
XTAL1
GND



Тактирование с использованием внешнего синхрогенератора для n-МОП технологии (а) и для k-МОП технологии (б)

Производительность (MIPS) = 1/ t ком (мкс) определяется по самой быстрой команде пересылки типа «регистр-регистр»

f_{xcLK} - частота времязадающего элемента – генератора синхронизации

f_{BUS} - тактовая частота обмена по внутренним магистралям – ША и ШД

Соотношение f_{xcLK} / f_{BUS} индивидуально для каждого ядра

Микроконтроллер	f xcLK / f BUS	архитектура
Intel MCS-51	12	CISC
Microchip PIC-16, 17, 18	4	RISC
Atmel AVR	1	RISC
HC08 Тактирование 1)от генератора кварцевого резонатора 2) От умножителя частоты	4 f BUS >f xcLK	CISC

RISC МК имеют большую производительность по сравнению с CISC при одной и той же f BUS ,т.к. выполняют команды за один машинный цикл, а CISC – за несколько (от 1 до 3 для команд типа регистр-регистр).

Для RISC tком = 1/ f BUS , следовательно, производительность (П) определяется по f BUS

PIC16 П=5 MIPS, т.к. f BUS =5МГц
AVR П=.20 MIPS, т.к. f BUS =20МГц

Модуль прерываний МК

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.

Могут быть 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.

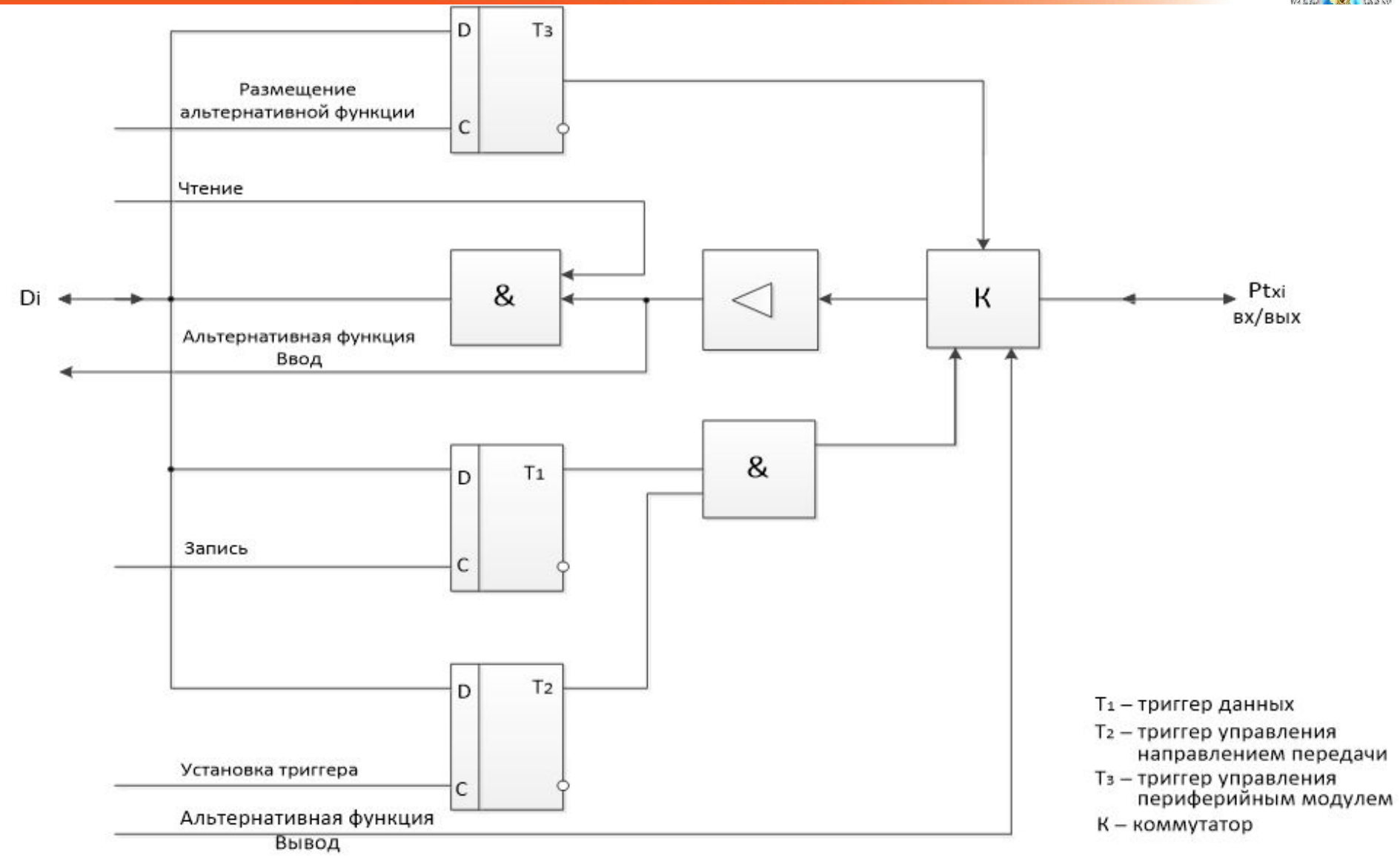
Поллинг - последовательный опрос по старшинству источников одного уровня.

Параллельный и последовательный ввод-вывод в МК

Порты параллельного ввода/вывода

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


D
T3
Размещение
альтернативной функции
C
Чтение
&
Di
Альтернативная функция
Ввод
К
Ptxi
вх/вых
D
T1
&
Запись
C
D
T2
Установка триггера
C
Альтернативная функция
Вывод
T1 – триггер данных
T2 – триггер управления
направлением передачи
T3 – триггер управления
периферийным модулем
К – коммутатор

Модули последовательного ввода/вывода

Задачи, которые решаются средствами модуля контроллера последовательного ввода/вывода

- 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;
- связь с внешними по отношению к МК периферийными ИС, а также с датчиками физических величин с последовательным выходом. Для этих целей используются интерфейсы I^2C **(Inter-Integrated Circuit)** , SPI **(Serial Peripheral Interface)**, а также нестандартные протоколы обмена;
- интерфейс связи с локальной сетью в мультимикроконтроллерных системах. В системах с числом МК до пяти обычно используются сети на основе интерфейсов I^2C, RS-232C и RS-485 с собственными сетевыми протоколами высокого уровня. В более сложных системах все более популярным становится протокол CAN**(Controller Area Network).** .

МОДУЛЬ UART

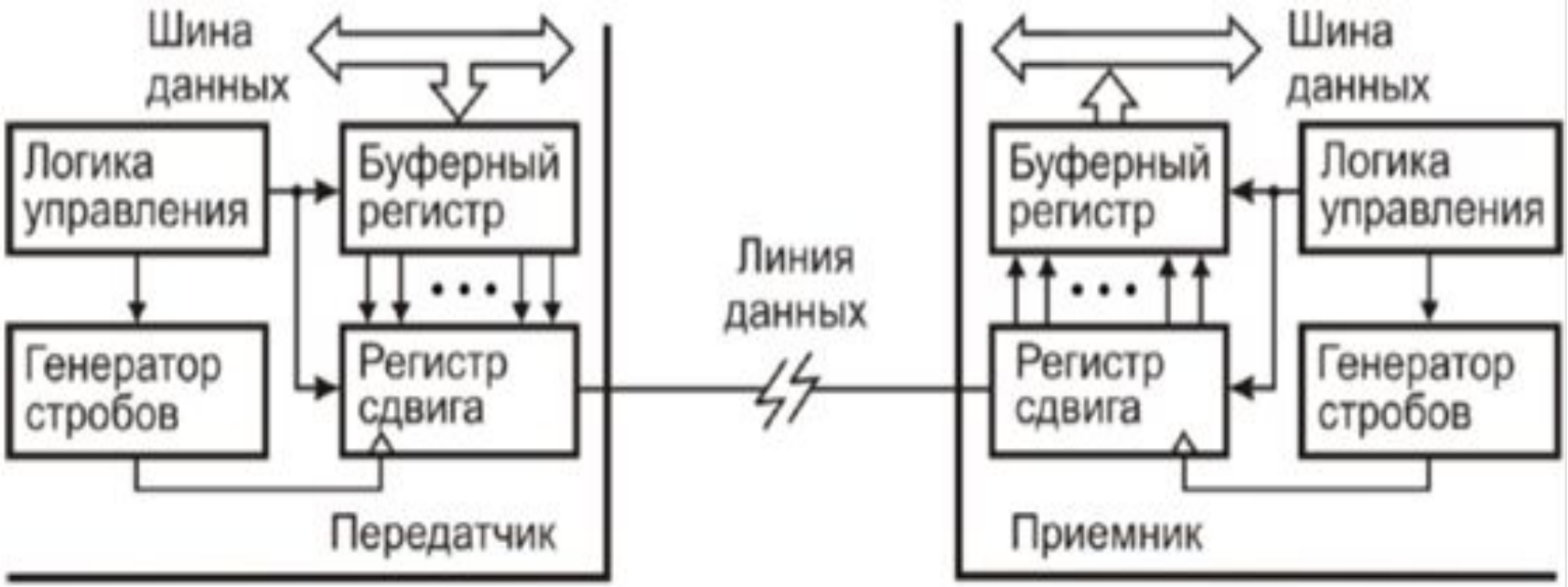
. Среди различных типов встроенных контроллеров последовательного обмена, которые входят в состав тех или иных 8-разрядных МК, сложился стандарт "де-факто" — модуль **UART (Universal Asynchronous Receiver and Transmitter).** UART — это универсальный асинхронный приемопередатчик

Модуль UART предназначен для связи контроллеров друг с другом и связи контроллеров с компьютером. В отличии от других интерфейсов контроллеров, он не предназначен для подключения периферийного оборудования

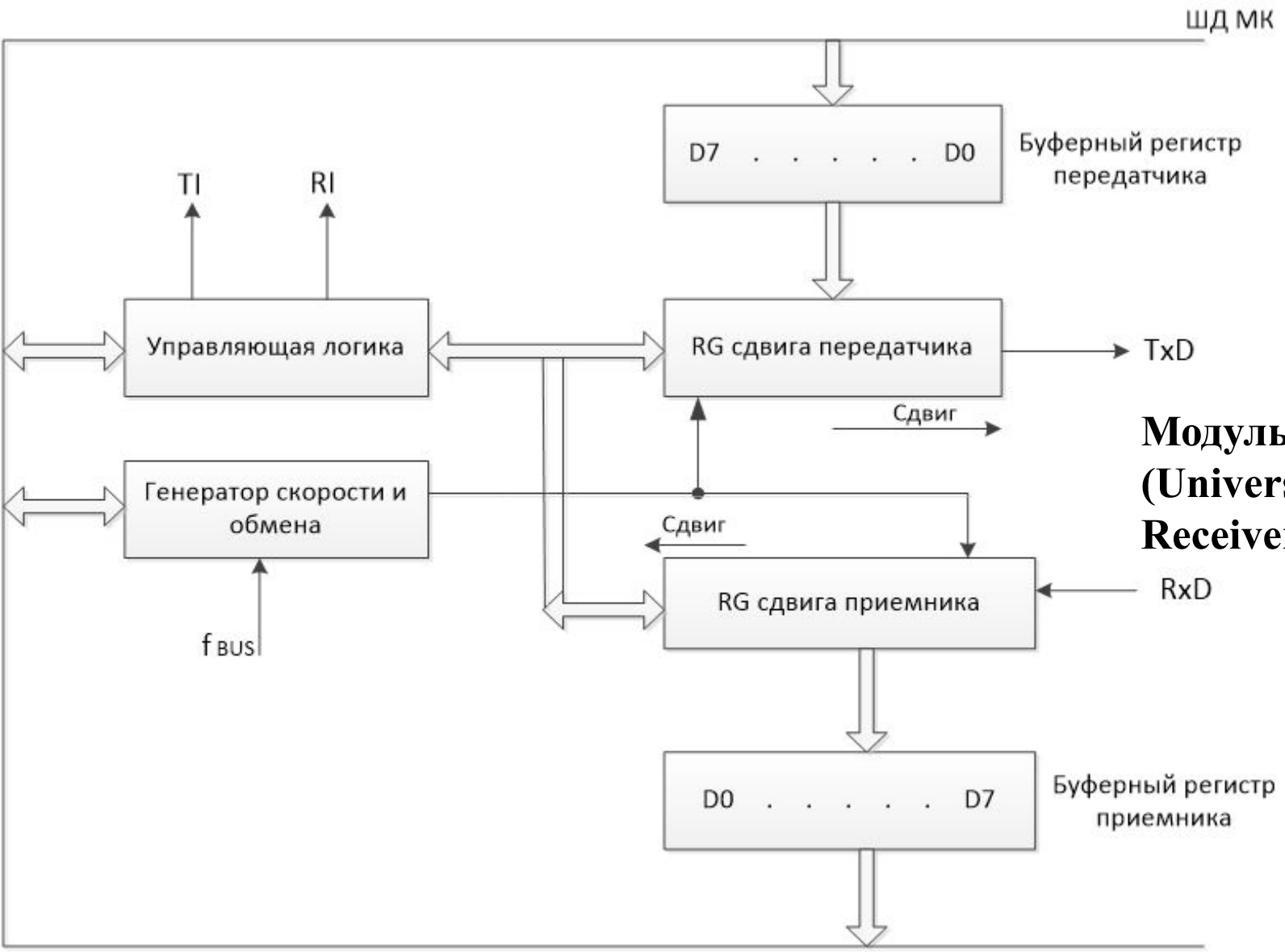
Изначально UART предназначался для связи двух устройств, по принципу «точка-точка». Впоследствии были созданы физические уровни, которые позволяют связывать более двух UART по принципу «один говорит — несколько слушают». Такие физические уровни называют *сетевыми* Логика UART обычно позволяет производить одновременную передачу и прием. Эта способность обозначается словом *дуплекс*. полнодуплексная работа (регистры приемника и передатчика разделены)

Другие названия:

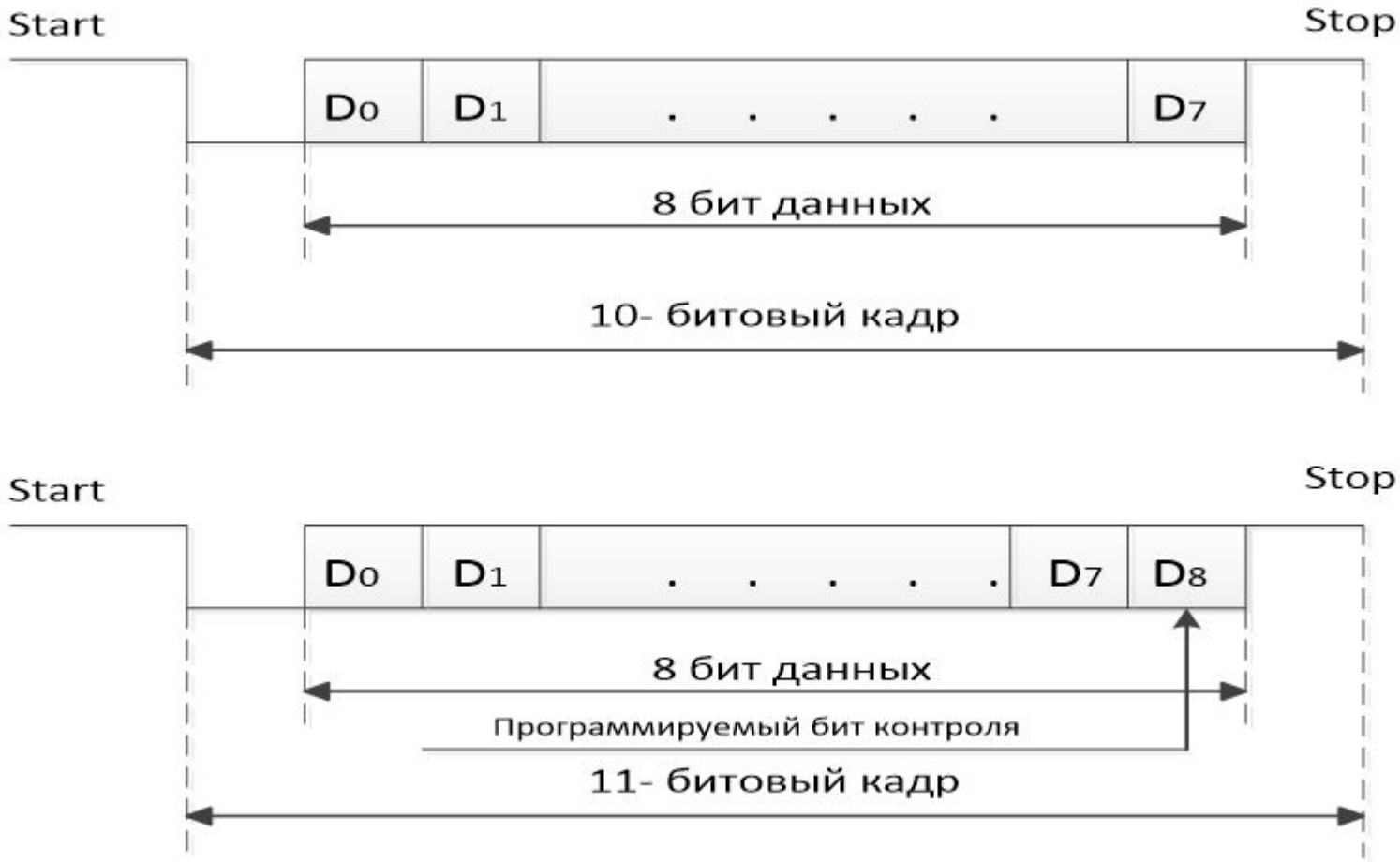
USART – Universal Synchronous/Asynchronous Receiver and Transmitter, SCI – Serial Communication Interface, USI – Universal Serial Interface



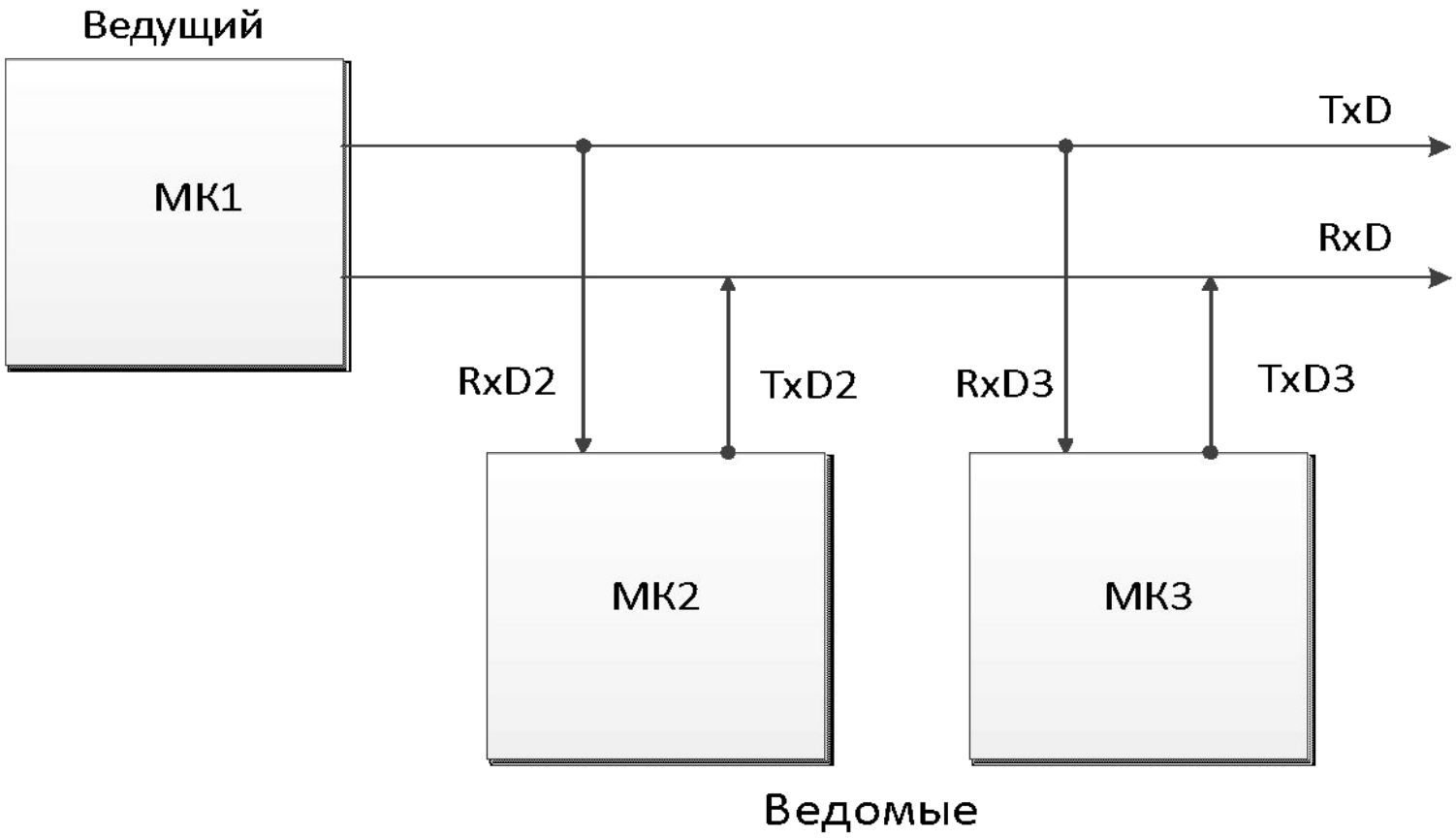
Для асинхронного режима принят следующий ряд стандартных скоростей обмена: 50, 75, 110, 150, 300, 600, 1200, 2400, 4800, 9600, 19 200, 38 400, 57 600 и 115 200 бит/с.



