# Цифровая схемотехника
# Интерфейсы

КУРС ЛЕКЦИЙ

ЧУ ПО «СОЦИАЛЬНО-ТЕХНОЛОГИЧЕСКИЙ КОЛЛЕДЖ»

ПРЕПОДАВАТЕЛЬ: БОРИСОВ АЛЕКСЕЙ АЛЬБЕРТОВИЧ

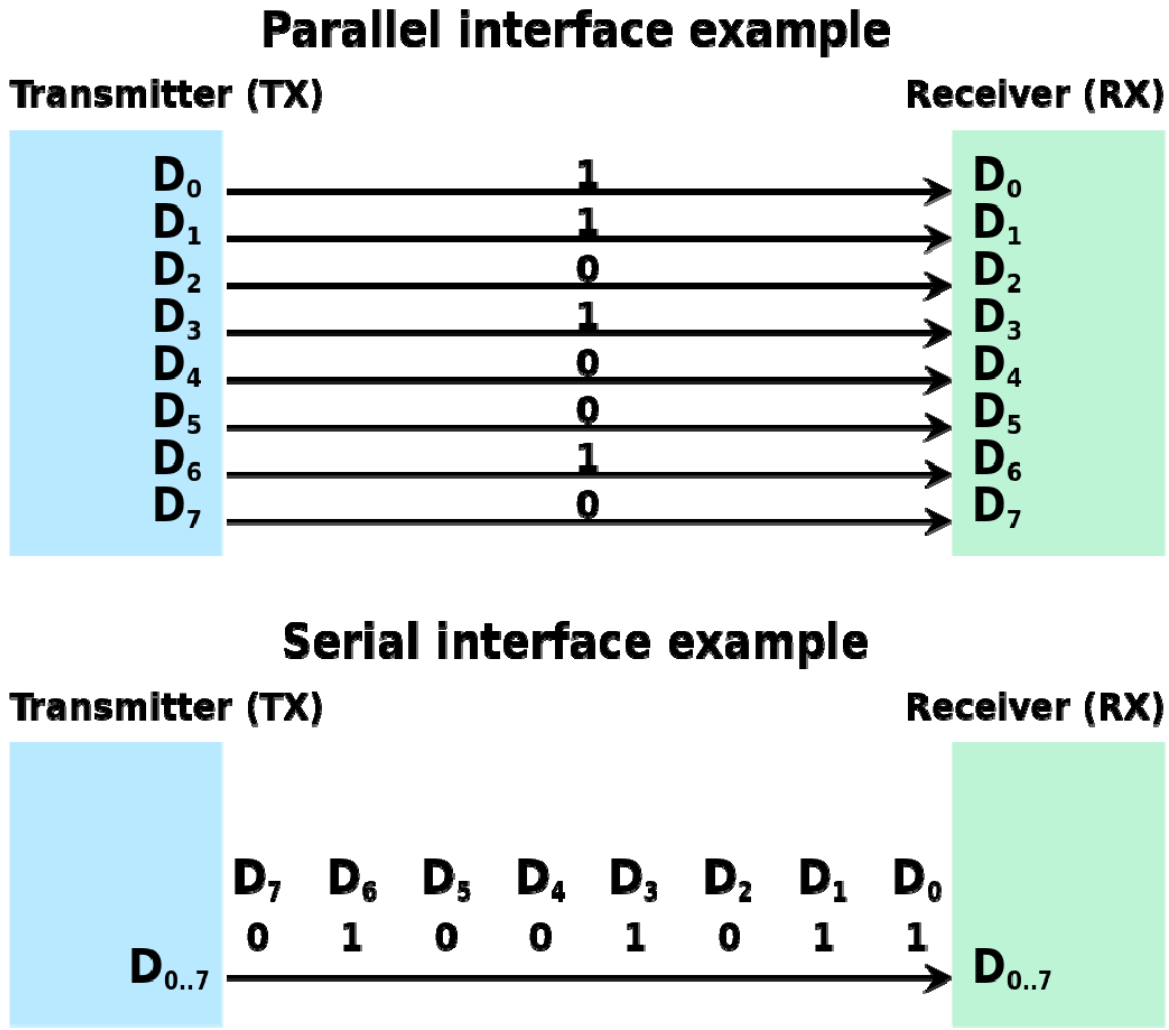
RAZUMDOM

СОЦИАЛЬНО ТЕХНОЛОГИЧЕСКИЙ КОЛЛЕДЖ

# Программа занятий

1. Вводная часть
2. Типы интерфейсов
3. Параллельные интерфейсы
4. Последовательный интерфейс SPI
5. Последовательный интерфейс I2C
6. Последовательный интерфейс 1-WIRE