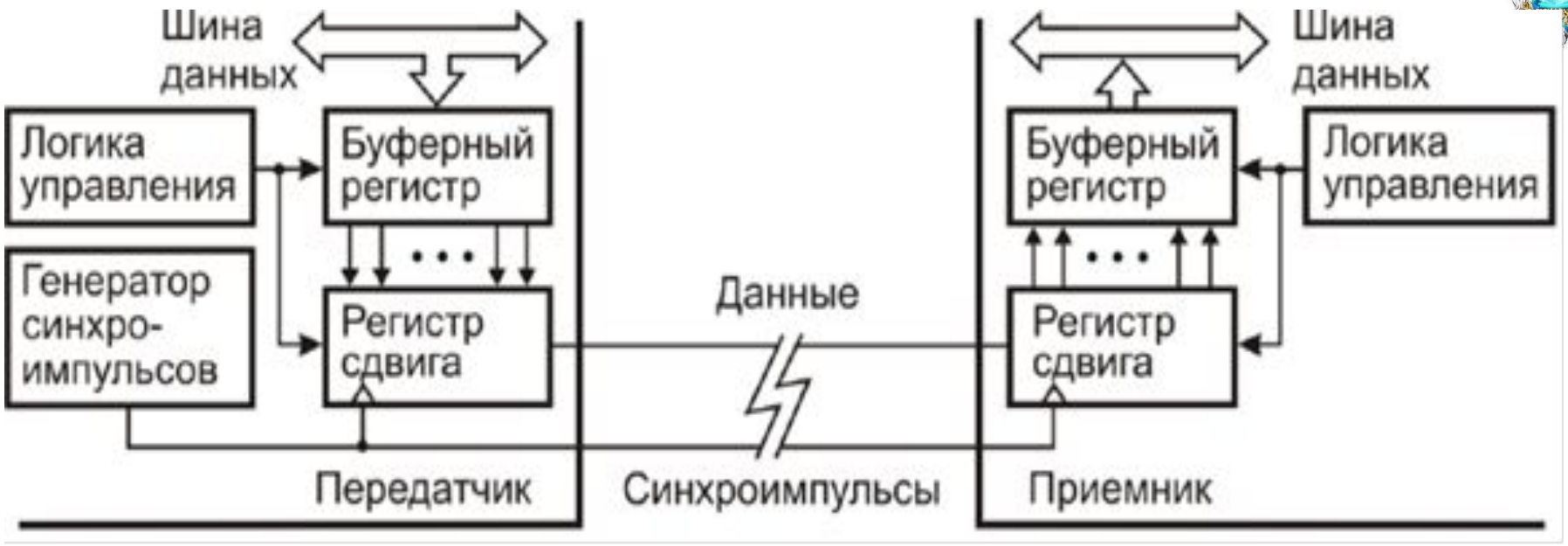
Модуль UART (Universal Asynchrono Receiver-Transmitter)

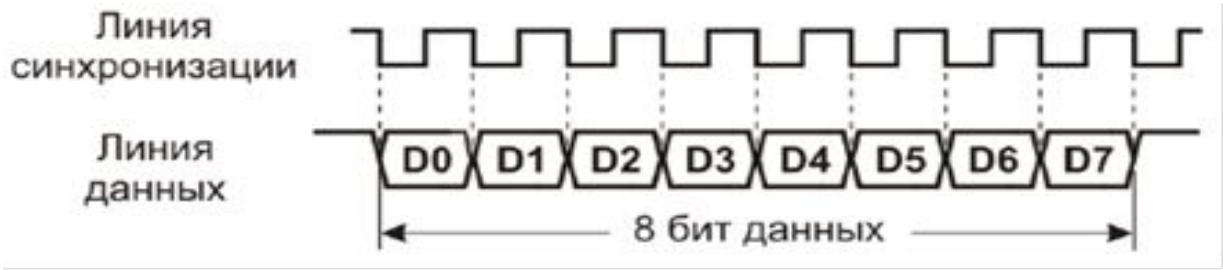


Два кадра асинхронного обмена модуля UART



Микроконтроллерная система, использующая модуль UART





Организация синхронной передачи данных

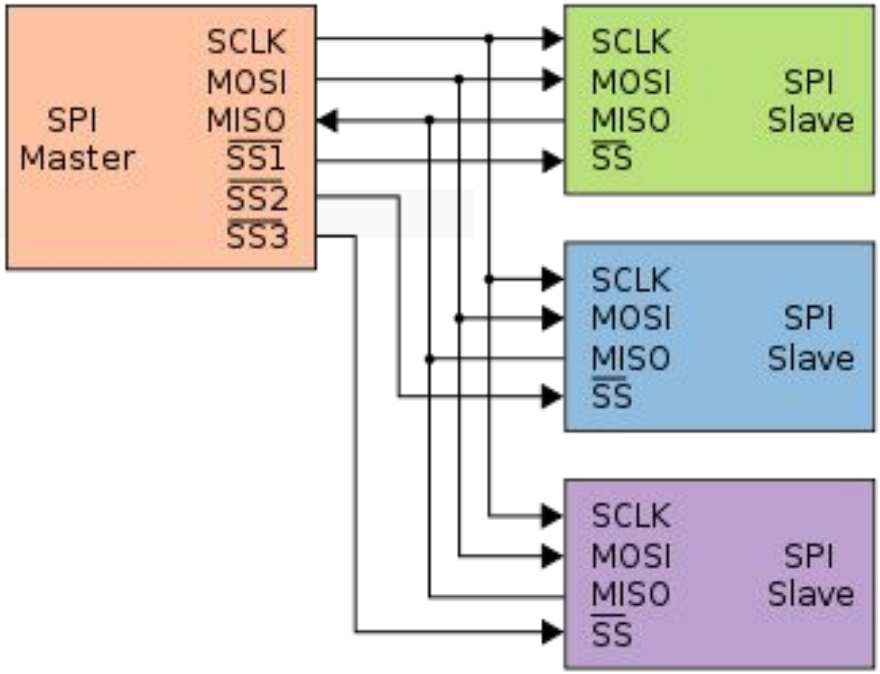
Устройства, которые имеют на своём борту UART, по часовой стрелке: мышка, ридер-эмулятор SMART-карт, КПК Palm m105, отладочная плата для микроконтроллера ATtiny2313 (или AT89C2051), модем.

Видов UART существует великое множество. главное отличие интерфейсов состоит в среде и способе передаче данных. Данные могут передаваться даже по оптоволокну.

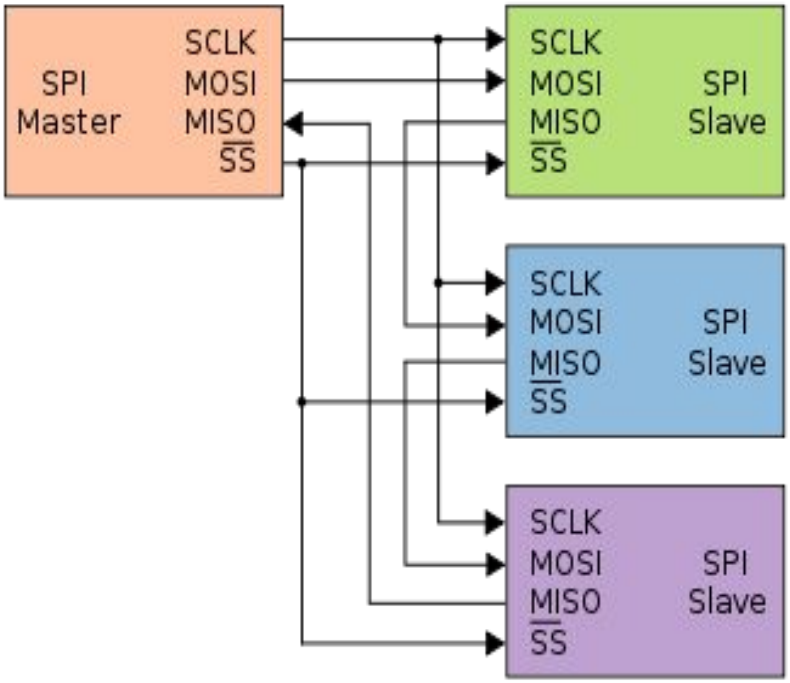
Интерфейс SPI

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.

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.



Радиальная структура связи с несколькими ведомыми устройствами через SPI



Кольцевая структура связи с несколькими ведомыми устройствами через SPI

Микроконтроллерные системы, использующие модуль SPI (Serial Peripheral Interface)

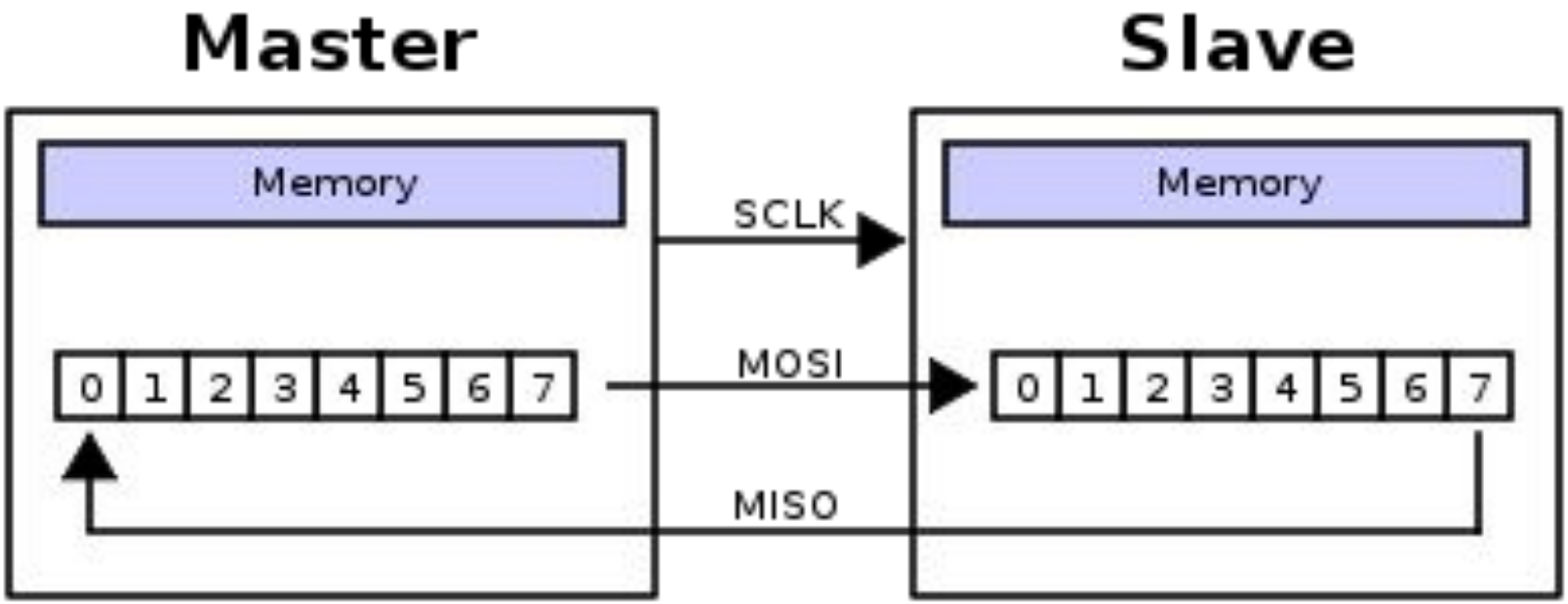
В SPI используются четыре цифровых сигнала:

MOSI — выход ведущего, вход ведомого (*Master Out Slave In*).
Служит для передачи данных от ведущего устройства ведомому.

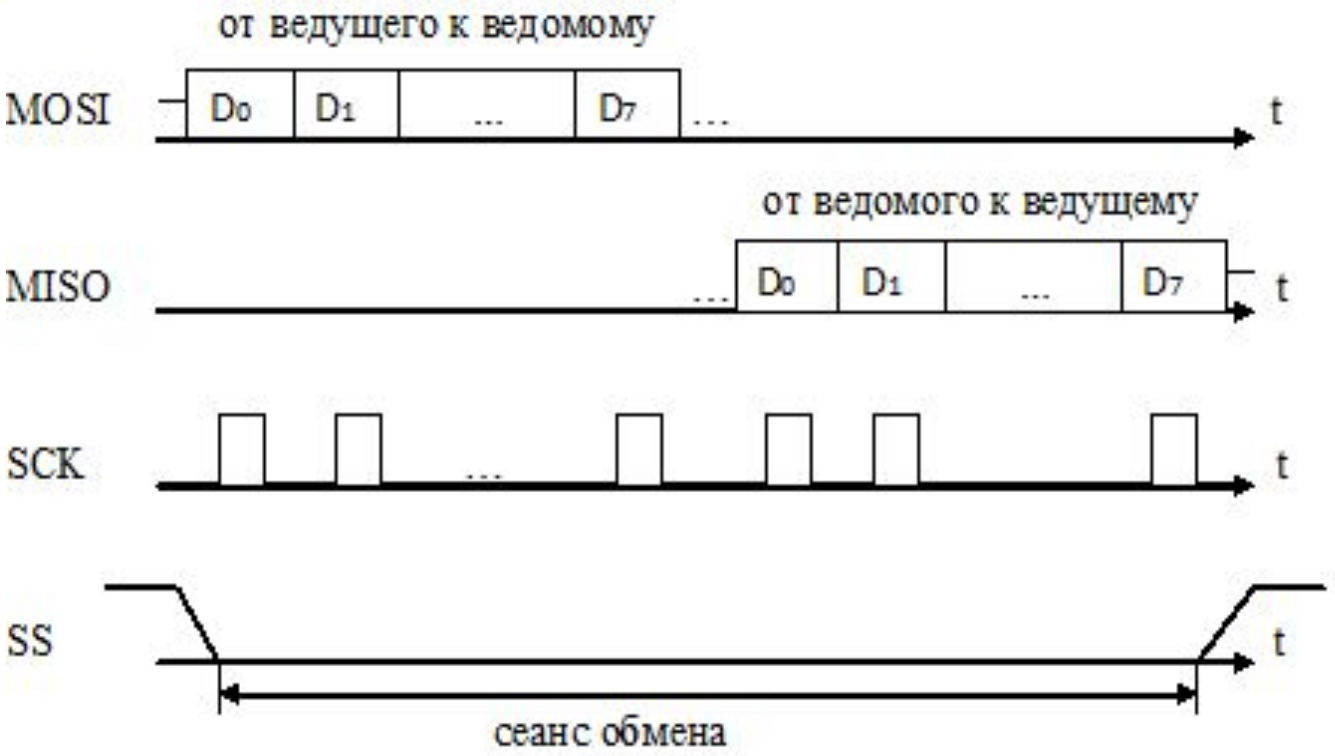
MISO — вход ведущего, выход ведомого (*Master In Slave Out*).
Служит для передачи данных от ведомого устройства ведущему.

SCLK — последовательный тактовый сигнал (*Serial Clock*).
Служит для передачи тактового сигнала для ведомых устройств.

CS или SS — выбор микросхемы, выбор ведомого (. *Chip Select, Slave Select*)

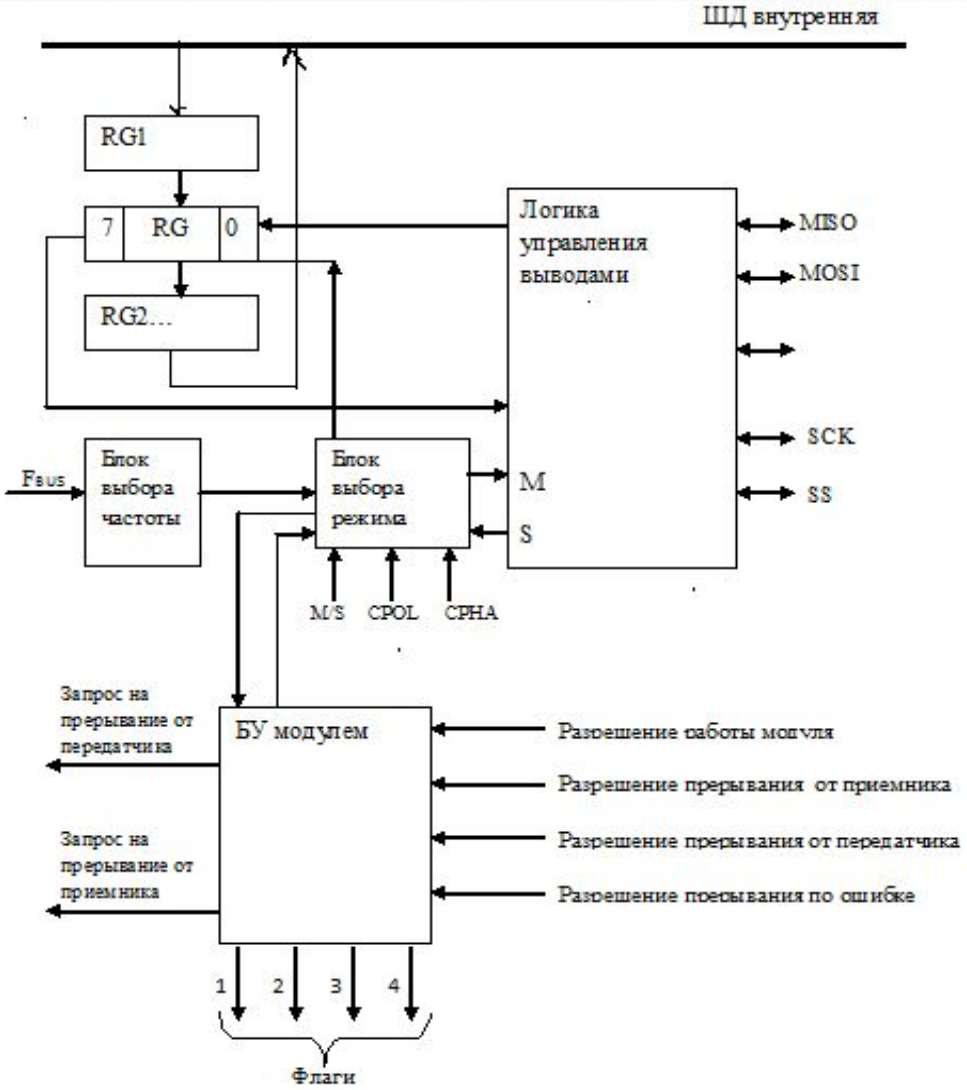


Типичная структура связей и линий интерфейса SPI



Обобщенная временная диаграмма обмена по интерфейсу SPI

Модуль SPI, схема функциональная



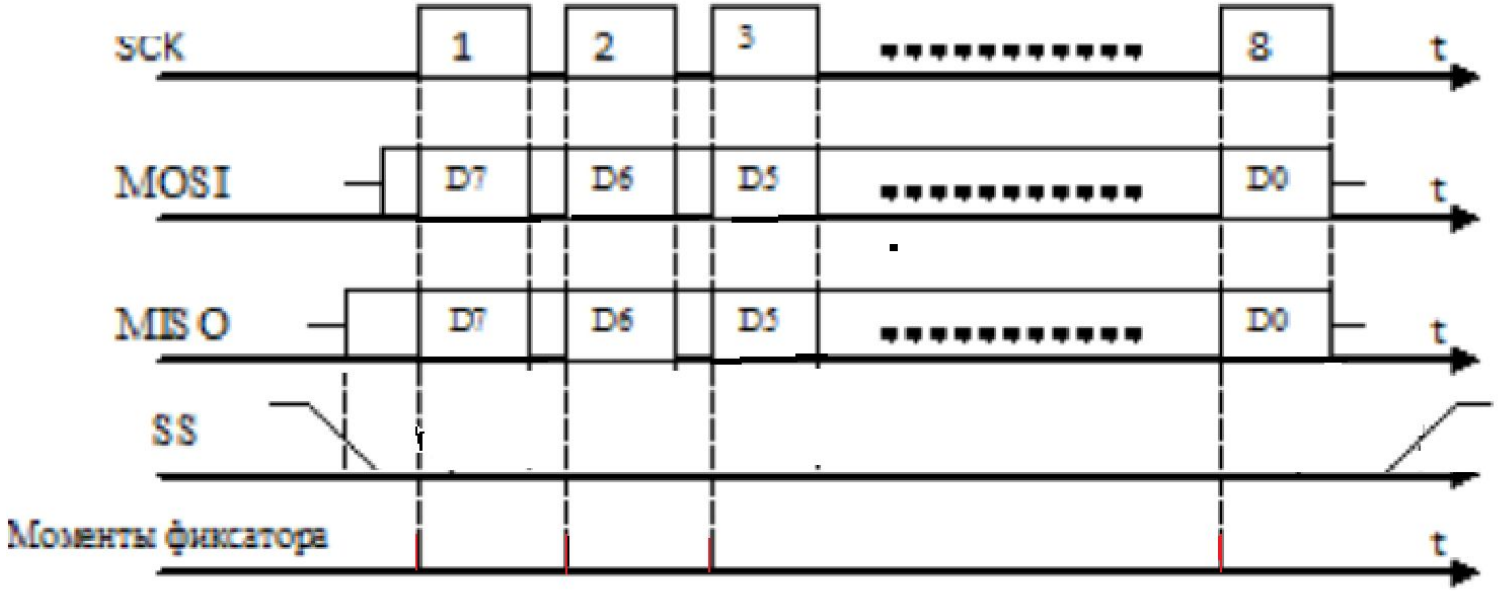
1.флаг ошибки приема
2.нарушение режима контроля
3.завершение приема байта данных
4. готовность к новому приему данных

Режимы работы интерфейса SPI

Возможны четыре комбинации фазы (CPHA) и полярности (CPOL) сигнала SCLK по отношению к сигналам данных. Режимы работы определяются комбинацией бит CPHA и CPOL:
CPOL = 0 — сигнал синхронизации начинается с низкого уровня;
CPOL = 1 — сигнал синхронизации начинается с высокого уровня;
CPHA = 0 — выборка данных производится по переднему фронту сигнала синхронизации;
CPHA = 1 — выборка данных производится по заднему фронту сигнала синхронизации.
Для обозначения режимов работы интерфейса SPI принято следующее соглашение:
режим 0 (CPOL = 0, CPHA = 0);
режим 1 (CPOL = 0, CPHA = 1);
режим 2 (CPOL = 1, CPHA = 0);
режим 3 (CPOL = 1, CPHA = 1).

Временная диаграмма работы модуля SPI для режима 00 и 10

CPHA=0
CPOL=0

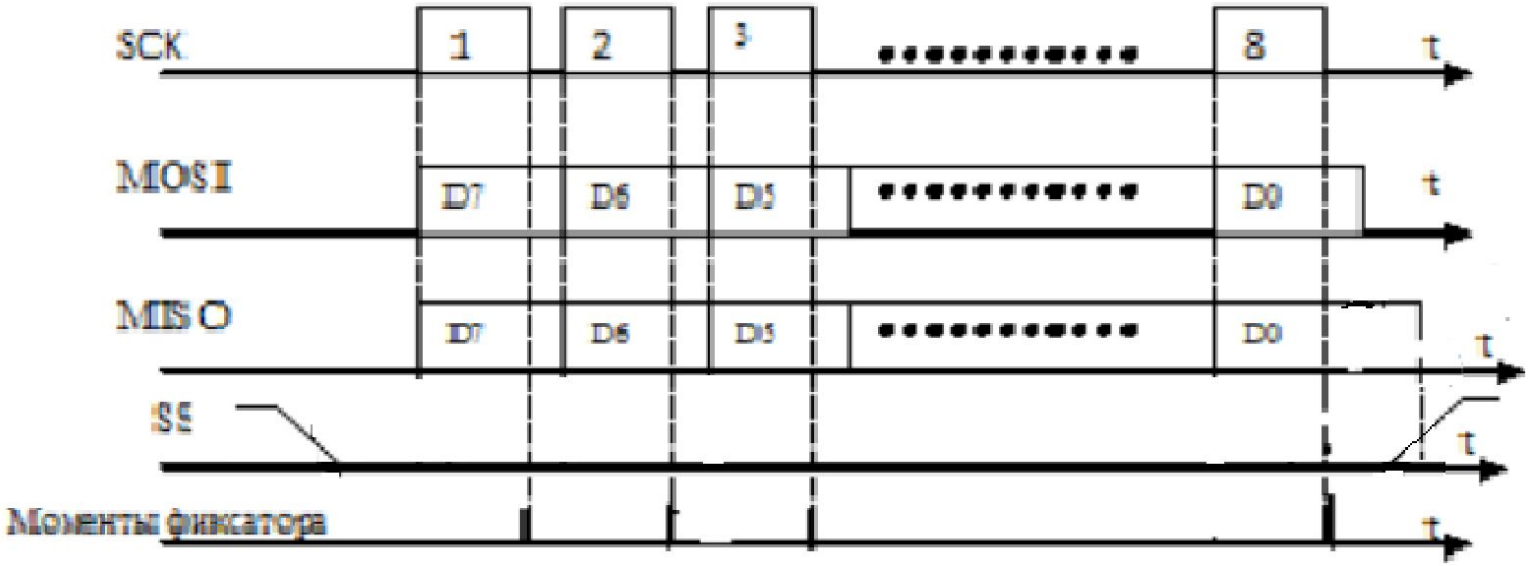


MOSI – от ведущего к ведомому

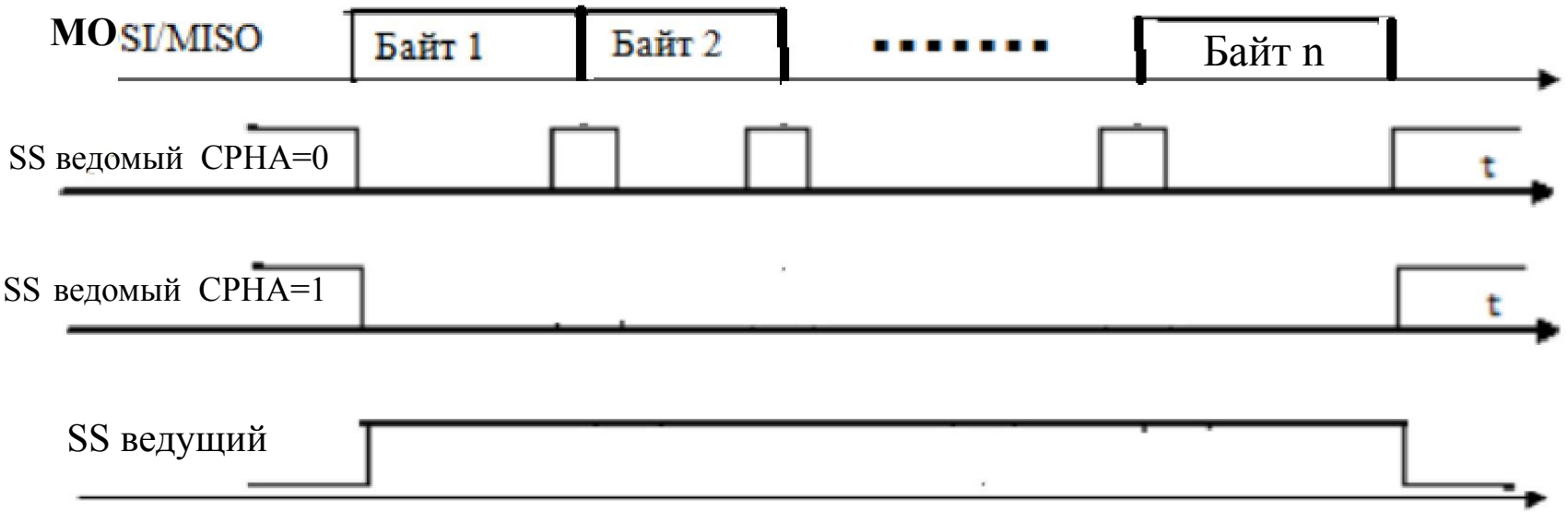
MISO – от ведомого к ведущему

Временная диаграмма работы модуля SPI для режима 01 и 11

CPHA=1
CPOL=0



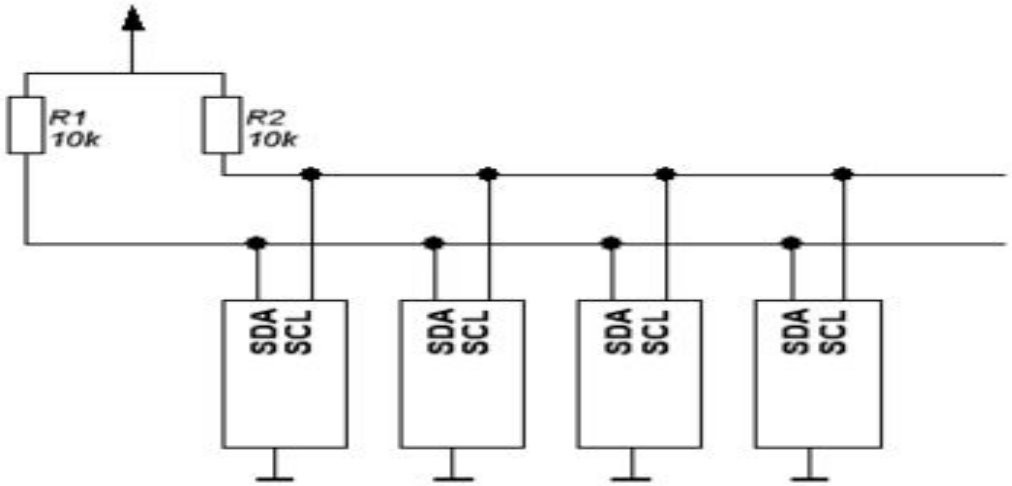
Передача нескольких байтов в любом направлении модулемSPI

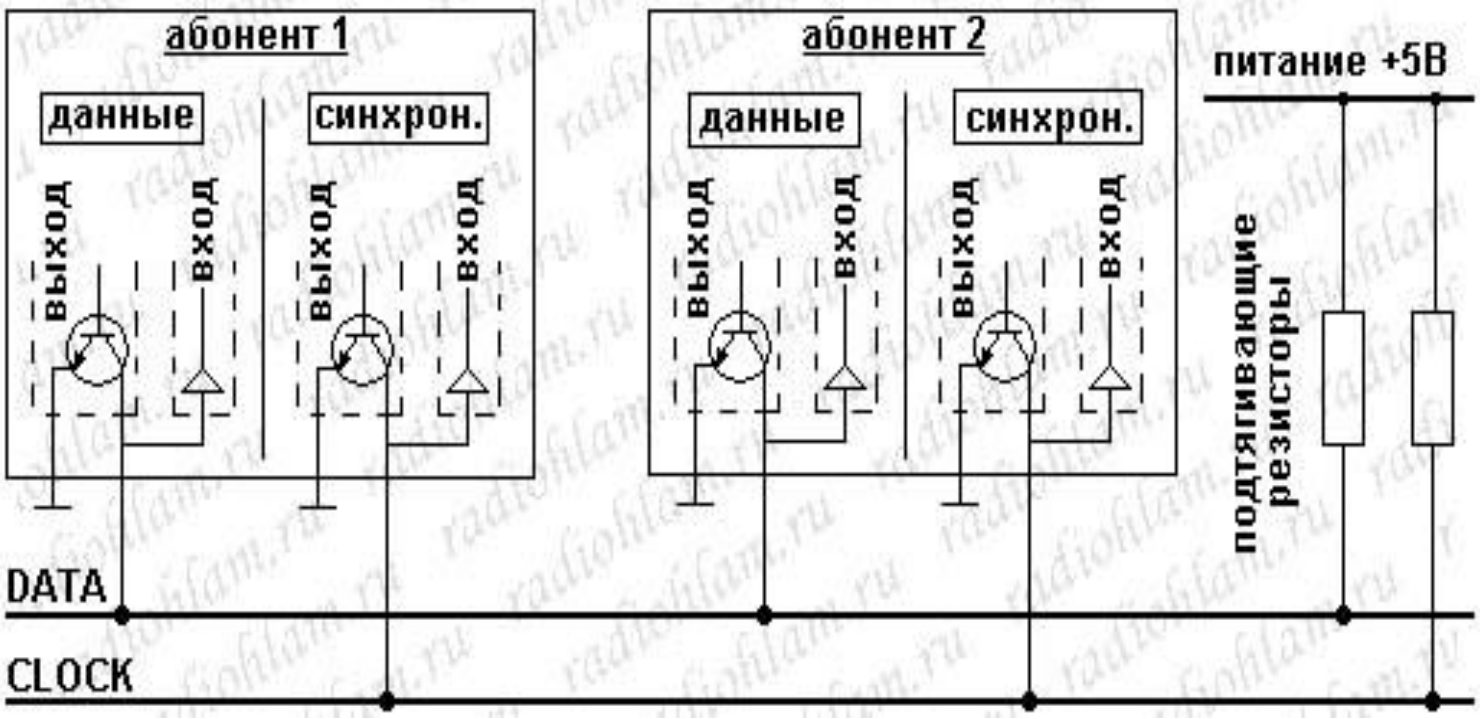


SS ведомого в режимах 00 и 01 после передачи каждого байта переходит в неактивное состояние. В режимах 10 и 11 сохраняет активное состояние

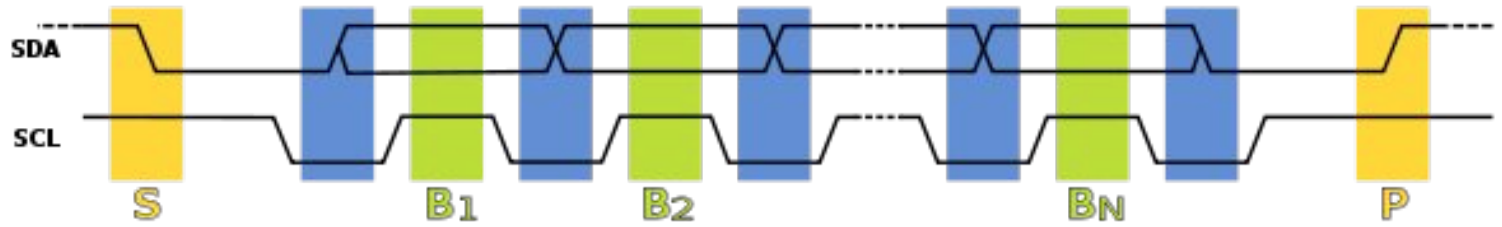
Интерфейс I2C

Последовательный протокол обмена данными IIC (также называемый **I2C – Inter-Integrated Circuits,** межмикросхемное соединение) использует для передачи данных две двунаправленные линии связи, которые называются шина последовательных данных **SDA (Serial Data)** и шина тактирования **SCL (Serial Clock)**. Также имеются две линии для питания. Шины SDA и SCL подтягиваются к шине питания через резисторы.

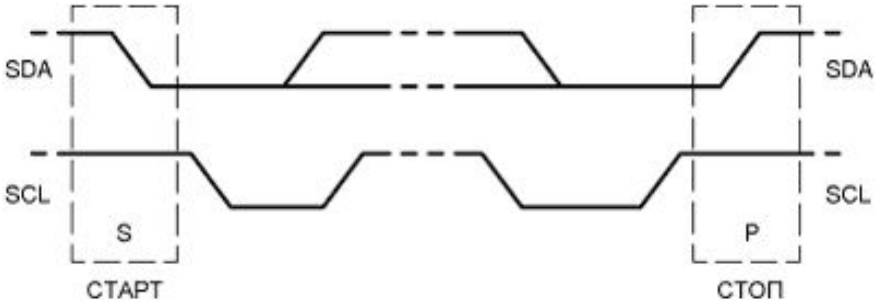



абонент 1
данные
синхрон.
выход
вход
выход
вход
абонент 2
данные
синхрон.
выход
вход
выход
вход
питание +5В
подтягивающие резисторы
DATA
CLOCK

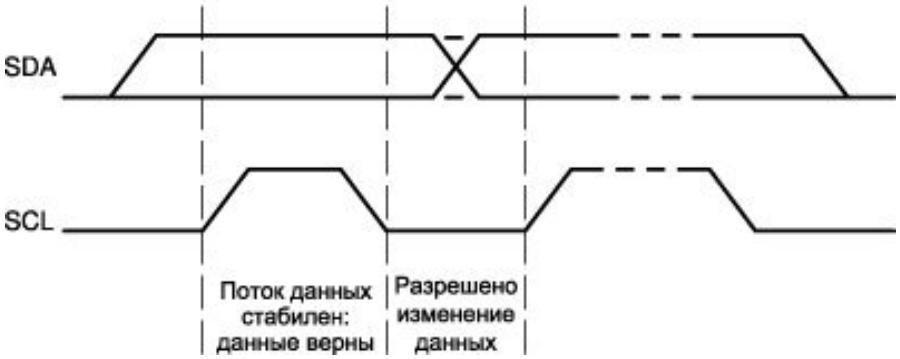
Процесс обмена по шине от момента формирования состояния СТАРТ до состояния СТОП



Состояние СТАРТ и СТОП.



Передача данных



•Начало передачи определяется **Start** последовательностью — провал **SDA** при высоком уровне **SCL**
•При передаче информации от Master к Slave, **ведущий генерирует такты** на **SCL** и выдает биты на **SDA**, которые **ведомый считывает когда SCL становится 1**.
•При передачи информации от Slave к Master, **ведущий генерирует такты на SCL**. **А ведомый, когда SCL уходит в 0, выставляет на SDA бит, который мастер считывает когда поднимет SCL обратно**.

Заканчивается все **STOP** последовательностью. когда при высоком уровне на **SCL** линия **SDA** переходит с низкого на высокий уровень.

То есть, изменение на шине данных в момент приема данных может быть только при низком уровне на SCL. Когда SCL вверху то идет чтение. Если же у нас SDA меняется при высоком SCL, то это уже служебные команды START или STOP.

Логический уровень

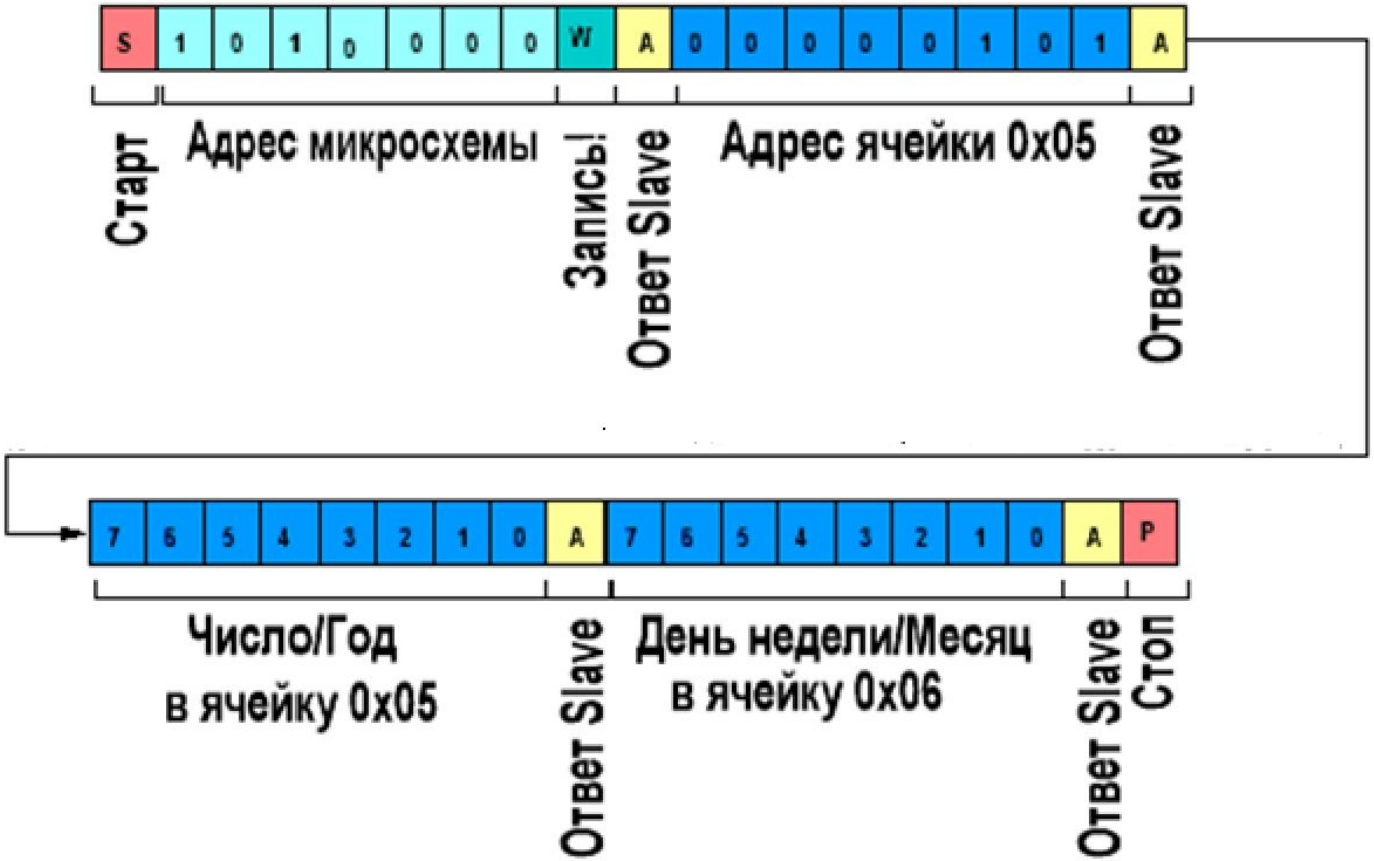
Первый пакет передается от ведущего к ведомому это физический адрес устройства и бит направления.

Запись W=0

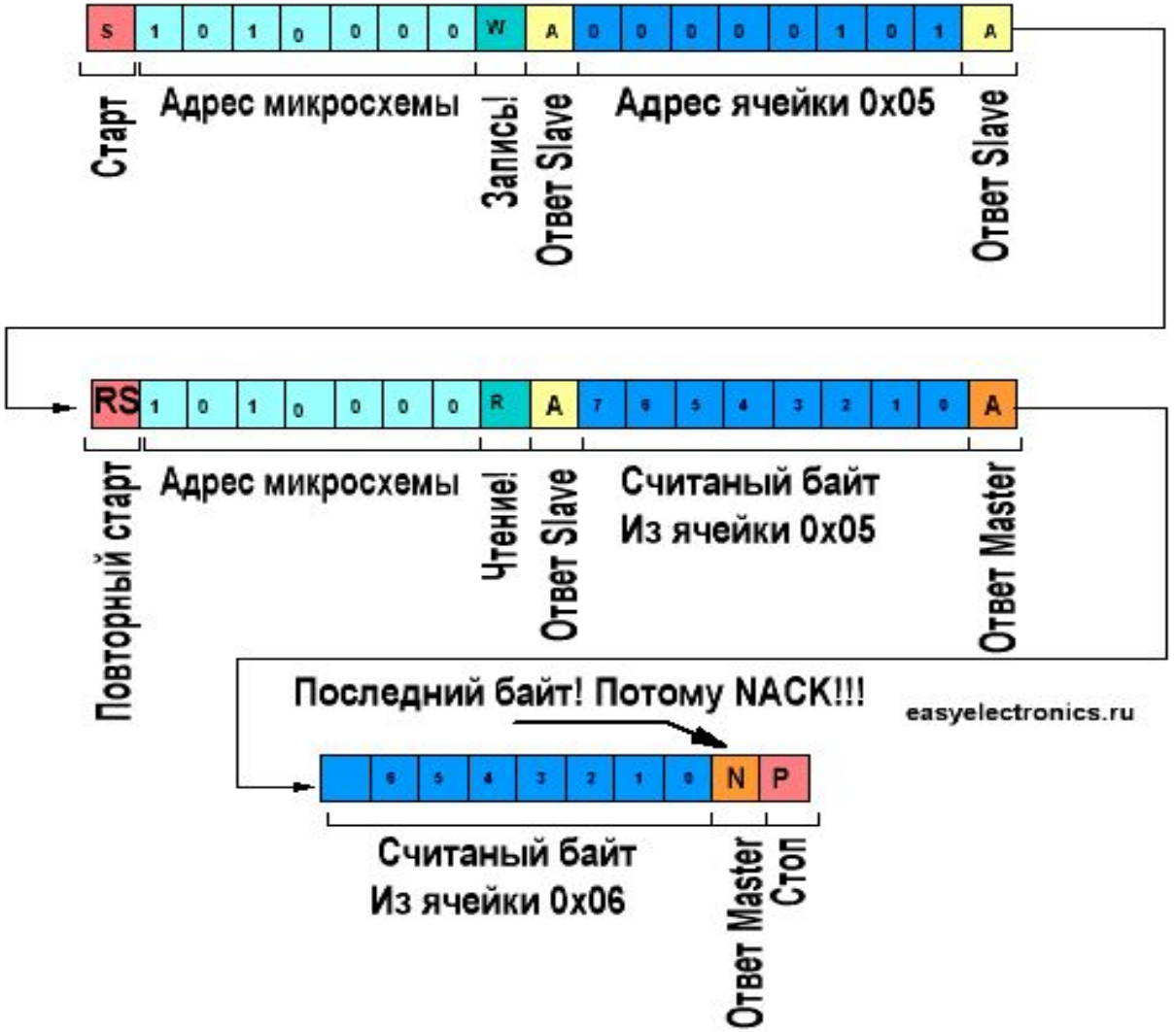
Чтение R=1


Последний байт! Потому NACK!!!
S
7 6 5 4 3 2 1
R
A
7 6 5 4 3 2 1 0
A
7 6 5 4 3 2 1 0
N
P
Старт
Slave адрес
Чтение!
Ответ Slave
Считаный байт 1
Ответ Master
Считаный байт 2
Ответ Master
Стоп
easyelectronics.ru

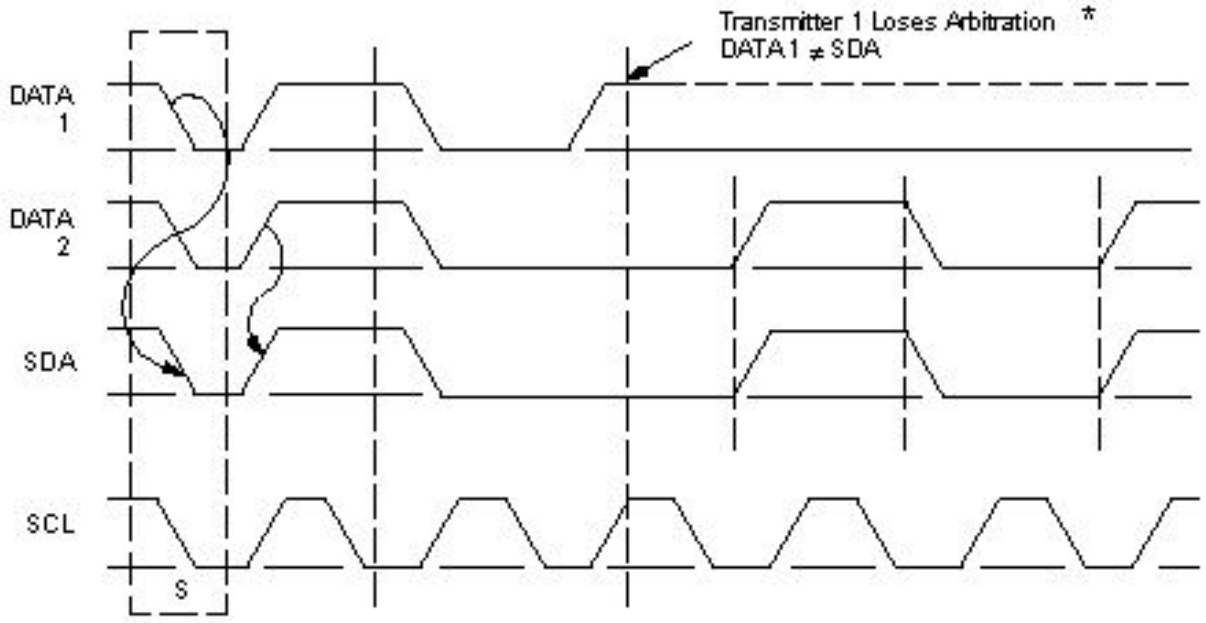
Обращение к микросхеме часов реального времени PCF8583 (запись)



Обращение к микросхеме часов реального времени PCF8583 (чтение)

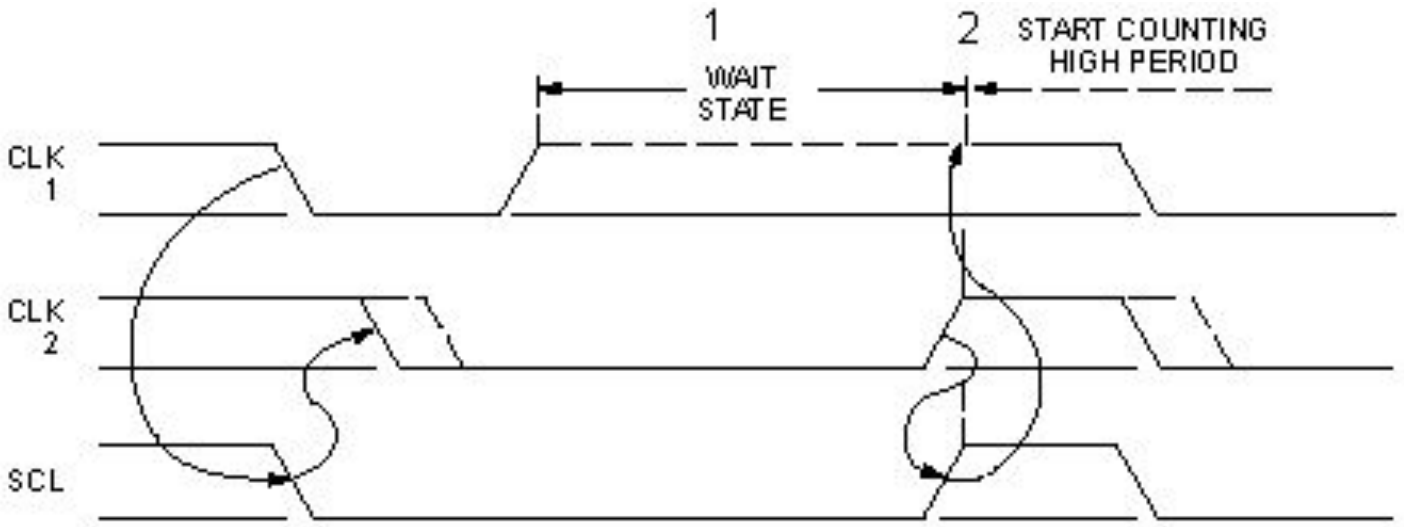


Арбитраж между двумя ведущими



Передатчик 1 проигрывает арбитраж - его линия данных не совпадает с SDA

Синхронизация во время арбитража



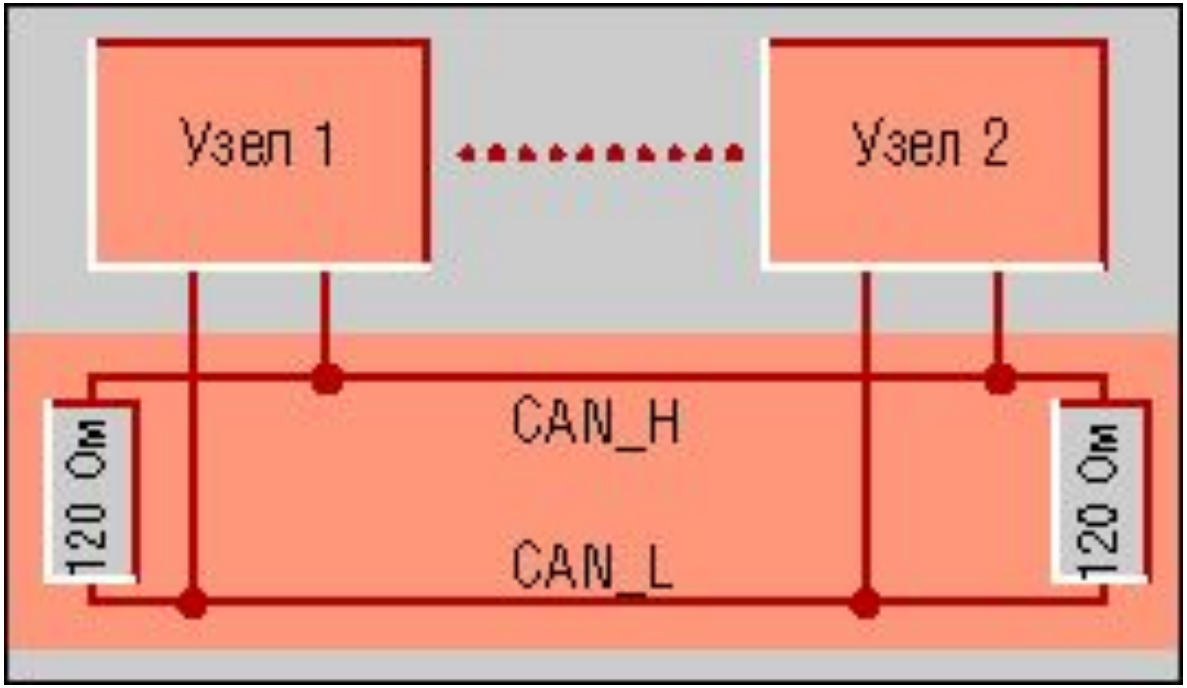
1. Состояние ожидания
2. Начало отсчета ВЫСОКОГО периода синхроимпульса

Период LOW определяется устройством с самым длинным периодом LOW, а период HIGH определяется устройством с самым коротким периодом HIGH.