# Интерфейсы связи



# Последовательные интерфейсы

## Последовательный периферийный интерфейс

Последовательный периферийный интерфейс – Serial Peripheral Interface или SPI-интерфейс. Данный интерфейс используется для работы с различными периферийными устройствами, например, это могут быть различные ЦАП/АЦП, цифровые потенциометры, различные датчики, расширители портов ввода/вывода (GPIO), Flash и EEPROM память и даже более сложная периферия, такая как, например, звуковые кодеки и контроллеры Ethernet и так далее.




Интерфейс был разработан компанией Motorola, но в настоящий момент используется всеми производителями. Данный интерфейс отличают простота использования и реализации, высокая скорость обмена, но малая дальность действия.

При любом обмене данными по интерфейсу SPI одно из устройств является **ведущим** (Master'ом), а другое **ведомым** (Slave'ом). Обычно (но не всегда) в роли ведущего выступает микроконтроллер.

Ведущий переводит ведомое (периферийное) устройство в активное состояние и формирует тактовый сигнал и данные. В ответ ведомое устройство передает ведущему свои данные. Передача данных в обе стороны (дуплексная) происходит синхронно с тактовым сигналом.

SPI представляет собой четырехпроводную синхронную шину, предназначенную для последовательного обмена данными между микросхемами.
SPI представляет собой соединение двух (или более) синхронных сдвиговых регистров, которые являются центральным элементом любого SPI устройства. Для соединения используется конфигурацию ведущий/ведомый.

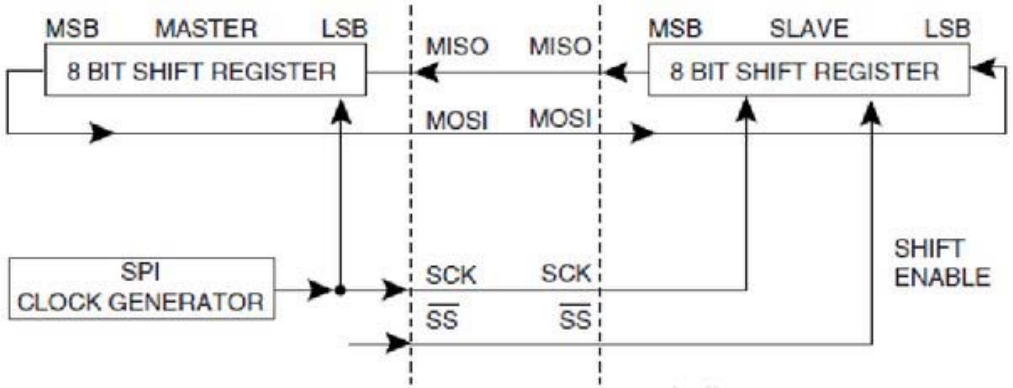



Рисунок 1 – Структурная схема интерфейса SPI

Сигналы, используемые данным интерфейсом, имеют следующее назначение:

- MOSI — Master Output / Slave Input. Выход ведущего / вход ведомого. Служит для передачи данных от ведущего устройства к ведомому.
- MISO – Master Input / Slave Output. Вход ведущего / выход ведомого. Служит для передачи данных от ведомого устройства к ведущему.
- SLK — Serial Clock. Сигнал синхронизации. Служит для передачи тактового сигнала всем ведомым устройствам.
- SS — Slave Select. Выбор ведомого. Служит для выбора ведомого устройства.

Производители микросхем часто используют другие названия для этих сигналов.
Альтернативные варианты могут быть такими:

MOSI – DO, SDO, DOUT. MISO – DI, SDI, DIN. SCK – CLK, SCLK. SS – CS, SYNC.

При подаче импульсов синхронизации на выход SCK, данные выталкиваются ведущим с его выхода MOSI, и захватываются ведомым по его входу MOSI. И обратно, данные передаются ведомым с выхода MISO и захватываются ведущим со его входа MISO.