CAN-шина и протокол (Control Area Network)

Шина CAN была предложена Робертом Бошем (Robert Bosch) в 80-х годах для автомобильной промышленности, затем стандартизована ISO (ISO 11898) и SAE (Society of Automotive Engineers).

Большинство европейских автомобильных гигантов (например, Audi, BMW, Renault, Saab, Volvo, Volkswagen) используют CAN в системах управления двигателем, безопасности и обеспечения комфорта.



Структурная схема CAN-шины

Основные характеристики CAN сети

1) Среда передачи данных в CAN не определена это может быть витая пара, оптоволокно.

2) Скорость передачи задается программно и может быть до 1 Мбит/с. Пользователь выбирает скорость, исходя из расстояний, числа абонентов и емкости линий передачи.

3) Максимальное расстояние 500 м.

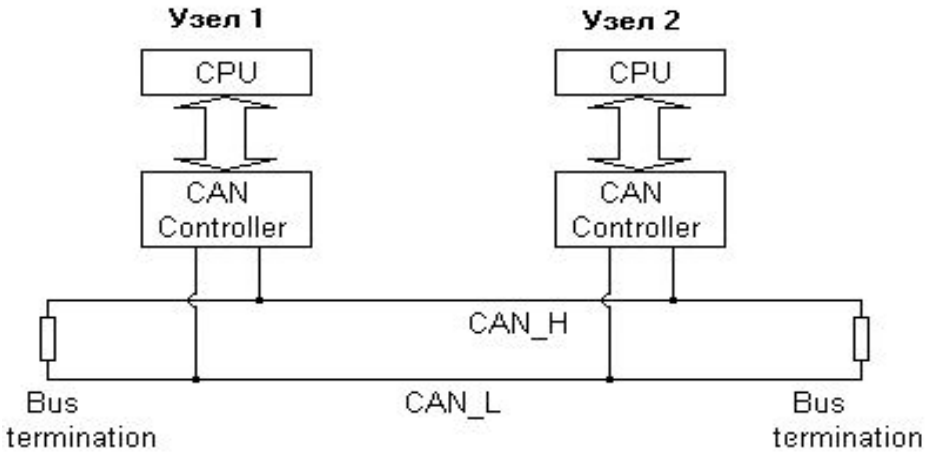
4) Максимальное количество узлов 64

5) Количество байтов данных настраивается от 0 до 8.

6) Если хоть один узел в сети принял сообщение с ошибкой, это сообщение признается ошибочным для всех узлов сети.

7) Отказавшие узлы динамически отключаются от шины.

Зависимость скорости обмена от длины линии передачи



CAN контроллеры соединяются с помощью дифференциальной шины, которая имеет две линии - CAN_H (can-high) и CAN_L (can-low), по которым передаются сигналы. Логический ноль регистрируется, когда на линии CAN_H сигнал выше, чем на линии CAN_L. Логическая единица - в случае когда сигналы CAN_H и CAN_L одинаковы (отличаются менее чем на 0.5 В).

типичные CAN_L и CAN_H значения при напряжении питания +5 В

Сигнальные уровни на CAN-шине

Типы фреймов (сообщений) в CAN-протоколе

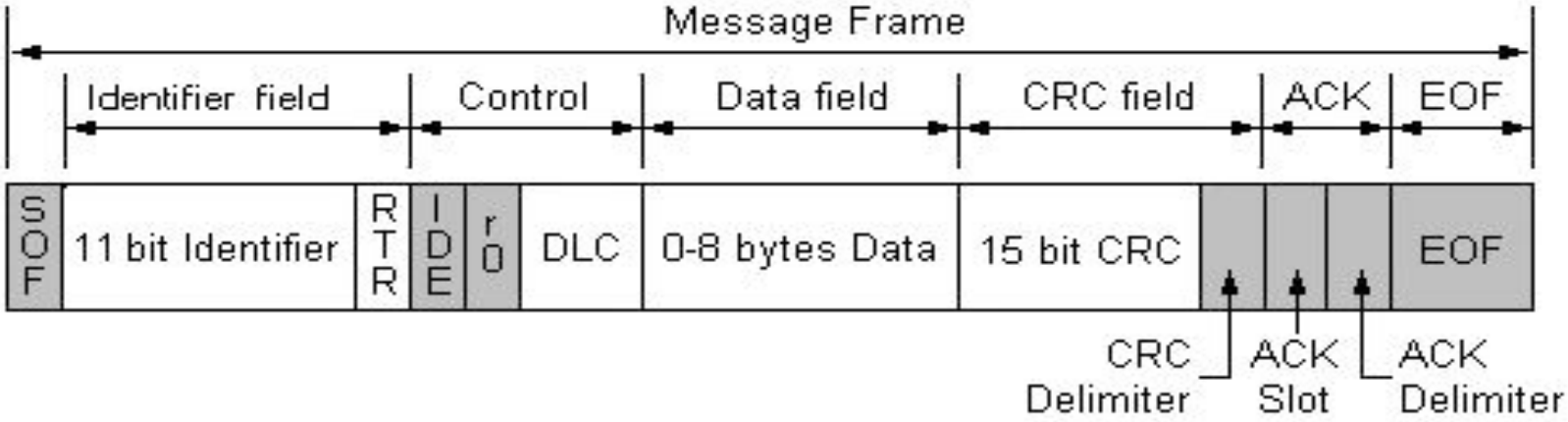
фрейм данных (data frame) перемещает данные с передатчика на приемник (приемники);

удаленный фрейм (remote frame) запрашивает передачу фрейма данных, связанного с определенным идентификатором;

фрейм ошибки (error frame) выражает, какой узел обнаружил ошибку шины/сети;

фрейм перегрузки (overload frame) обеспечивает задержку между передачей фреймов, чтобы управлять потоком данных.

Data frame стандарта CAN 2.0A



Поле SOF (Start of Frame) находится в начале фрейма данных и содержит один доминирующий бит.

Поле арбитража Arbitration Field состоит из

для стандарта CAN-2.0A, 11-битного идентификатора + 1 бит RTR (retransmit)

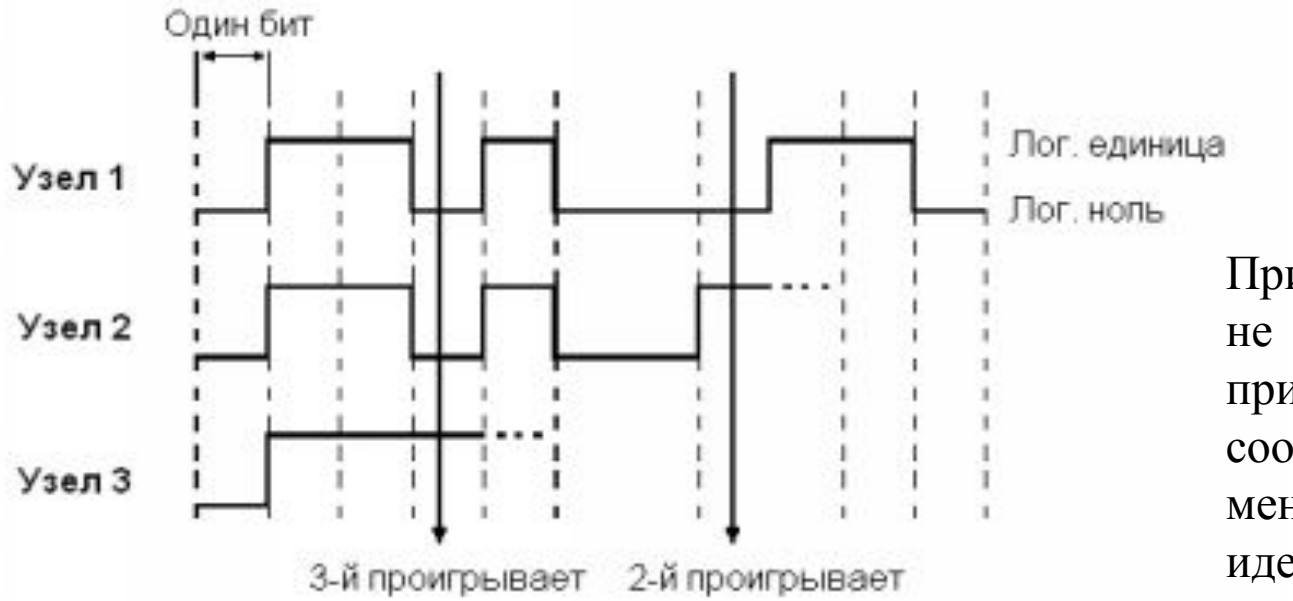
для стандарта CAN-2.0B, 29-битного идентификатора + 1 бит RTR (retransmit)

Для Data кадра бит RTR всегда выставлен в логический ноль (доминантный сигнал).

Управляющее поле (Control Field) содержит 6 битов, из которых 4 бита (DLC0-DLC4) составляют поле Data Length Code, показывающее количество байтов данных, которое будет передаваться в поле данных; два других бита зарезервированы для следующих редакций протокола.
Поле данных (data field) содержит от 0 до 8 байт данных
Поле CRC (CRC field) содержит 15-битную контрольную сумму сообщения.
Поле подтверждения ACK Field содержит участки ACK Slot и ACK Delimiter и выполняет следующую функцию: передающий узел посылает по одному рецессивному биту, а приемник, если он принял сообщение без сбоев, устанавливает на линии доминирующий бит в поле ACK Slot.
Поле конца фрейма EOF состоит из семи рецессивных битов

Побитовый арбитраж на шине CAN

Поле арбитража CAN-кадра используется в CAN для разрешения коллизий доступа к шине методом неразрушающего арбитража. Контроллер, который передавал логический ноль (более приоритетный сигнал) будет продолжать передачу, а другой контроллер прервёт свою передачу до того времени, пока шина вновь не освободится.



Приоритетным является не передающий или приемный узел, а сообщение, имеющее меньшее значение идентификатора

CAN-протокол обеспечивает механизмы обнаружения следующих типов ошибок:

Разрядная ошибка **Bit monitoring** появляется, когда передатчик сравнивает уровень на шине с уровнем, который должен передаваться, и 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** 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 CRC-ошибка.
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.

Общая вероятность необнаруженной ошибки **4.7×10^{-11}**.

Механизм ограничения ошибок

Каждый узел ведет два счетчика ошибок: Transmit Error Counter (счетчик ошибок передачи) и Receive Error Counter (счетчик ошибок приема).

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 1.

Каждый узел CAN сети может находится 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, состоящий из 6 доминантных бит, поэтому все узлы его регистрируют. Узел в состоянии Passive Error при обнаружении ошибки в сети передает в сеть Passive Error Flags , состоящий 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.

Адресация в CAN-протоколе

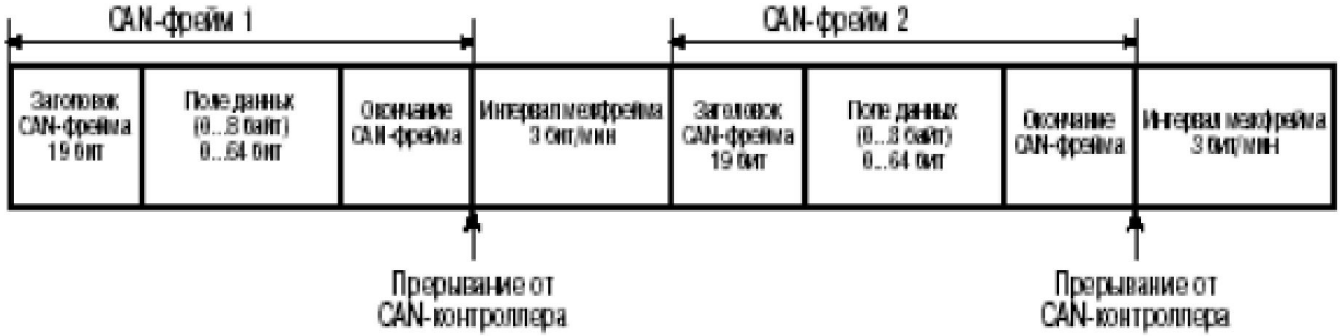
Типы входных фильтров:
фиксированные — фильтры, которые требуют, чтобы биты соответствовали точно один к одному (one-for-one).
Mask-and-Match (маскируемые) — фильтры, которые применяют маску к полю идентификатора, прежде чем он сравнивается с приемным регистром кода

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»)

ПРИМЕР: если биты 10-6 идентификатора установлены в 11110, то эти сообщения принимаются независимо от значений битов 5-0.

Прерывания в CAN-протоколе



Так как фрейм данных в CAN-протоколе короткий (от 0 до 8 байт), скорость поступления прерываний на процессор может быть высокой. В связи с этим следует рассматривать CAN как высокоскоростную сеть. Рисунок демонстрирует два передаваемых подряд CAN-фрейма данных с минимальным интервалом между фреймами, называемым интервалом межфрейма.

Количество байтов данных	0	1	2	3	4	5	6	7	8
Количество битов в CAN-фрейме	47	55	63	71	79	87	95	103	111
Интервал между прерываниями на скорости передачи 125 кбит/с (период 1 бита – 8 мс)	376	440	504	568	632	696	760	824	888
Интервал между прерываниями на скорости передачи 250 кбит/с (период 1 бита – 4 с)	188	220	252	284	316	348	380	412	444
Интервал между прерываниями на скорости передачи 500 кбит/с (период 1 бита – 2 мс)	94	110	126	142	158	174	190	206	222

Таблица показывает самый жесткий режим прерывания для случая, если CANприемник получает все фреймы во время текущей связи (непрерывные фреймы в режиме back-to-back).

Строка «Число битов в CAN-протоколе» в таблице принимается с условием, что заполнение дополнительными битами отсутствует (естественно, что такое заполнение увеличило бы время между прерываниями). Из таблицы видно, что трафик прерываний достаточно интенсивен. На скорости 500 кбит/с прерывания могут происходить каждые 94 мкс при отсутствии информации в фреймах данных.

Большинство микроконтроллеров нижнего уровня не может поддерживать такую высокую скорость обработки прерываний. Следовательно, нужно находить компромисс между возможностями CAN-контроллера и его стоимостью. Следует выбирать CAN-контроллер, который обеспечивает соответствующий уровень предварительной фильтрации.
Контроллер должен иметь достаточное время для обработки прикладной программы и успевать обслуживать запросы от CAN-сети, или необходимо выделять отдельный микроконтроллер для обслуживания CAN-приемника.

Микросхемы, которые поддерживают CAN-протокол, выпускаются различными поставщиками, такими как Philips, Motorola, Siemens, National Instruments и Intel. Существуют следующие два типа микросхем.

Встроенные — микросхемы, которые включают в себя CAN-контроллер и один из видов интегрированного микроконтроллера. Это Intel 80196CA, содержащий в одном кристалле стандартный контроллер 80196 и CAN-контроллер 82527; Philips 82C592 и 82C598, имеющие контроллер 80C51 и CAN-контроллер 82C200; Motorola 68HC05X4, 68HC705X4, 68HC705X32 на основе M6805.

Периферийные — микросхемы, которые содержат только CAN-контроллер. Это Intel 82527 с 14 фиксированными входными фильтрами, одним типа Mask-and-Match и поддержкой стандартного и расширенного фреймов; Philips 82C200 с одним входным фильтром типа Mask-and-Match и поддержкой стандартного фрейма; Siemens SAB 81C90, 81C91 с 16 фиксированными входными фильтрами.

Применение в индустриальных приложениях

В настоящее время CAN-протокол активно используется в индустриальных сетях. Известные фирмы Hoheywell и Allan-Bradley, разработали и поддерживают сетевые протоколы верхнего уровня SDS и DeviceNet, причем последний является открытым и на данный момент более 200 фирм выпускают и разрабатывают свои изделия в этом стандарте. Кроме того, достаточно известными в Европе являются стандарты сети верхнего уровня CanOpen, CAL (Германия) и CanKingdom (Швеция). Все эти сети используют CAN-протокол на физическом и транспортном уровнях.

Микроконтроллеры семейств PIC (Peripheral Interface Controller) компании Microchip

Основных семейств 8-разрядных RISC-микроконтроллеров

PIC12CXXX – семейство микроконтроллеров, выпускаемых в миниатюрном 8-выводном исполнении. Система команд как 12-разрядная (33 команды), так и 14-разрядная (35 команд). Содержат встроенный тактовый генератор, таймер/счетчик, сторожевой таймер, может быть встроенный АЦП. Способны работать при напряжении питания до 2,5 В;

PIC16C5X – базовое семейство с 12-разрядными командами (33 команды), выпускаемое в 18-, 20- и 28-выводных корпусах. Простые недорогие микроконтроллеры с минимальной периферией. Способность работать при малом напряжении питания (до 2В) делает их удобными для применения в переносных конструкциях.

PIC16CXXX – семейство среднего уровня с 14-разрядными командами (35 команд). Наиболее многочисленное семейство, с разнообразными периферийными устройствами, в число которых входят аналоговые компараторы, АЦП, контроллеры последовательных интерфейсов SPI, USART и I2C, таймеры-счетчики, модули захвата/сравнения, широтно-импульсные модуляторы, сторожевые таймеры и т.д.;

PIC17CXXX – семейство высокопроизводительных микроконтроллеров с расширенной системой команд 16-разрядного формата (58 команд), с объемом памяти программ до 16 Кслов. Кроме обширной периферии, 16-уровневого аппаратного стека и векторной системы прерываний, имеют встроенный аппаратный умножитель 8x8, выполняющий операцию умножения за один машинный цикл;

PIC18CXXX – семейство высокопроизводительных микроконтроллеров с расширенной системой команд 16-разрядного формата (75 команд) и встроенным 10-разрядным АЦП, работающие на частоте до 40 МГц. Содержат 31-уровневый аппаратный стек, встроенную память команд до 32 Кслов и способны адресовать до 4 Кбайт памяти данных и до 2 Мбайт внешней памяти программ. Расширенное RISC-ядро данного семейства оптимизировано под использование Си-компилятора.

Общие сведения о микроконтроллерах семейства PIC16CXXX

Микроконтроллеры семейства PIC16CXXX, выполненные по технологии HCMOS представляют собой 8-разрядные микроконтроллеры на основе RISC-процессора, выполненные по гарвардской архитектуре.

Имеют встроенное ПЗУ команд объемом от 0,5 до 4 Кслов (разрядность слова команд равна 12 – 14 бит).

Память данных PIC-контроллеров организована в виде регистрового файла объемом 32 – 128 байт, в котором от 7 до 16 регистров отведено для управления системой и обмена данными с внешними устройствами.

Широкий диапазон напряжений питания (2 – 6 В).

Ток потребления на частоте 32768 Гц составляет менее 15 мкА,

на частоте 4 МГц – 1 – 2 мА

на частоте 20 МГц 5 – 7 мА

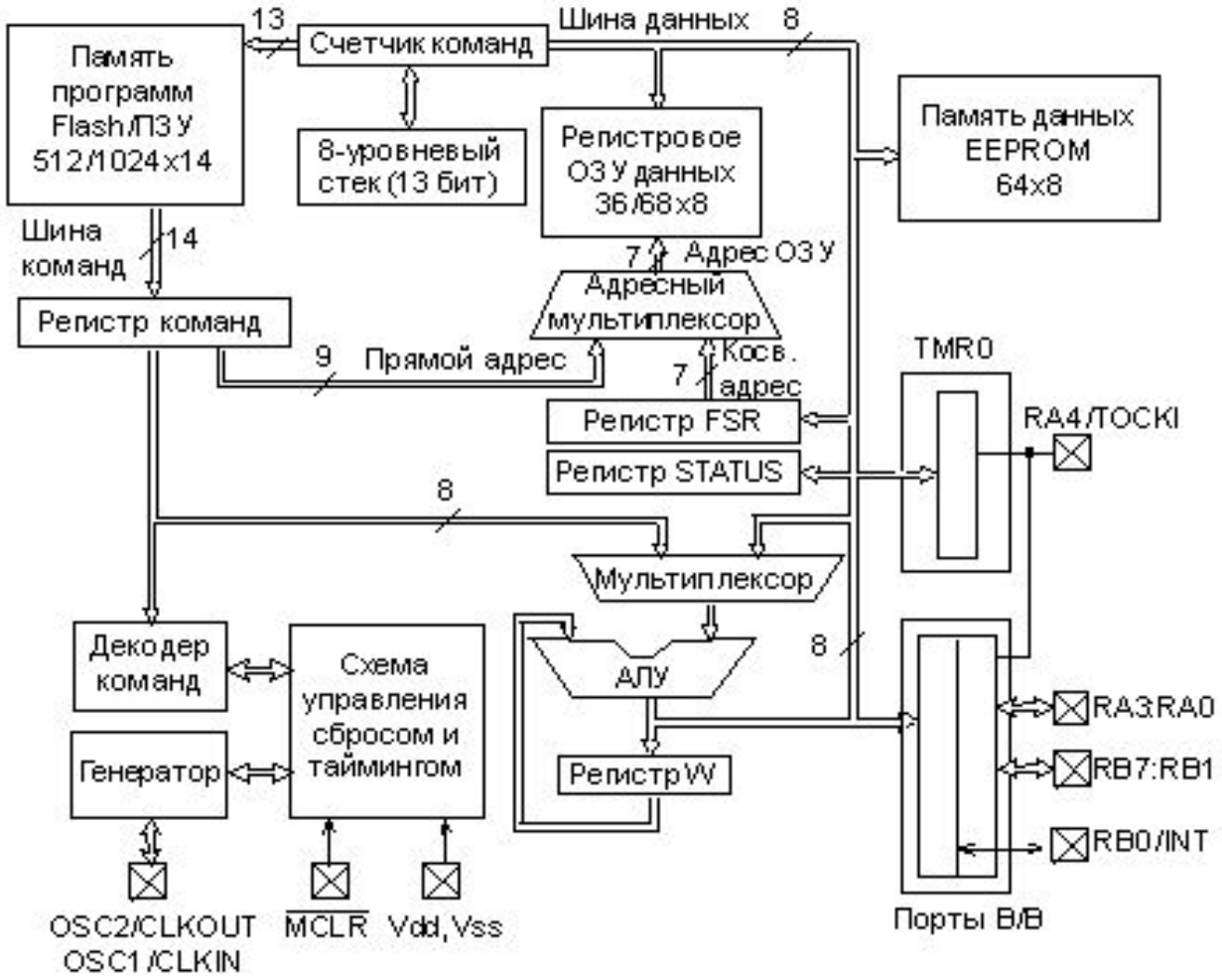
в режиме микропотребления (режим SLEEP) – 1 – 2 мкА

.Выпускаются модификации для работы в трех температурных диапазонах: от 0 до +70°С, от -40 до +85°С и от -40 до +125°С.

Каждый из контроллеров содержит универсальные (от 1 до 3) и сторожевой таймеры, а также надежную встроенную систему сброса при включении питания.

Частота внутреннего тактового генератора задается либо кварцевым резонатором, либо RC-цепочкой в диапазоне 0 – 25 МГц.

PIC-контроллеры имеют от 12 до 33 линий цифрового ввода-вывода, причем каждая из них может быть независимо настроена на ввод или вывод.
Помимо памяти программ в PIC предусмотрено несколько индивидуально прожигаемых перемычек, с помощью которых можно на этапе программирования кристалла выбрать тип тактового генератора, отключить сторожевой таймер или систему сброса, включить защиту памяти программ от копирования, а также записать серийный номер кристалла .
Число команд небольшое — от 33 до 35.
Все команды имеют одинаковую длину и, кроме команд ветвления, выполняются за четыре периода тактовой частоты .
.Поддерживаются непосредственный, косвенный и относительный методы адресации можно эффективно управлять отдельными битами в пределах всего регистрового файла.
Стек реализован аппаратно. Его максимальная глубина составляет два или восемь уровней в зависимости от типа контроллера.
Почти во всех микросхемах PIC есть система прерываний, источниками которых могут быть таймер и внешние сигналы.
Система команд практически симметрична


Память программ Flash/ПЗУ 512/1024x14
13
Счетчик команд
Шина данных
8
8-уровневый стек (13 бит)
Регистровое ОЗУ данных 36/68x8
Память данных EEPROM 64x8
Шина команд
14
7
Адрес ОЗУ
Адресный мультиплексор
Регистр команд
9
Прямой адрес
7
Косв. адрес
TMR0
Регистр FSR
RA4/TOCKI
Регистр STATUS
8
Мультиплексор
Декодер команд
Схема управления сбросом и таймингом
АЛУ
8
Генератор
Регистр W
RA3:RA0
RB7:RB1
RB0/INT
OSC2/CLKOUT
OSC1/CLKIN
MCLR
Vdd, Vss
Порты B/B

Характеристики микроконтроллеров подгруппы PIC16F8X

используются только 35 простых команд;
все команды выполняются за один цикл (400 нс при частоте 10 МГц), кроме команд перехода, которые требуют 2 цикла;
рабочая частота 0 Гц ... 10 МГц;
раздельные шины данных (8 бит) и команд (14 бит);
512 x 14 или 1024 x 14 память программ, выполненная на ПЗУ или электрически перепрограммируемой Flash- памяти;
15 восьмиразрядных регистров специальных функций (SFR);
восьмиуровневый аппаратный стек;
прямая, косвенная и относительная адресация данных и команд;
36 или 68 восьмиразрядных регистров общего назначения (GPR) или ОЗУ;

четыре источника прерывания:

внешний вход RB0/INT;

переполнение таймера TMR0;

изменение сигналов на линиях порта B;

завершение записи данных в память EEPROM;

64 x 8 электрически перепрограммируемая EEPROM память данных с возможностью выполнения 1000000 циклов стирания/записи;

сохранение данных в EEPROM в течение как минимум 40 лет.

развитые возможности ввода/вывода:

13 линий ввода-вывода с индивидуальной установкой направления обмена;

высокий втекающий/вытекающий ток, достаточный для управления светодиодами:

максимальный втекающий ток – 25 мА;

максимальный вытекающий ток – 20 мА;

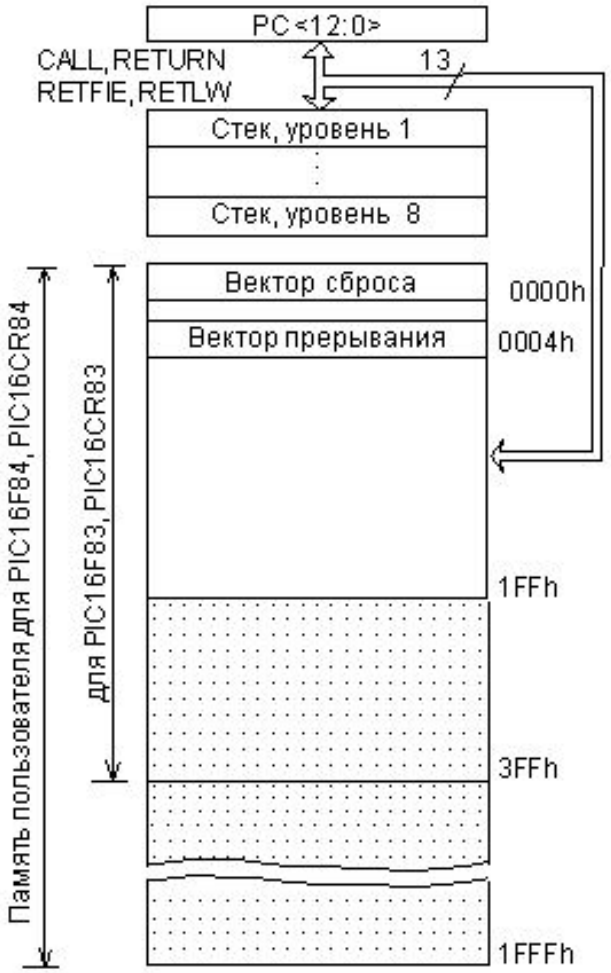
8-битный таймер/счетчик TMR0 с 8-битным программируемым предварительным делителем.

Основные характеристики МК подгруппы PIC16F8X.

Параметр	PIC16F83	PIC16CR83	PIC16F84	PIC16CR84
Максимальная частота, МГц	10	10	10	10
Flash-память программ, слов	512	-	1K	-
ПЗУ программ, слов	-	512	-	1K
Память данных, байт	36	36	68	68
Память данных в РПЗУ (EEPROM), байт	64	64	64	64
Таймеры	TMR0	TMR0	TMR0	TMR0
Число источников прерываний	4	4	4	4
Число линий ввода/вывода	13	13	13	13
Диапазон напряжений питания, В	2,0 – 6,0	2,0 – 6,0	2,0 – 6,0	2,0 – 6,0
Число выводов и тип корпуса	18 DIP, SOIC	18 DIP, SOIC	18 DIP, SOIC	18 DIP, SOIC

Микроконтроллеры семейств PIC (Peripheral Interface Controller) компании Microchip (продолжение)

Память программ и стека



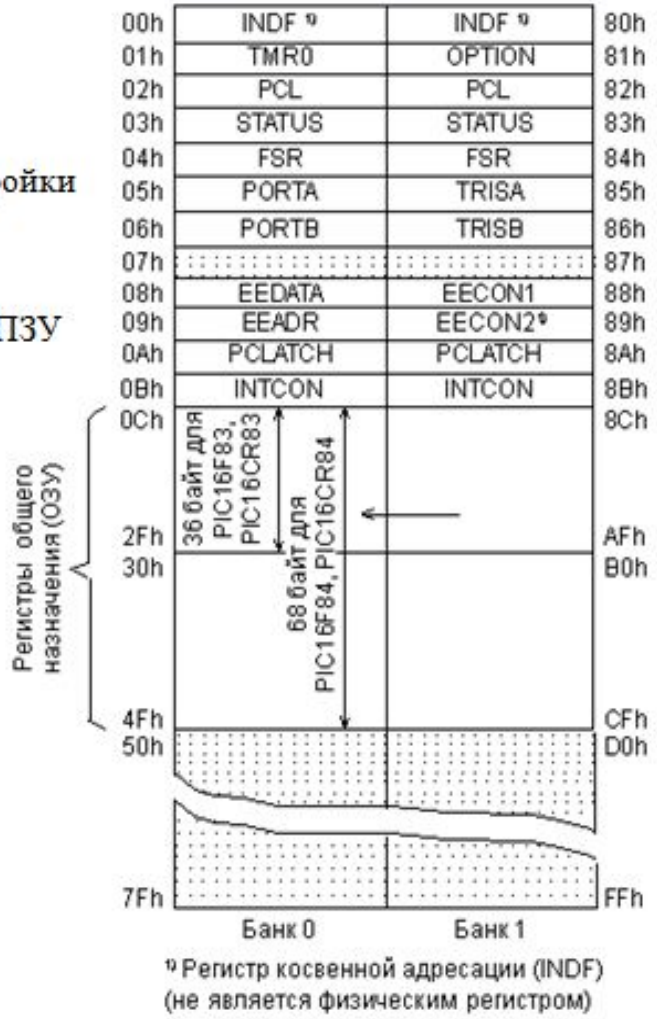
Вектор сброса находится по адресу 0000h, вектор прерывания – по адресу 0004h

Стек работает как циклический буфер. После того как стек был загружен 8 раз, девятая загрузка перепишет значение первой. Признаков положения стека в контроллере не предусмотрено, поэтому пользователь должен самостоятельно следить за уровнем вложения подпрограмм.

Память данных

Регистры настройки портов А иВ

Данные ПЗУ
Адрес ячейки ПЗУ



Регистр признака косвенной адресации
Регистр конфигураций
Младший байт счетчика команд
Регистр признаков (флагов)
Регистр косвенной адресации
Регистры направления портов А и В

Регистры упр.чт/зп в ПЗУ

Старший байт счетчика команд
Регистр условий прерывания

Назначение бит регистра STATUS (адрес 03h, 83h).							
R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	/TO	/PD	Z	DC	C
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0

Бит 7: IRP: бит выбора страницы банка данных (используется при косвенной адресации) В рассматриваемой модели поддерживается в 0.

0 = банк 0,1 (00h – FFh)

1 = банк 2,3 (100h – 1FFh)

Биты 6-5: RP1:RP0: биты выбора страницы банка данных (используются при прямой адресации) В МК подгруппы PIC16F8X используется только бит RP0

00 = банк 0 (00h – 7Fh)

01 = банк 1 (80h – FFh)

10 = банк 2 (100h – 17Fh)

11 = банк 3 (180h – 1FFh)

Бит 4: /TO: бит срабатывания сторожевого таймера

1 = после включения питания, а также командами CLRWDT и SLEEP

0 = по завершении выдержки сторожевого таймера

Бит 3: /PD: бит снижения потребляемой мощности
1 = после включения питания, а также командой CLRWDT
0 = по команде SLEEP
Бит 2: Z: бит нулевого результата
1 = результат арифметической или логической операции нулевой
0 = результат арифметической или логической операции ненулевой
Бит 1: DC: бит десятичного переноса/заема (для команд ADDWF и ADDLW)
1 = имеет место перенос из 4-го разряда
0 = нет переноса из 4-го разряда
Бит 0: C: бит переноса/заема (для команд ADDWF и ADDLW)
1 = имеет место перенос из самого старшего разряда
0 = нет переноса из самого старшего разряда

Назначение бит регистра OPTION (адрес 81h).

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
/RBPU	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0

Бит 7: /RBPU: бит установки резисторов "pull-up" на выводах PORTB
0 = резисторы "pull-up" подключены
1 = резисторы "pull-up" отключены
Бит 6: INTEDG: бит выбора перехода сигнала прерывания
0 = прерывание по спаду сигнала на выводе RB0/INT
1 = прерывание по фронту сигнала на выводе RB0/INT
Бит 5: T0CS: бит выбора источника сигнала таймера TMR0
0 = внутренний тактовый сигнал (CLKOUT)
1 = переход на выводе RA4/T0CKI
Бит 4: T0SE: бит выбора перехода источника сигнала для TMR0
0 = приращение по фронту сигнала на выводе RA4/T0CKI
1 = приращение по спаду сигнала на выводе RA4/T0CKI

У каждой ножки порта B имеется небольшая активная нагрузка (около 100мкА) на линию питания (pull-up). Она автоматически отключается, если эта ножка запрограммирована как вывод. Более того, управляющий бит /RBPU регистра OPTION<7> может отключить (при RBPU=1) все нагрузки. Сброс при включении питания также отключает все нагрузки.

Бит 3: PSA: бит назначения пределителя
0 = предделитель подключен к TMR0
1 = предделитель подключен к сторожевому таймеру WDT
Биты 2-0: PS2:PS0: биты выбора коэффициента деления пределителя

Значения бит	Скорость TMR0	Скорость WDT
000	1:2	1:1
001	1:4	1:2
010	1:8	1:4
011	1:16	1:8
100	1:32	1:16
101	1:64	1:32
110	1:128	1:64
111	1:256	1:128

В том случае, когда предделитель обслуживает сторожевой таймер WDT, таймеру TMR0 назначается коэффициент предварительного деления 1:1.

Назначение бит регистра INTCON (адреса 0Bh, 8Bh).

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0

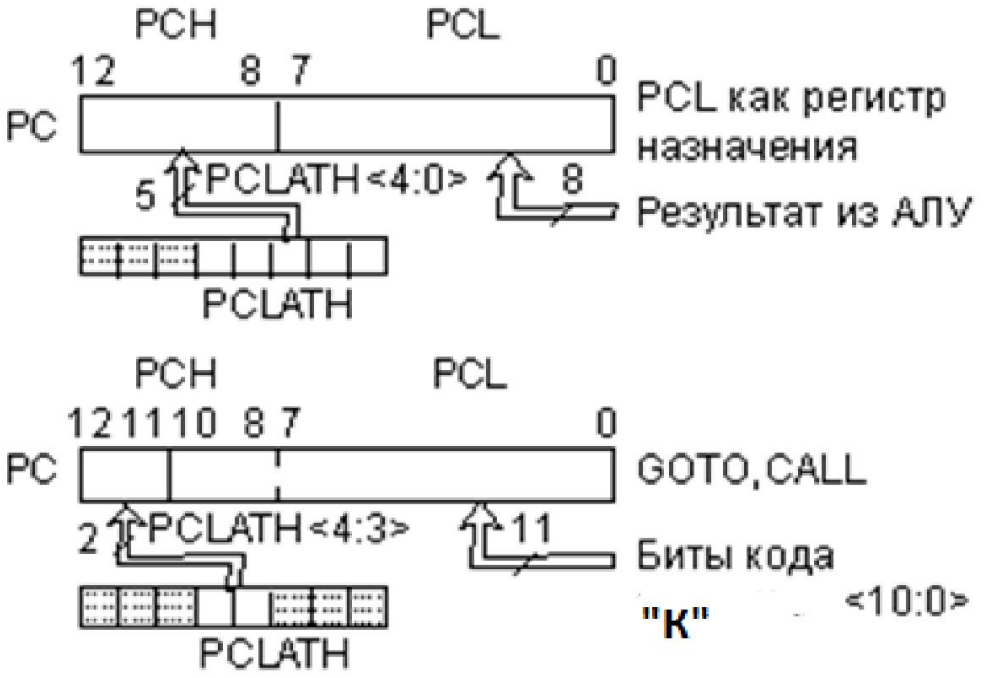
Бит 7: GIE: бит разрешения всех прерываний
0 = запрещены все прерывания
1 = разрешены все незамаскированные прерывания
Бит 6: EEIE: бит разрешения прерывания записи в EEPROM
0 = запрещены прерывания записи в EEPROM
1 = разрешены прерывания записи в EEPROM
Бит 5: T0IE: бит разрешения прерывания по переполнению TMR0
0 = запрещены прерывания от TMR0
1 = разрешены прерывания от TMR0
Бит 4: INTE: бит разрешения прерываний по входу RB0/INT
0 = запрещены прерывания по входу RB0/INT
1 = разрешены прерывания по входу RB0/INT

Бит 3: RBIE: бит разрешения прерываний по изменению PORTB
0 = запрещены прерывания по изменению PORTB
1 = разрешены прерывания по изменению PORTB
Бит 2: T0IF: бит запроса прерывания по переполнению TMR0
0 = прерывание по переполнению TMR0 отсутствует
1 = прерывание по переполнению TMR0 имеет место
Бит 1: INTF: бит запроса прерывания по входу RB0/INT
0 = прерывание по входу RB0/INT отсутствует
1 = прерывание по входу RB0/INT имеет место
Бит 0: RBIF: бит запроса прерывания по изменению PORTB
0 = ни на одном из входов RB7:RB4 состояние не изменилось
1 = хотя бы на одном из входов RB7:RB4 изменилось состояние

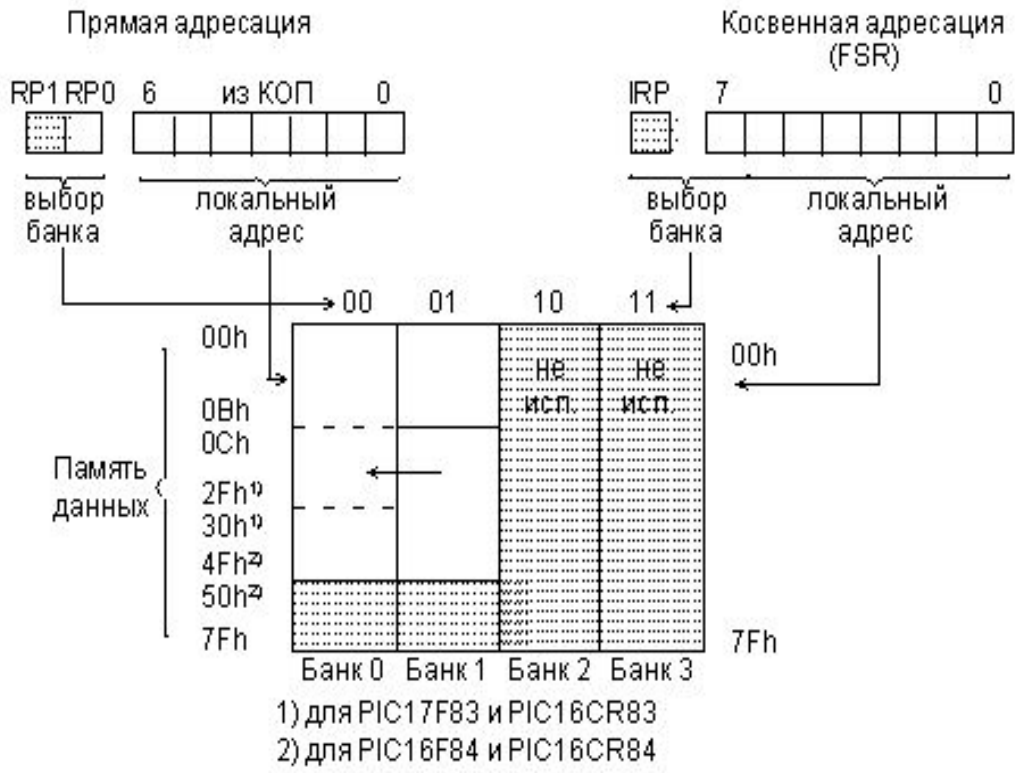
Микроконтроллеры семейств PIC (Peripheral Interface Controller) компании Microchip (продолжение)

Счетчик команд

Счетчик команд PCL и PCLATH имеет разрядность 13 бит. Младший байт счетчика (PCL) доступен для чтения и записи и находится в регистре 02h. Старший байт счетчика команд не может быть напрямую записан или считан и берется из регистра PCLATH (PC latch high), адрес которого 0Ah. Содержимое PCLATH передается в старший байт счетчика команд, когда он загружается новым значением. В зависимости от того, загружается ли в счетчик команд новое значение во время выполнения команд CALL, GOTO, или в младший байт счетчика команд (PCL) производится запись, – старшие биты счетчика команд загружаются из PCLATH разными способами



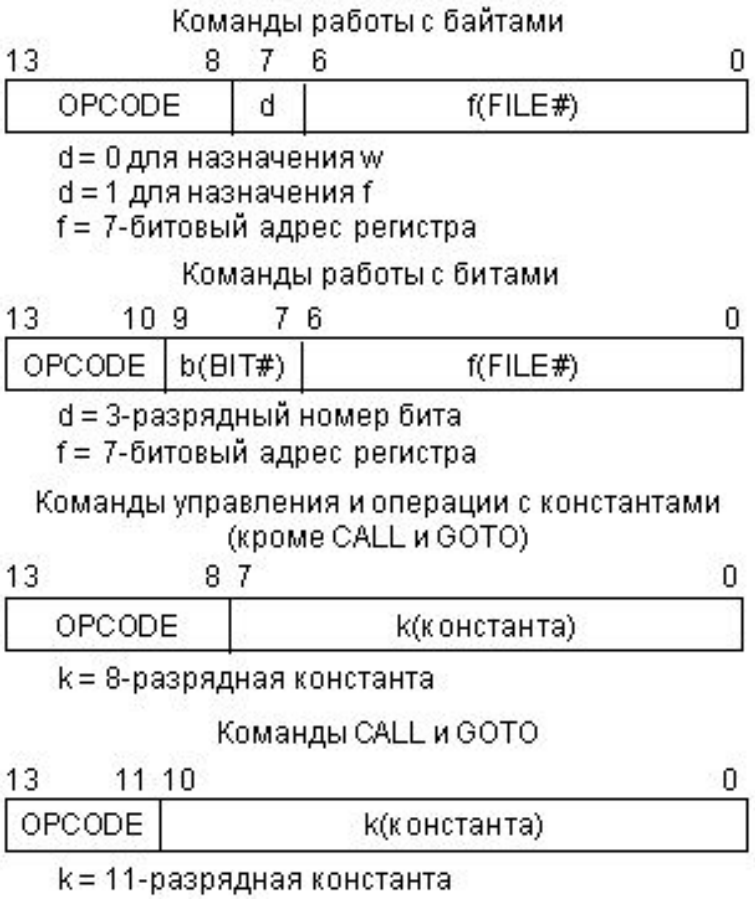
Методы адресации данных



При прямой 9-битовой адресации младшие 7 бит берутся как прямой адрес из команды (поле f), а два бита указателя страниц (RP1, RP0) из регистра статуса. Признаком косвенной адресации является обращение к регистру INDF. Любая команда, которая использует INDF (адрес 00h) в качестве регистра фактически обращается к указателю, который хранится в FSR (адрес 04h). Необходимый 9-битный адрес формируется объединением содержимого 8-битного FSR регистра и бита IRP из регистра статуса

Форматы команд

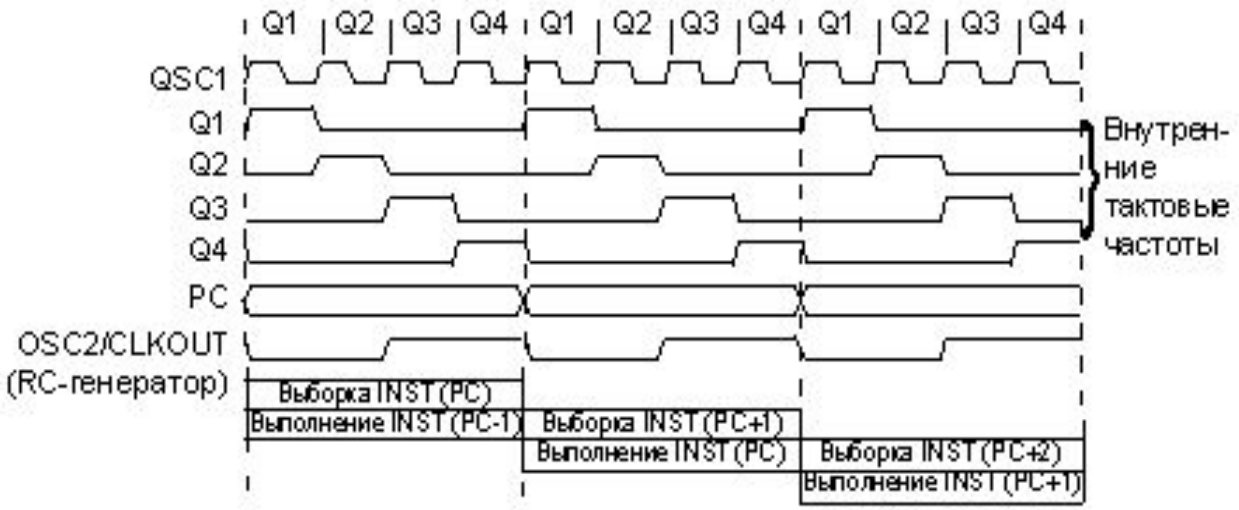
Каждая команда МК подгруппы PIC16F8X представляет собой 14-битовое слово, разделенное на код операции (OPCODE), и поле для одного и более операндов, которые могут участвовать или не участвовать в этой команде. Система команд PIC16F8X является ортогональной и включает в себя команды работы с байтами, команды работы с битами и операции с константами и команды управления



Все команды микроконтроллеров PIC16CXXX однословные 14 - разрядные. 14 - разрядная шина доступа к памяти программ позволят выполнить выборку 14 - разрядной команды за один машинный цикл микроконтроллера. При использовании однословных команд число слов в памяти программ равняется максимальному числу команд программы микроконтроллера. Это означает, что все ячейки памяти имеют силу команды.