Таким образом, если подать количество импульсов синхронизации соответствующее разрядности сдвигового регистра, то данные в обоих регистрах обменяются местами. Отсюда следует что SPI всегда работает в полнодуплексном режиме.

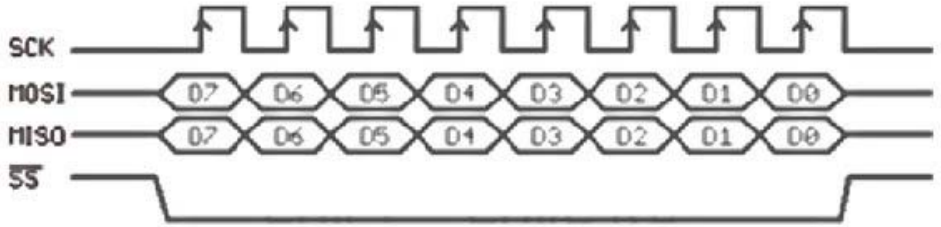
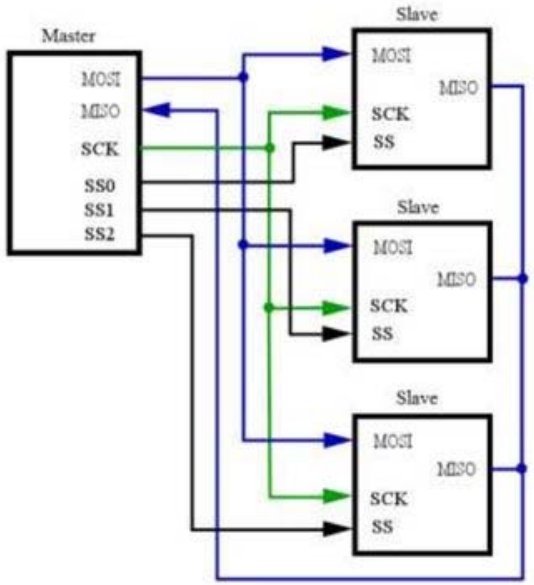



Рисунок 2 – Временная диаграмма работы интерфейса SPI

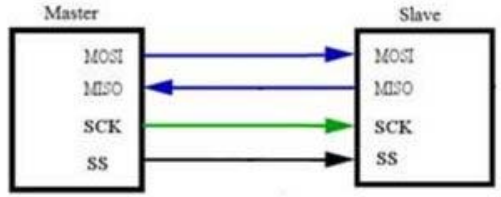
Контроллер SPI, как правило, реализуется специальным периферийным блоком в микроконтроллере. В большинстве чипов он программируется и может работать как в режиме ведущего, так и в режиме ведомого.

## Схемы соединений по SPI

Типовая схема соединения устройств по SPI (рисунок 3).




а)




б)

Рисунок 3 – Типовая схема соединения устройств по SPI-интерфейсу

К ведущему все ведомые подключаются параллельно, за исключением сигнала выбора ведомого (SSx). Для каждого ведомого необходим отдельный сигнал выбора ведомого (рисунок 3, а). Для сигналов выбора ведомого могут использоваться как специально предназначенные для этого выходы SPI-контроллера, так и порты ввода/вывода общего назначения (GPIO) микроконтроллера.

Частным случаем независимого подключения является вариант с одним единственным ведомым (рисунок 3, б). Крайне не рекомендуется подтягивать сигнал SS к земле, чтобы устройство всегда было в активном состоянии, так как ведомое устройство может использовать сигнал SS для инициализации или для других служебных целей.

Основное неудобство при независимом подключении ведомых (см. рисунок 3) в том, что для каждого из ведомых необходим отдельный сигнал выбора ведомого (SS).
Каскадная схема подключения, в зарубежной литературе называемая «daisy-chain» (можно перевести как «гирлянда»), лишена такого недостатка. Каскадная схема подключения устройств по SPI (рисунок 4).