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. Задержка в один машинный цикл необходима для выборки новой команды, которая должна быть выполнена следующей.

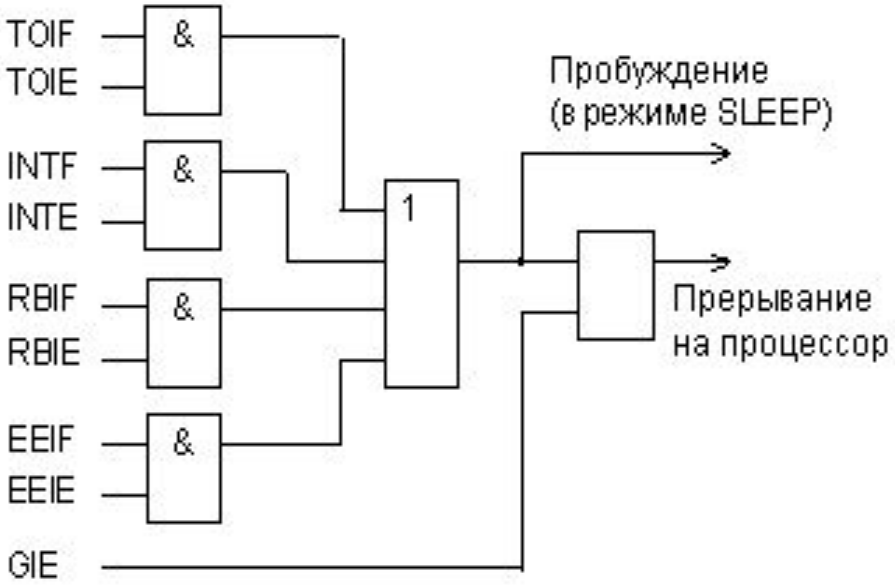
Схема тактирования и выполнения команды.



Входная тактовая частота, поступающая с вывода OSC1/CLKIN, делится внутри на четыре, и из нее формируются четыре циклические не перекрывающиеся тактовые последовательности Q1, Q2, Q3 и Q4. В цикле выборки счетчик команд увеличивается в такте Q1, команда считывается из памяти программ и защелкивается в регистре команд в такте Q4. Команда декодируется и выполняется в течение последующего цикла в тактах Q1...Q4. В такте Q1 – дешифрация КОП, Q2 - считывается память данных (чтение операнда), Q3 – выполнение, запись - в такте Q4.

Организация прерываний

МК подгруппы PIC16F8X имеют четыре источника прерываний:
- внешнее прерывание с вывода RB0/INT;
- прерывание от переполнения счетчика/таймера TMR0;
- прерывание от изменения сигналов на линиях порта RB<7:4>;
- прерывание по окончании записи данных в EEPROM.



Все прерывания имеют один и тот же вектор/адрес – 0004h. Однако в управляющем регистре прерываний INTCON соответствующим битом-признаком записывается, от какого именно источника поступил запрос прерывания. Исключение составляет прерывание по завершении записи в EEPROM, признак которого находится в регистре EECON1.

Порядок действий при возникновении запросов на прерывания

1.Запрос прерывания фиксируется в регистре **INTCON .**

2.Заканчивается выполнение текущей команды.

3.Проверяется наличие глобального и локальных разрешений прерывания, которые должны быть предварительно установлены в регистре **INTCON.** Если разрешения нет, то переход в режим прерывания не происходит и выполняется очередная команда основной программы. Если разрешение есть, то автоматически выполняются следующие действия.

4.Содержимое СК передается в стек по адресу, который находится в указателе стека: (PC) $\longrightarrow$ ((SP)).

5.Адрес вектора прерываний 0x004 заносится в СК. Бит глобального разрешения прерываний обнуляется: (GIE=0). Все запросы прерываний до окончания программы обслуживания прерывания игнорируются.

●

Структура подпрограммы обработки прерывания

1. Подпрограмма начинается с сохранения содержимого регистров W и STATUS в предварительно зарезервированных для них ячейках памяти данных. Затем сохраняется содержимое управляющих регистров,используемых, как в основной программе, так и в программе обработки прерывания, в специально выделенных ячейках памяти данных (сохраняется контекст)
2. Производится поиск источника, пославшего запрос прерывания, путем проверки флагов только тех источников, которым были даны разрешения.
3. После нахождения источника вызывается подпрограмма обслуживания данного источника, которая должна начинаться со сброса флага для исключения повторного вызова. Затем выполняется содержательная часть.
4. Восстановление содержимого управляющих регистров. Последним восстанавливается содержимое рабочего регистра W.
5. Подпрограмма обработки прерывания заканчивается командой RETFIE. По этой команде в СК из стека возвращается адрес первой невыполненной команды прерванной программы. Бит глобального разрешения запроса GIE в регистре INTCON аппаратно устанавливается,разрешая последующие прерывания.

EEPROM память данных

EEPROM память данных доступна для записи/чтения в нормальном режиме работы микроконтроллера во всем диапазоне рабочего напряжения питания (VDD). EEPROM память не отображается на адресное пространство памяти данных, а доступна через регистры специального назначения. Для доступа к EEPROM памяти данных используются
4регистра специального назначения:

- EECON1
- EECON2 (не физический регистр)
- EEDATA
- EEADR

Врегистре EEDATA сохраняются 8-разрядные данные записи/чтения, а регистр EEADR содержит адрес ячейки EEPROM памяти данных. С помощью 8 - разрядного регистра EEADR можно адресовать 256 байт EEPROM памяти данных.В выпускаемых микроконтроллерах среднего семейства минимальный объем EEPROM памяти данных 64 байта. EEADR можно рассматривать как регистр косвенной адресации для ячеек EEPROM памяти. Регистр EECON1 содержит биты управления чтением/записью EEPROM памяти данных, а виртуальный регистр EECON2 предназначен для защиты от случайной записи в EEPROM память данных.

Область реализованной EEPROM памяти данных всегда начинается с адреса 00h.

EEPROM память данных позволяет выполнить чтение и запись байта. При записи байта происходит автоматическое стирание ячейки и запись новых данных (стирание перед записью). EEPROM память данных рассчитана на большое количество циклов стирание/запись. Время записи управляется интегрированным таймером и зависит от напряжения питания, температуры и технологического разброса параметров кристалла и обычно составляет 4-8мс

Назначение бит регистра EECON1 (адреса 88h).

U	U	U	R/W-0	R/W-x	R/W-0	R/S-0	R/S-x
-	-	-	EEIF	WRERR	WREN	WR	RD
Бит 7	Бит 6	Бит 5	Бит 4	Бит 3	Бит 2	Бит 1	Бит 0

Биты 7:5 не используются (читаются как ‘0’)

Бит 4: EEIF: бит запроса прерывания по записи в EEPROM
0 = операция записи не завершена или не начиналась
1 = операция записи завершена (должен быть сброшен программно)

Бит 3: WRERR: бит признака ошибки записи в EEPROM
0 = операция записи завершена
1 = операция записи прервана преждевременно (сбросом по /MCLR или сбросом от WDT)

Бит 2: WREN: бит разрешения записи в EEPROM
0 = запрещена запись в EEPROM
1 = разрешены циклы записи

Бит 1: WR: бит управления записью
0 = цикл записи данных в EEPROM завершен
1 = инициирует цикл записи (сбрасывается аппаратно по завершении записи. Бит WR может быть только установлен (но не сброшен) программно)

Бит 0: RD: бит управления чтением
0 = чтение данных EEPROM не инициировано
1 = инициирует чтение данных EEPROM (чтение занимает один цикл. Бит RD сбрасывается аппаратно. Бит RD может быть только установлен (но не сброшен) программно)

Регистр EECON2 не является физическим регистром. Он используется исключительно при организации записи данных в EEPROM. Чтение регистра EECON2 дает нули

Считывание данных из памяти EEPROM

Необходимо записать нужный адрес в регистр EEADR
Установить бит RD EECON1<0> в единицу.
Данные появятся в следующем командном цикле в регистре EEDATA и могут быть прочитаны. Данные в регистре EEDATA фиксируются

Чтение из EEPROM памяти данных

```
        BCF STATUS, RP0  ; Выбрать банк 0
        MOVLW CONFIG_ADDR    Адрес считываемого регистра
;       MOVWF EEADR    ;
        BSF STATUS, RP0   Выбрать банк 1
        BSF  EECON1, RD  ; Чтение
        BCF  STATUS, RP0  Выбрать банк 0
        MOVF EEDATA, W   W = EEDATA


;
;
```

Записи в память EEPROM

необходимо сначала записать адрес в EEADR-регистр и данные в EEDATA-регистр. Затем следует выполнить специальную последовательность команд, производящую непосредственную запись:

```
movlw 55
movwf EECON2
 movlw AAh
movwf EECON2
 bsf EECON1,WR;установить WR бит, начать запись
```

Запись байта не будет произведена, если не выполнена указанная последовательность (запись 55h в EECON2, запись AAh в EECON2, установка бита WR в ‘1’ для каждого байта). Рекомендуется запрещать прерывания при выполнении обязательной последовательности команд. Если во время выполнения указанной последовательности произойдет переход по вектору прерывания, запись байта выполнена не будет.

Чтобы разрешить запись в EEPROM память данных, необходимо установить бит WREN (EECON1<2>) в ‘1’, защищающий от случайной записи. Пользователь должен установить бит WREN в ‘1’ перед началом записи, а после окончания записи сбросить его в ‘0’ (аппаратно бит WREN в ‘0’ не сбрасывается).

После инициализации записи сброс бита WREN в ‘0’ не повлияет на цикл записи, но установка бита WR в ‘1’ будет запрещена, пока WREN = 0.

По окончанию записи бит WR аппаратно сбрасывается в ‘0’, а флаг прерывания EEIF устанавливается в ‘1’. Пользователь может использовать прерывания для проверки окончания записи в EEPROM память данных. Флаг EEIF сбрасывается в ’0’ программно.

```
BCF         STATUS, RP0          ;Банк 0
MOVLW       ADDR  EEPROM
MOVWF       EEADR                ;Загрузка адреса
MOVLW       .D
MOVWF       EEDATA               ;Записываемый байт
BSF         STATUS, RP0          ;Банк 1
BSF          EECON1, WREN        ;Разрешить запись
WRITE:
BCF         INTCON, GIE          ;Запретить прерывания
  MOVLW      55h                 ;
  MOVWF      EECON2              ; Записать 55h
  MOVLW      AAh                 ;
  MOVWF      EECON2              ; Записать AAh
  BSF        EECON1,WR           ; Установить бит WR
                                 ; для начала записи

WRITE       REPEAT:
BTFSC        EECON1, WR          ;Ожидание окончания записи
GOTO        $-1
BTFSC        EECON1, WRERR       ;Был ли сброс во время записи
GOTO         WRITE  REPEAT       ;Был, повторить запись
BSF         INTCON, GIE          ;Разрешить прерывания
BCF          EECON1, WREN        ;Запретить запись в EEPROM
```

Рекомендуется после выполнения операции записи в EEPROM память данных произвести контрольное чтение. Выполнять контрольное чтение особенно рекомендуется, если возможно исчерпание гарантированных циклов стирание/запись. Основные ошибки возникают при записи отдельных битов равных 1, чтение будет давать результат 0.

```
BCF     STATUS, RP0     ; Выбрать банк 0
:                       ; Текст программы
:                       ;
MOVF    EEDATA, W       ; Чтение записываемых данных
BSF     STATUS, RP0     ; Выбрать банк 1
BSF     EECON1, RD      ; Инициализация чтения из EEPROM
                        ; записанных данных
BCF     STATUS, RP0     ; Выбрать банк 0
                        ;
                        ; Проверить, равно значение в регистре W
                        ; и прочитанные данные из EEPROM (EEDATA)?
                        ;
SUBWF   EEDATA, W       ;
BTFSS   STATUS, Z       ; Результат 0?
GOTO    WRITE_ERR       ; НЕТ, данные записаны неправильно
:                       ; ДА, данные записаны правильно
:                       ; Продолжение программы
```

Специальные функции

В PIC16F8X реализованы следующие специальные функции:

сброс;
сторожевой таймер (WDT);
режим пониженного энергопотребления (SLEEP);
выбор типа генератора;
защита кода от считывания;
биты идентификации;
последовательное программирование в составе схемы.

В PIC16F8X существуют различия между вариантами сбросов:

- сброс по включению питания;
- сброс по внешнему сигналу /MCLR при нормальной работе;
- сброс по внешнему сигналу /MCLR в режиме SLEEP;
- сброс по окончании задержки таймера WDT при нормальной работе;
- сброс по окончании задержки таймера WDT в режиме SLEEP.

Для реализации сброса по включению питания в МК подгруппы PIC16F8X предусмотрен встроенный детектор включения питания. Таймер установления питания (PWRT) начинает отсчет времени после того, как напряжение питания пересекает уровень около 1,2…1,8 Вольт. По истечении выдержки около 72мс считается, что напряжение достигло номинала и запускается другой таймер – таймер запуска генератора (OST), формирующий выдержку на стабилизацию кварцевого генератора. Программируемый бит конфигурации позволяет разрешать или запрещать выдержку от встроенного таймера установления питания. Выдержка запуска меняется в зависимости от экземпляров кристалла, от питания и температуры. Таймер на стабилизацию генератора отсчитывает 1024 импульса от начавшего работу генератора. Считается, что кварцевый генератор за это время выходит на режим. При использовании RC генераторов выдержка на стабилизацию не производится.

Микроконтроллеры подгруппы PIC16F8X имеют встроенный сторожевой таймер WDT. Для большей надежности он работает от собственного внутреннего RC-генератора и продолжает функционировать, даже если основной генератор остановлен, как это бывает при исполнении команды SLEEP. Таймер вырабатывает сигнал сброса. Выработка таких сбросов может быть запрещена путем записи нуля в специальный бит конфигурации WDTE. Эту операцию производят на этапе прожига микросхем.

Номинальная выдержка WDT составляет 18 мс (без использования делителя). Она зависит от температуры, напряжения питания, особенностей типов микросхем. Если требуются большие задержки, то к WDT может быть подключен встроенный предделитель с коэффициентом деления до 1:128, который программируется битами PS2:PS0 в регистре OPTION. В результате могут быть реализованы выдержки до 2,3 секунд.

Команды "CLRWDT" и "SLEEP" обнуляют WDT и пределитель, если он подключен к WDT. Это запускает выдержку времени сначала и предотвращает на некоторое время выработку сигнала сброса. Если сигнал сброса от WDT все же произошел, то одновременно обнуляется бит /TO в регистре статуса. В приложениях с высоким уровнем помех содержимое регистра OPTION подвержено сбою. Поэтому регистр OPTION должен обновляться через равные промежутки времени.

Некоторые из специальных регистров при сбросе не инициализируются. Они имеют случайное состояние при включении питания и не изменяются при иных видах сброса. Другая часть специальных регистров инициализируется в "состояние сброса" при всех видах сброса, кроме сброса по окончанию задержки таймера WDT в режиме SLEEP. Просто этот сброс рассматривается как временная задержка в нормальной работе. Есть еще несколько исключений. Счетчик команд всегда сбрасывается в ноль (0000h). Биты регистра статуса /TO и /PD устанавливаются или сбрасываются в зависимости от варианта сброса. Эти биты используются программой для определения природы сброса

Режим пониженного энергопотребления SLEEP предназначен для обеспечения очень малого тока потребления в ожидании (менее 1 мкА при выключенном сторожевом таймере). Выход из режима SLEEP возможен по внешнему сигналу сброса или по окончании выдержки сторожевого таймера.

Кристаллы PIC16F8X могут работать с четырьмя типами встроенных генераторов. Пользователь может запрограммировать два конфигурационных бита (FOSC1 и FOSC0) для выбора одного из четырех режимов: RC, LP, XT, HS. Здесь XT – стандартный кварцевый генератор, HS – высокочастотный кварцевый генератор, LP – низкочастотный генератор для экономичных приложений. Микроконтроллеры PIC16F8X могут тактироваться и от внешних источников.
Генератор, построенный на кварцевых или керамических резонаторах, требует периода стабилизации после включения питания. Для этого встроенный таймер запуска генератора держит устройство в состоянии сброса примерно 18 мс после того, как сигнал на /MCLR ножке кристалла достигнет уровня логической единицы.
Возможность выбора типа генератора позволяет эффективно использовать микроконтроллеры семейства в различных приложениях. Применение RC генератора позволяет уменьшить стоимость системы, а низкочастотный LP-генератор сокращает энергопотребление.

Программный код, который записан в кристалл, может быть защищен от считывания при помощи установки бита защиты (CP) в слове конфигурации в ноль. Содержимое программы не может быть прочитано так, чтобы с ним можно было работать. Кроме того, при установленном бите защиты невозможно изменять программу. То же относится и к содержимому памяти данных EEPROM.

Если установлена защита, то бит CP можно стереть только вместе с содержимым кристалла. Сначала будет стерта EEPROM программная память и память данных, и в последнюю очередь – бит защиты кода CP. При считывании защищенного кристалла чтение любого адреса памяти даст результат вида 0000000XXXXXXX(двоичный код), где X – это 0 или 1.

Память данных EEPROM невозможно проверить после установки бита защиты.

Состояние регистров МК после сброса.

Регистр	Адрес	Сброс по включению питания	Другие виды сброса
W	-	xxxx xxxx	uuuu uuuu
INDF	00h	——— ———	——— ———
TMR0	01h	xxxx xxxx	uuuu uuuu
PCL	02h	0000 0000	0000 0000
STATUS	03h	0001 1xxx	000q quuu
FSR	04h	xxxx xxxx	uuuu uuuu
PORT A	05h	—x xxxx	—u uuuu
PORT B	06h	xxxx xxxx	uuuu uuuu
TRIS A	85h	—1 1111	—1 1111
TRIS B	86h	1111 1111	1111 1111
OPTION	81h	1111 1111	1111 1111
EEDATA	08h	xxxx xxxx	uuuu uuuu
EEADR	09h	xxxx xxxx	uuuu uuuu
EECON1	88h	—0 0000	—0 q000
EECON2	89h	——— ———	——— ———
PCLATH	0Ah	—0 0000	—0 0000
INTCON	0Bh	0000 000x	0000 000u

Здесь: x — неизвестное значение; u — неизменяемый бит; "–" — неиспользуемый бит (читается как "0"); q — значение бита зависит от условий сброса.

Для выбора различных режимов работы используются биты конфигурации которые хранятся в памяти программ по **адресу 2007h** и устанавливаются на этапе программирования кристалла. Пользователю следует помнить, что этот адрес находится ниже области кодов и недоступен программе.
Биты конфигурации позволяют настроить некоторые режимы работы микроконтроллера в соответствии с требованиями конкретного приложения. При включении питания состояние этих битов определяет режим работы микроконтроллера. Программа пользователя не может изменять и читать состояние битов конфигурации (эта операция возможна только в режиме программирования микроконтроллера).

Возможность изменения битов конфигурации после их программирования зависит от технологии изготовления памяти программ и типа корпуса микроконтроллера.
В микроконтроллерах с однократно программируемой памятью (OTP), если бит конфигурации был запрограммирован ('0'), то он не может быть изменен.

В микроконтроллерах с Flash памятью программ эти биты могут быть стерты и повторно запрограммированы.

Назначение бит конфигурации МК PIC15F83 и PIC16F84.

R-u	R-u	R-u	R-u	R-u
CP	/PWRTE	WDTE	FOSC1	FOSC0
Бит 13:4	Бит 3	Бит 2	Бит 1	Бит 0

Биты 13:4 CP: бит защиты памяти программ
0 = память программ защищена
1 = защита отсутствует

Бит 3 /PWRTE: бит использования таймера по включению питания
0 = таймер используется (есть задержка)
1 = таймер не используется

Бит 2: WDTE: бит использования сторожевого таймера
0 = WDT не используется
1 = WDT используется

Биты 1:0 FOSC1:FOSC0: бит выбора типа генератора
11 = генератор RC внешний
10 = генератор HS высокочастотный кварцевый резонатор
01 = генератор XT стандартный кварцевый керамический резонатор
00 = генератор LP низкочастотный кварцевый резонатор

Четыре слова памяти, расположенные по адресам **2000h-2003h,** предназначены для хранения идентификационного кода (ID) пользователя, контрольной суммы или другой информации. Как и слово конфигурации, они могут быть прочитаны или записаны только с помощью программатора. Доступа из программы к ним нет.
Расположение и состав битов конфигурации указывается в технической документации на микроконтроллер.

8-разрядные микроконтроллеры AVR фирмы Atmel.

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

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

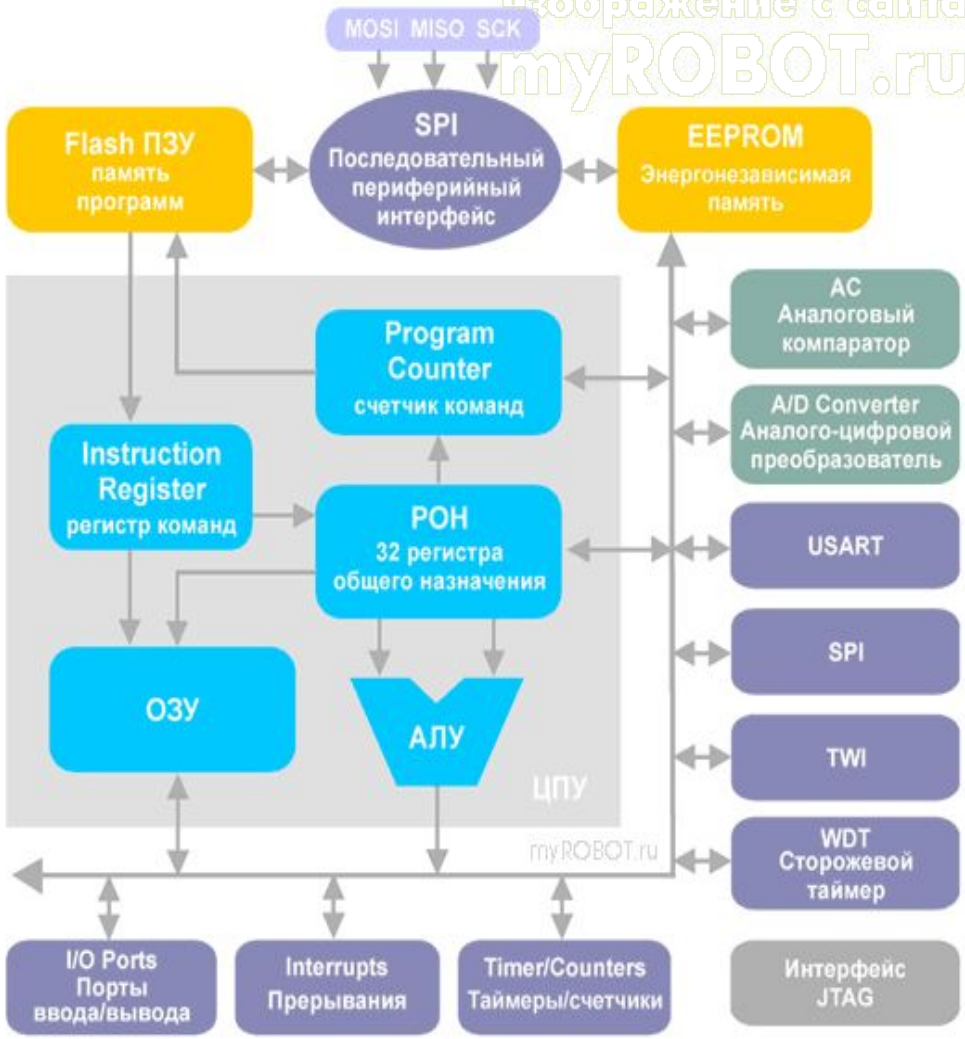
Области применения AVR

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.

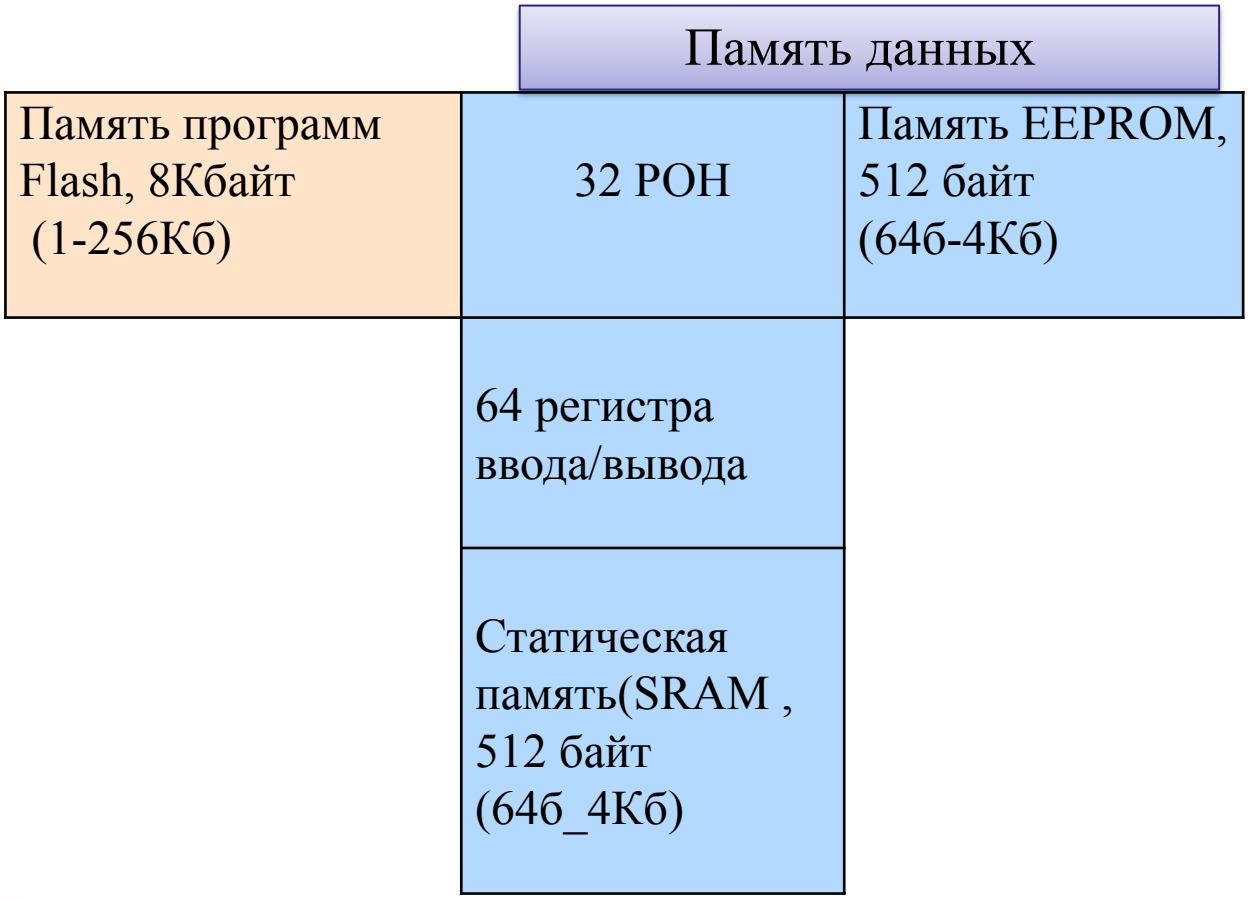
Микроконтроллер AVR содержит: быстрый RISC-процессор, два типа энергонезависимой памяти (Flash-память программ и память данных EEPROM), оперативную память RAM, порты ввода/вывода и различные периферийные интерфейсные схемы



В состав микроконтроллера входят:

■ генератор тактового сигнала (GCK);
■ процессор (CPU);
■ постоянное запоминающее устройство для хранения программы, выполненное по технологии Flash, (FlashROM);
■ оперативное запоминающее устройство статического типа для хранения данных (SRAM);
■ постоянное запоминающее устройство для хранения данных, выполненное по технологии EEPROM, (EEPROM);
■ набор периферийных устройств для ввода и вывода данных и управляющих сигналов и выполнения других функций.

Карта памяти микроконтроллера ATxS8515

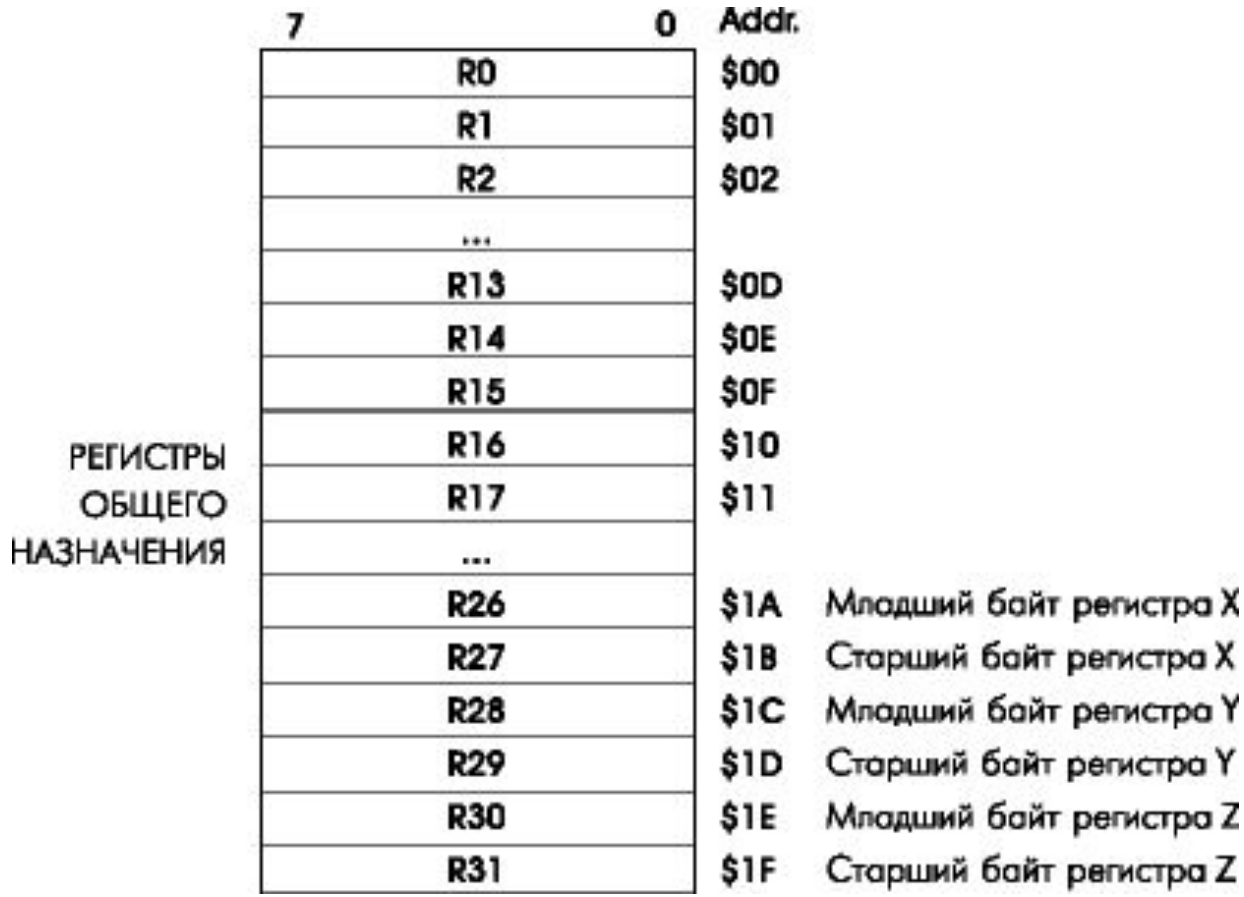


В состав процессора (CPU) входят:

■ счетчик команд (PC);
■ арифметико-логическое устройство (ALU);
■ блок регистров общего назначения

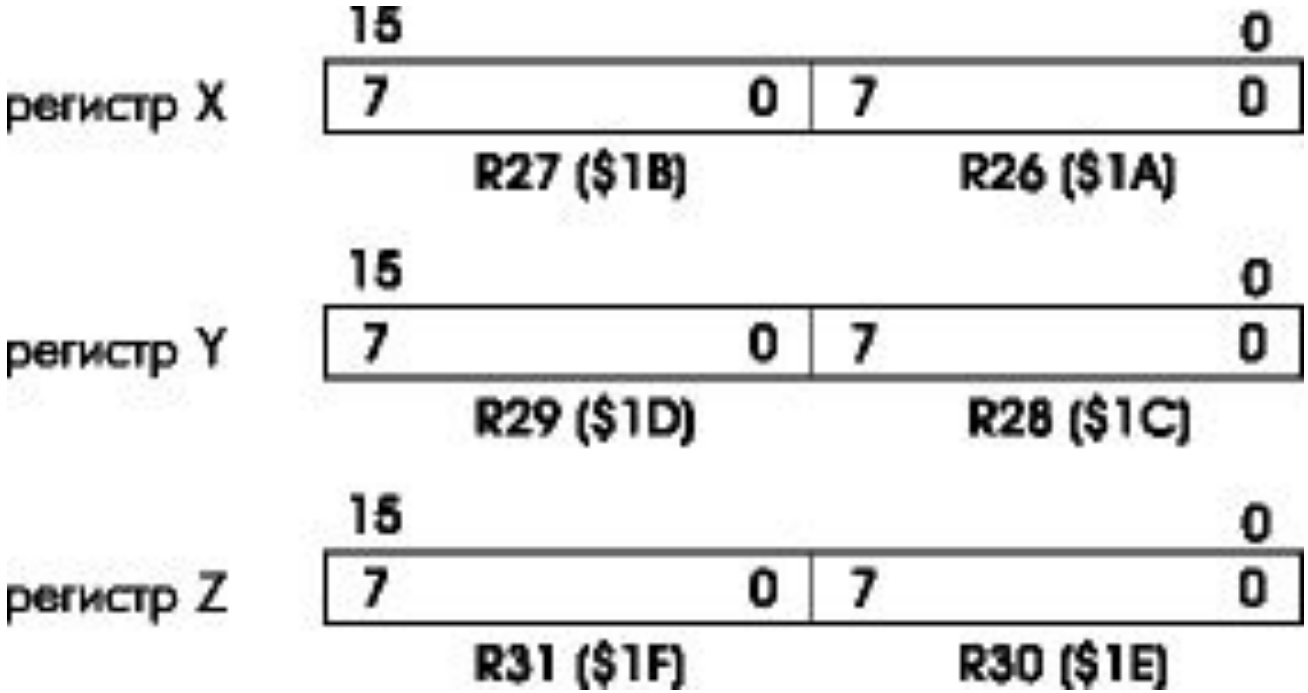
Кроме регистров общего назначения в микроконтроллере имеются регистры специальных функций, которые в семействе AVR называются **регистрами ввода-вывода (I/O Registers, IOR).**

С участием этих регистров осуществляются:
■ управление работой микроконтроллера и отдельных его устройств;
■ определение состояния микроконтроллера и отдельных его устройств;
■ ввод данных в микроконтроллер и отдельные его устройства и вывод данных и выполняются другие функции.

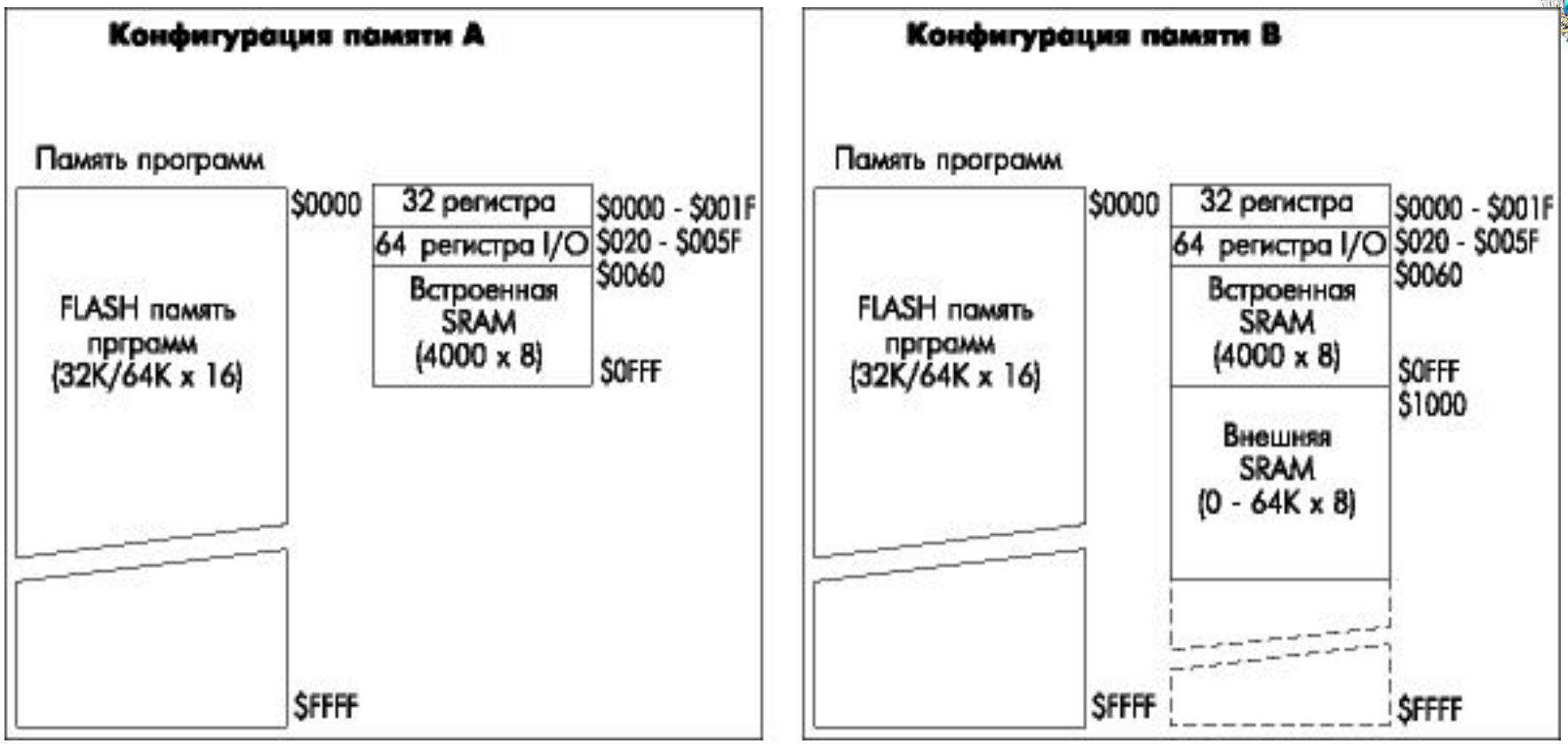


Регистры общего назначения CPU микроконтроллеров AVR

Шесть регистров (с R26 по R31) регистрового файла, кроме обычной для прочих регистров функций, выполняют функцию 16-разрядных регистров указателей адреса при косвенной адресации SRAM. Эти три регистра косвенной адресации определяются как регистры X,Y и Z. В различных режимах адресации эти регистры выполняют функции фиксированного смещения, автоматического инкремента и декремента.



Регистры X, Y и Z



Конфигурация памяти

Первые 96 адресов занимают регистровый файл и пространство памяти I/O, в следующих 4000 адресов размещается встроенная SRAM.

8-разрядные микроконтроллеры AVR фирмы Atmel (продолжение)

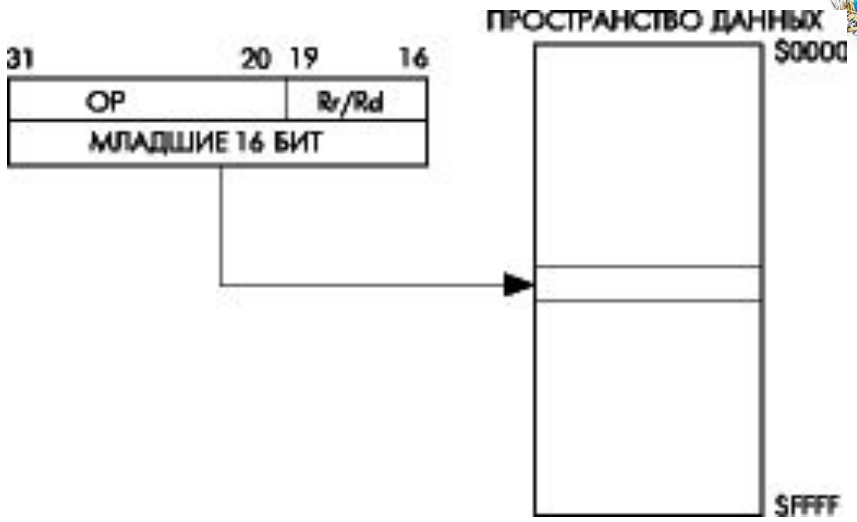
Режимы адресации памяти программ и данных

При обращении к Flash памяти программ и памяти данных (SRAM, регистровому файлу и памяти I/O) AVR Enhanced RISC микроконтроллерами AVR используются мощные и эффективные режимы адресации.

- Непосредственная адресация одного регистра
- Непосредственная регистровая адресация двух регистров
- Непосредственная адресация I/O
- Непосредственная адресация данных
- Косвенная адресация данных со смещением
- Косвенная адресация данных
- Косвенная адресация данных с преддекрементом
- Косвенная адресация данных с постинкрементом
- Адресация константы кода памяти
- Непосредственная адресация памяти программ
- Косвенная адресация памяти программ
- Относительная адресация памяти программ

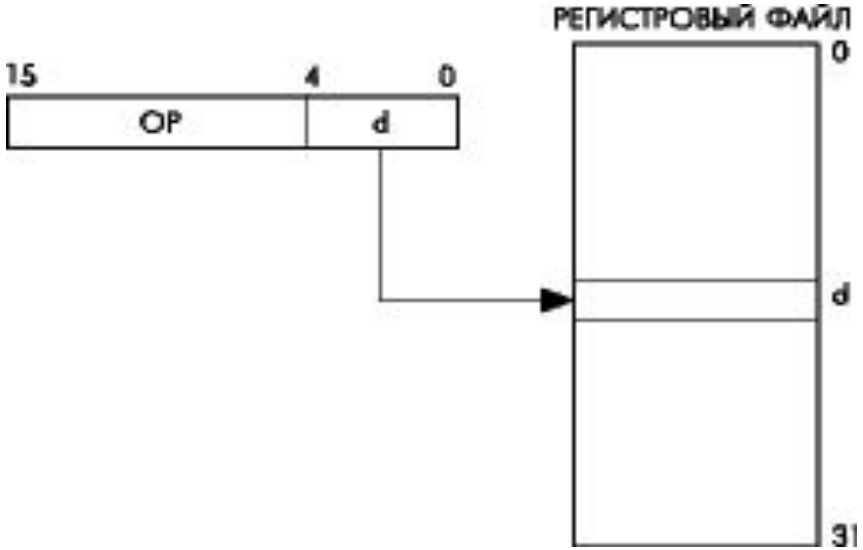
Непосредственная адресация данных

16-разрядный адрес данных содержится в 16 младших разрядах 32-разрядной команды. Rd/Rr определяют регистр источник или регистр назначения.



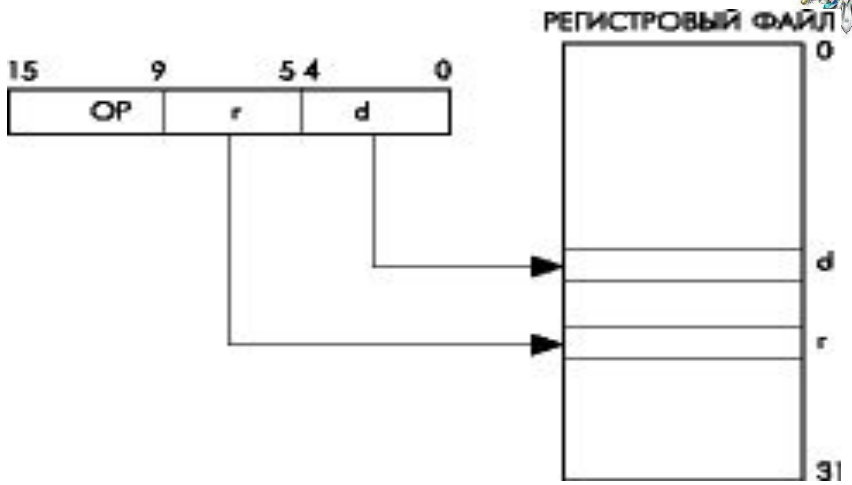
Непосредственная адресация, одиночный регистр Rd

Операнд содержится в регистре d (Rd).



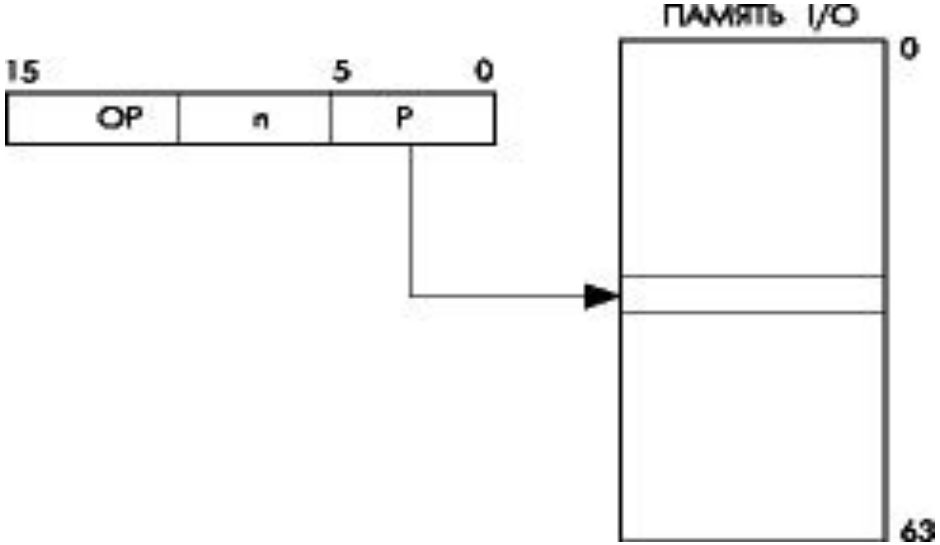
Непосредственная адресация, два регистра Rd и Rr

Операнды содержатся в регистрах r (Rr) и d (Rd).
Результат сохраняется в регистре d (Rd).



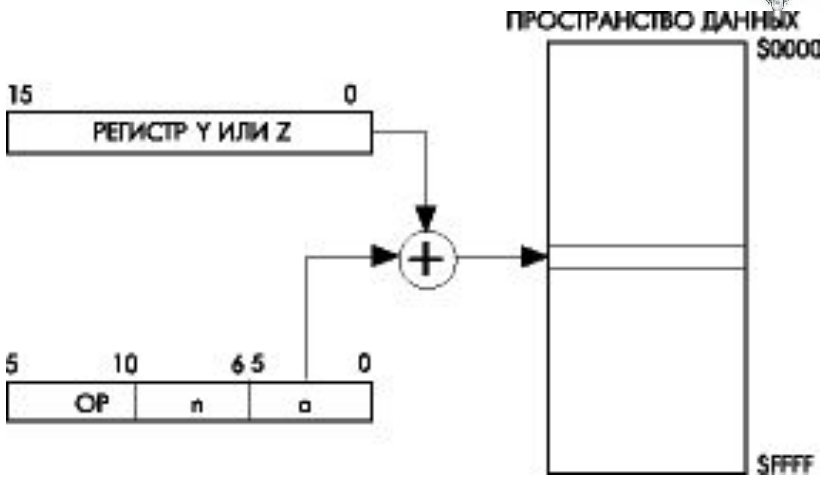
Непосредственная адресация I/O.

Адрес операнда содержится в 6 битах слова команды. Величина n определяет адрес регистра источника или регистра назначения.



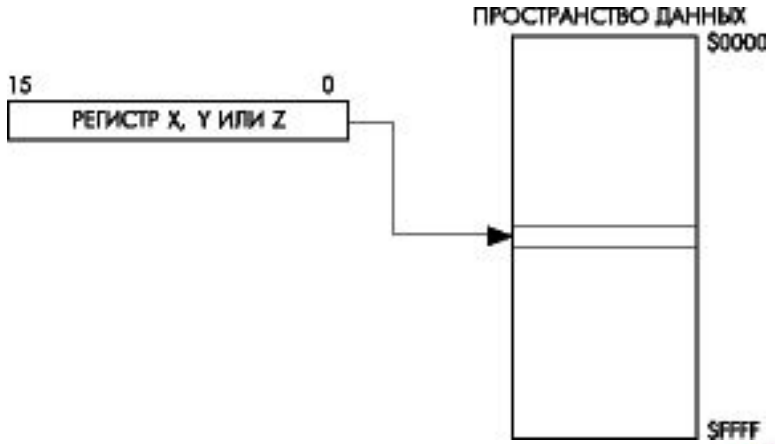
Косвенная адресация данных со смещением

.Косвенная адресация данных со смещением
Адрес операнда вычисляется суммированием содержимого регистра Y или Z с 6 битами адреса, содержащимися в слове команды.



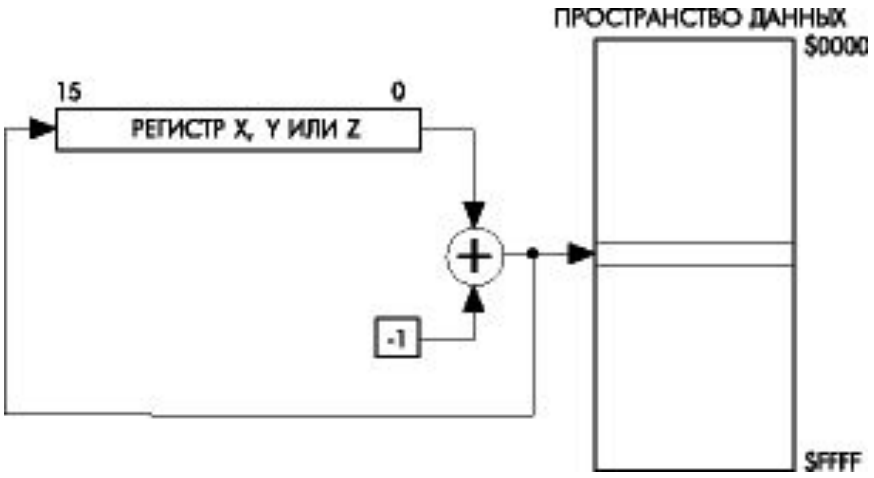
Косвенная адресация данных

Косвенная адресация данных
Адрес операнда содержится в регистре X, Y или Z.



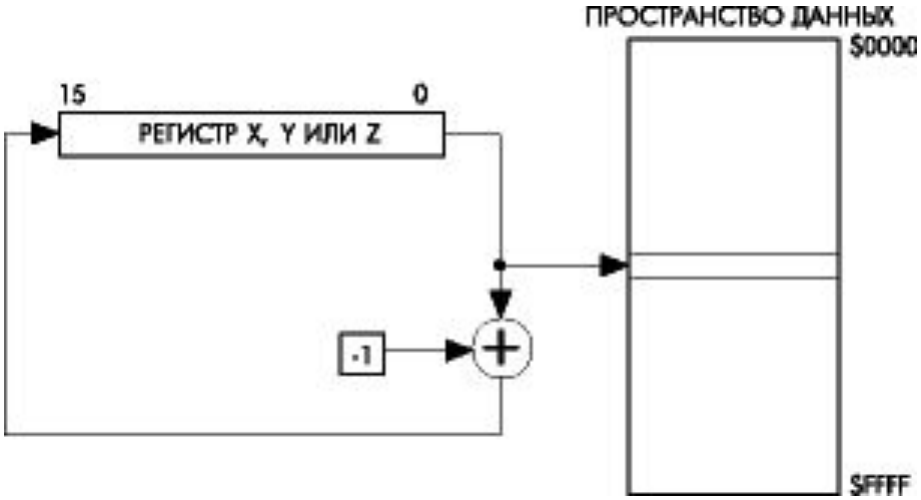
Косвенная адресация данных с преддекрементом

Перед выполнением операции регистр X, Y или Z декрементируется. Декрементированное содержимое регистра X, Y или Z является адресом операнда



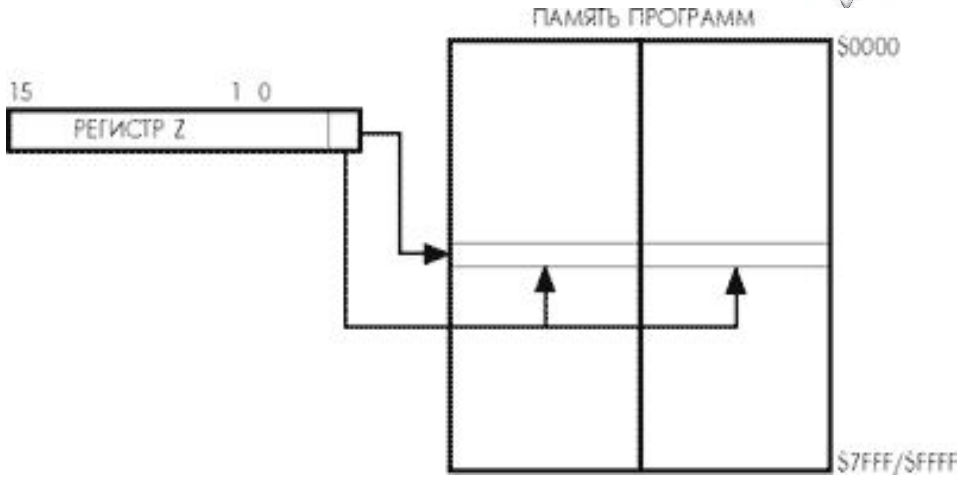
Косвенная адресация данных с постинкрементом

После выполнения операции регистр X, Y или Z инкрементируется. Адресом операнда является содержимое X, Y или Z регистра предшествовавшее инкрементированию



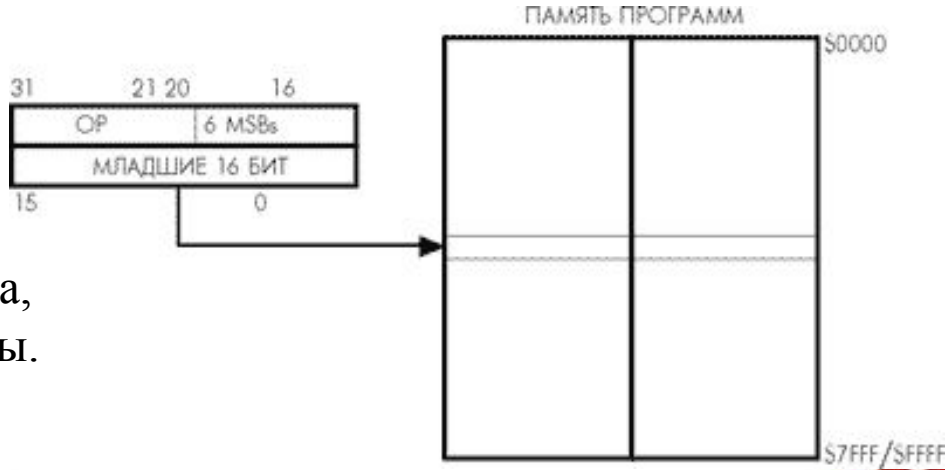
Адресация константы с использованием команд LPM и ELPM

Адрес байта константы определяется содержимым регистра Z. Старшие 15 битов определяют слово адреса (от 0 до 32К). Состояние младшего бита определяет выбор младшего байта (LSB = 0) или старшего байта (LSB = 1). При использовании команды ELPM младший бит (RAM Page) регистра Z - RAMPZ используется для выбора страницы памяти (RAMPZ0 = 0: младшая страница, RAMPZ0 = 1:старшая страница)..



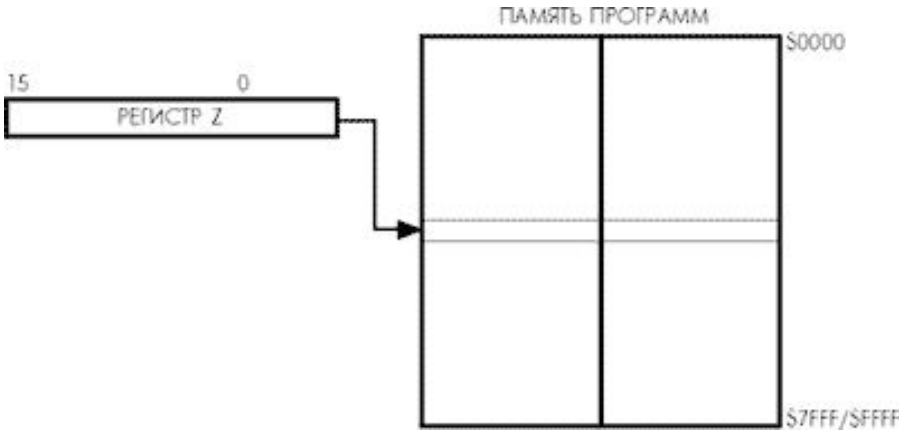
Непосредственная адресация памяти программ, команды JMP и CALL

Выполнение программы продолжается с адреса, записанного непосредственно в адресе команды.



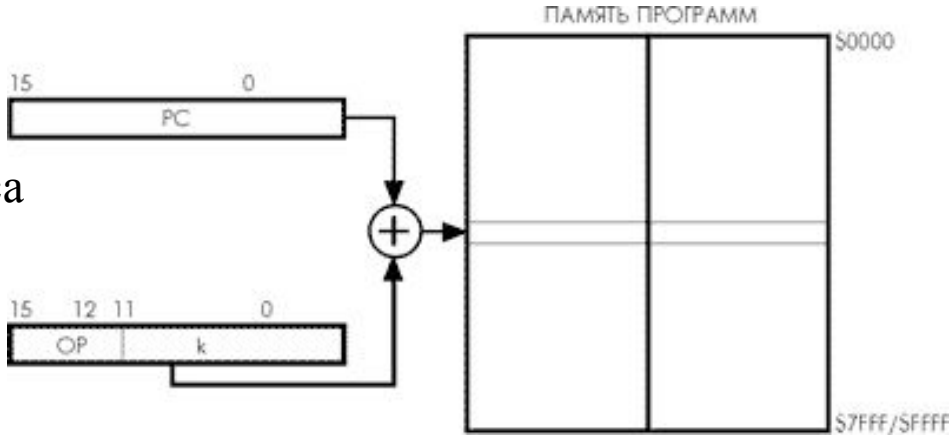
Косвенная адресация памяти программ, команды IJMP и ICALL

Выполнение программы продолжается с адреса, содержащегося в регистре Z (т.е. счетчик команд загружается содержимым регистра Z).



Относительная адресация памяти программ, команды RJMP и RCALL

Выполнение программы продолжается с адреса PC + k + 1. Значение относительного адреса может быть от -2048 до 2047.



Периферия AVR

В состав микроконтроллера помимо процессорного ядра и блоков памяти входят периферийные устройства, обеспечивающие различные дополнительные функции. К внутренней периферии относятся аналого-цифровые и цифро-аналоговые преобразователи, порты, таймеры, контроллеры интерфейсов UART, JTAG, SPI, и другие устройства.

Внешняя периферия представлена различными запоминающими устройствами, внешними аналого-цифровыми и цифро-аналоговыми преобразователями, средствами сопряжения с каналом связи, устройствами отладки и рядом других.

Порты ввода/вывода (I/O)

С каждым портом связаны три регистра ввода-вывода: регистр DDRx направления передачи данных, регистр PORTx данных порта и регистр PINx выводов порта, где x – имя порта (A…F). Содержимое разрядов регистра DDRx определяет направление передачи данных по каждой сигнальной линии порта (0 – порт используется для ввода данных, 1 – для вывода данных). Регистр PORTx позволяет задавать состояние сигнальных линий порта при выводе информации; регистр PINx – считывать состояние сигнальных линий порта при вводе информации. Т.о., для каждого физического вывода (пина) существует 3 бита контроля/управления.

Общая токовая нагрузка на все линии одного порта не должна превышать 80 мА для напряжения питания 5 В.

Прерывания (INTERRUPTS)

Все микроконтроллеры AVR имеют многоуровневую систему прерываний. Область векторов прерываний размещается в начале памяти программ; каждый вектор состоит из одной ячейки. Источниками всех прерываний являются аппаратные средства (внешние или внутренние); источники программных прерываний отсутствуют. Все источники прерываний являются маскируемыми.

Количество векторов прерываний в AVR-микроконтроллерах составляет от 3 до 35 в зависимости от типа.

Для каждого события может быть установлен приоритет. Выполняемая подпрограмма прерывания может быть прервана другим событием только при условии, что оно имеет более высокий приоритет, чем текущее.

Таймеры/счетчики (TIMER/COUNTERS)

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 (WatchDog Timer) предназначен для предотвращения катастрофических последствий от случайных сбоев программы. Он имеет свой собственный RC-генератор, работающий на частоте 1 МГц.

Аналоговый компаратор (AC)

Сравнивает напряжения на двух выводах (пинах) микроконтроллера. Результатом сравнения будет логическое значение, которое может быть прочитано программно.

Аналого-цифровой преобразователь (A/D CONVERTER)

Служит для получения числового значения напряжения, поданного на его вход. Этот результат сохраняется в регистре данных АЦП. Основные характеристики:

- Разрешение 10 разрядов
- Точность ±1/2 LSB
- Время преобразования 70...280 мс
- 8 мультиплексируемых каналов входа
- Режимы циклического и однократного преобразования

Прерывание по завершению ADC преобразования

Тактовый генератор

Тактовый генератор вырабатывает импульсы для синхронизации работы всех узлов микроконтроллера. Внутренний тактовый генератор AVR может запускаться от нескольких источников опорной частоты (внешний генератор, внешний кварцевый резонатор, внутренняя или внешняя RC-цепочка). Минимальная допустимая частота ничем не ограничена (вплоть до пошагового режима). Максимальная рабочая частота определяется конкретным типом микроконтроллера и указывается Atmel в его характеристиках

Система реального времени (RTC)

Таймер/счетчик RTC имеет отдельный предделитель, который может быть программным способом подключен или к источнику основной тактовой частоты, или к дополнительному асинхронному источнику опорной частоты (кварцевый резонатор или внешний синхросигнал). Для этой цели зарезервированы два вывода микросхемы.

Питание

AVR функционируют при напряжениях питания от 1,8 до 6,0 Вольт. Ток потребления в активном режиме зависит от величины напряжения питания и частоты, на которой работает микроконтроллер, и составляет менее 1 мА для 500 кГц, 5 … 6 мА для 5 МГц и 8 … 9 мА для частоты 12 МГц. AVR могут быть переведены программным путем в один из трех режимов пониженного энергопотребления.

Режим холостого хода (IDLE).

Прекращает работу только процессор и фиксируется содержимое памяти данных, а внутренний генератор синхросигналов, таймеры, система прерываний и сторожевой таймер продолжают функционировать. Ток потребления не превышает 2,5 мА на частоте 12 МГц.

Стоповый режим (POWER DOWN).

Сохраняется содержимое регистрового файла, но останавливается внутренний генератор синхросигналов, и, следовательно, останавливаются все функции, пока не поступит сигнал внешнего прерывания или аппаратного сброса. При включенном сторожевом таймере ток потребления в этом режиме составляет около 80 мкА, а при выключенном – менее 1 мкА. (Все приведенные значения справедливы для напряжения питания 5 В).

Экономичный режим (POWER SAVE).

Продолжает работать только генератор таймера, что обеспечивает сохранность временной базы. Все остальные функции отключены.

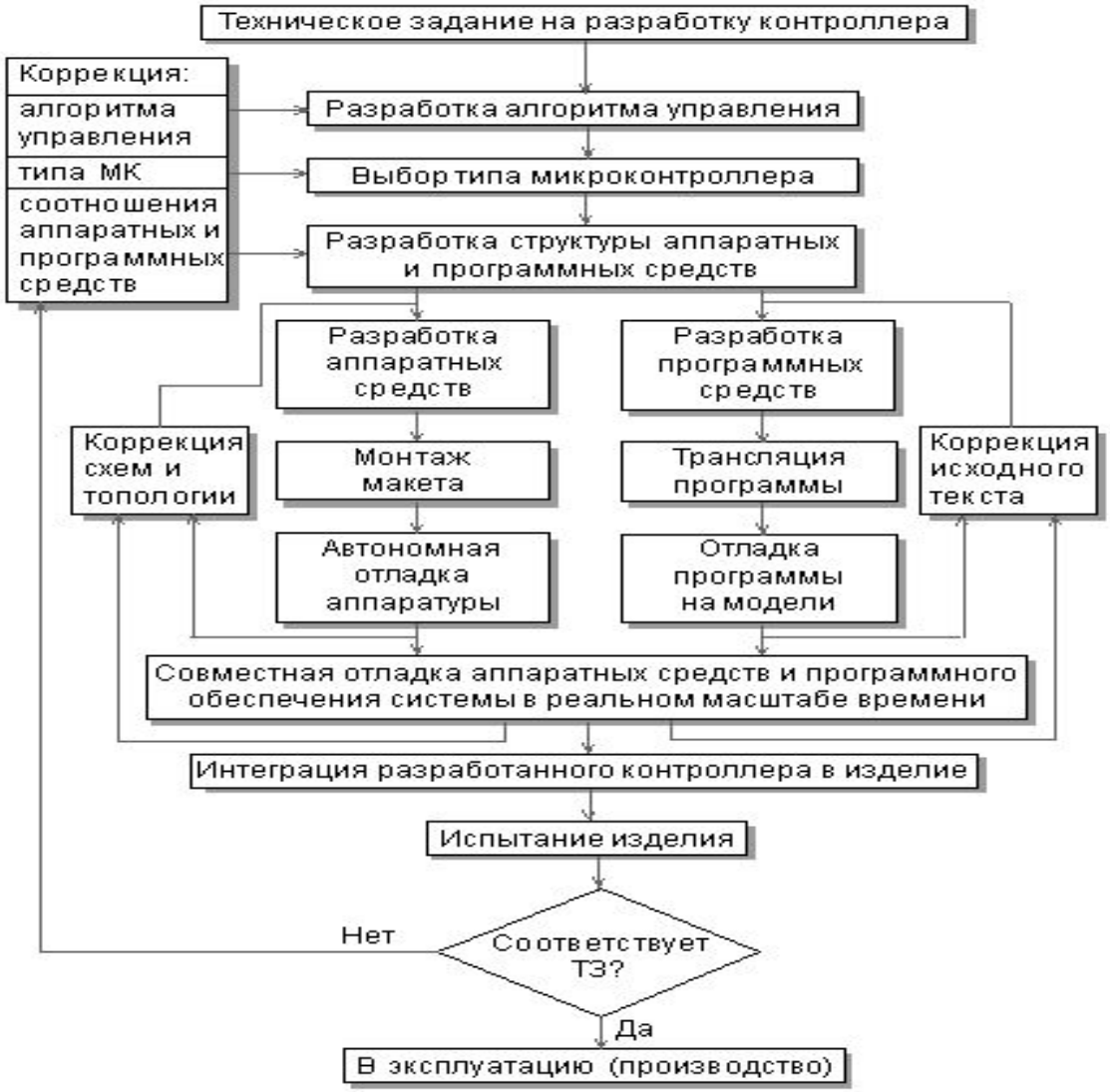
Основные этапы разработки микропроцессорной системы на основе микроконтроллеров

МПС на основе МК используются чаще всего в качестве встроенных систем для решения задач управления некоторым объектом. Важной особенностью данного применения является работа в реальном времени, т.е. обеспечение реакции на внешние события в течение определенного временного интервала. Такие устройства получили название *контроллеров.*

Технология проектирования контроллеров на базе МК полностью соответствует принципу неразрывного проектирования и **отладки аппаратных и программных средств**, принятому в микропроцессорной технике.

Перед разработчиком такого рода МПС стоит задача реализации полного цикла проектирования, начиная от **разработки** алгоритма функционирования и заканчивая **комплексными испытаниями** в составе изделия, а, возможно, и сопровождением при производстве.

Сложившаяся к настоящему времени методология проектирования контроллеров



В техническом задании формулируются требования к контроллеру с точки зрения реализации определенной функции управления. Техническое задание включает в себя набор требований, который определяет, что пользователь хочет от контроллера и что разрабатываемый прибор должен делать

На основании требований пользователя составляется **функциональная спецификация**, которая определяет функции, выполняемые контроллером Она включает в себя описания форматов данных, как на входе, так и на выходе, а также внешние условия, управляющие действиями контроллера.

Функциональная спецификация и требования пользователя являются критериями оценки функционирования контролера после завершения проектирования.

Этап разработки алгоритма управления является наиболее ответственным, поскольку ошибки данного этапа обычно обнаруживаются только при испытаниях законченного изделия и приводят к необходимости дорогостоящей переработки всего устройства. Разработка алгоритма обычно сводится к выбору одного из нескольких вариантов, отличающихся соотношением объема программного обеспечения и аппаратных средств

Критерием выбора является возможность максимальной реализации заданных функций программными средствами при минимальных аппаратных затратах и при условии обеспечения заданных показателей быстродействия и надежности в полном диапазоне условий эксплуатации.

При выборе типа МК учитываются следующие основные характеристики:

- разрядность;
- быстродействие;
- набор команд и способов адресации;
- требования к источнику питания и потребляемая мощность в различных режимах;
- объем ПЗУ программ и ОЗУ данных;
- возможности расширения памяти программ и данных;
- наличие и возможности периферийных устройств, включая средства поддержки работы в
- реальном времени (таймеры, процессоры событий и т.п.);
- возможность перепрограммирования в составе устройства;
- наличие и надежность средств защиты внутренней информации;
- возможность поставки в различных вариантах конструктивного исполнения;
- стоимость в различных вариантах исполнения;
- наличие полной документации;
- наличие и доступность эффективных средств программирования и отладки МК;
- количество и доступность каналов поставки, возможность замены изделиями других фирм.

На этапе разработки структуры контроллера окончательно определяется состав имеющихся и подлежащих разработке *аппаратных модулей*, протоколы обмена между модулями, типы разъемов. Выполняется предварительная проработка конструкции контроллера. *В части программного обеспечения* определяются состав и связи программных модулей, язык программирования. На этом же этапе осуществляется выбор средств проектирования и отладки.

После разработки структуры аппаратных и программных средств дальнейшая работа над контроллером может быть распараллелена.

Разработка аппаратных средств включает в себя разработку общей принципиальной схемы, разводку топологии плат, монтаж макета и его автономную отладку

Этап разработки программного обеспечения, его трансляции и отладки на моделях существенно зависит от используемых системных средств. В настоящее время самым мощным средством разработки программного обеспечения для МК являются интегрированные среды разработки, имеющие в своем составе менеджер проектов, текстовый редактор и симулятор, а также допускающие подключение компиляторов языков высокого уровня типа Си.

Этап совместной отладки аппаратных и программных средств в реальном масштабе времени является самым трудоемким и требует использования инструментальных средств отладки. К основным из них относятся:

- внутрисхемные эмуляторы;
- платы развития (оценочные платы);
- мониторы отладки;
- эмуляторы ПЗУ

Этап интеграции разработанного контроллера в изделие заключается в повторении работ по совместной отладке аппаратуры и управляющей программы, но при работе в составе изделия, питании от штатного источника и с информацией от штатных источников сигналов и датчиков.

Состав и объем испытаний разработанного и изготовленного контроллера зависит от условий его эксплуатации и определяется соответствующими нормативными документами. Проведение испытаний таких функционально сложных изделий, как современные контроллеры, может потребовать разработки специализированных средств контроля состояния изделия во время испытаний.