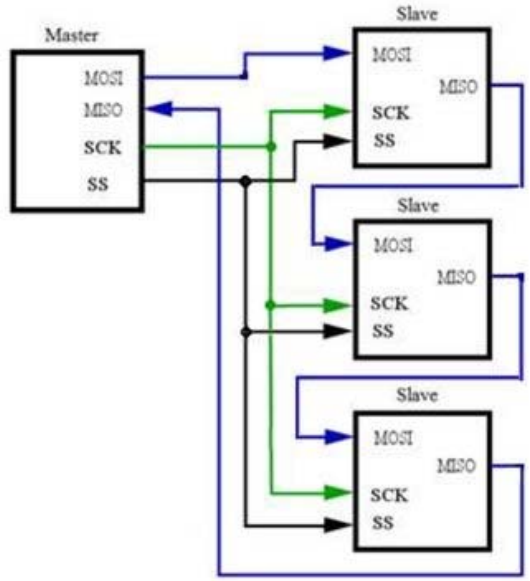



Рисунок 4 – Каскадная схема соединения устройств по SPI-интерфейсу

Как видно из рисунка 4, в каскадной схеме подключения используется общий сигнал выбора ведомого для всех ведомых. Выход каждого из ведомых соединяется со входом следующего. Выход последнего ведомого соединяется со входом ведущего, таким образом образуется замкнутая цепь. При таком подключении можно считать что последовательно соединённые устройства образуют один большой сдвиговый регистр. Соответственно, данные можно записать сразу во все устройства, предварительно собрав нужный пакет, объединяющий данные для каждого из устройств в порядке соответствующем физическому порядку соединения.

Сокращенный (однонаправленны) вариант схемы подключения устройств по SPI (рисунок 5).

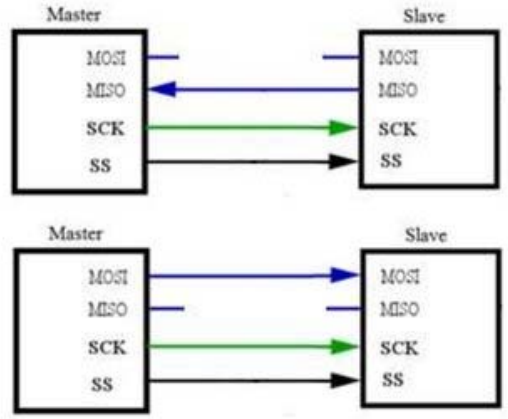



Рисунок 5 – Сокращенная схема соединения устройств по SPI-интерфейсу

Сокращенный вариант схемы подключения, когда линия MOSI или MISO не используется, то есть передача данных осуществляется только в одну сторону используются, например, при подключении к микроконтроллеру внешних микросхем ЦАП и АЦП.

## Протокол обмена по SPI

Протокол обмена по SPI аналогичен логике работы сдвигового регистра и заключается в последовательном побитном выводе/вводе данных по определенным фронтам тактового сигнала.

Установка данных и выборка осуществляется по противоположным фронтам тактового сигнала.

Спецификация SPI предусматривает 4 режима передачи данных, которые отличаются между собой соотношением фазы и полярности тактового сигнала и передаваемых данных.

Эти режимы описываются двумя параметрами:

- **CPOL** – clock polarity. Полярность тактового сигнала — определяет исходный уровень сигнала синхронизации
- **CPHA** – clock phase. Фаза тактового сигнала — определяет последовательность установки и выборки данных.

### SPI mode 0

CPOL = 0, CPHA=0. Тактовый сигнал начинается с уровня логического нуля. Защелкивание данных выполняется по нарастающему фронту. Смена данных происходит по падающему фронту. Моменты защелкивание данных показаны на рисунках стрелочками

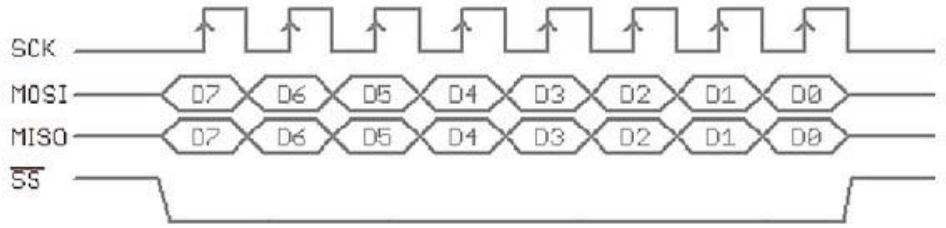



Рисунок 6 – Временная диаграмма работы SPI-интерфейса в режиме mode 0.

**SPI mode 1**

CPOL = 0, CPHA=1. Тактовый сигнал начинается с уровня логического нуля. Смена данных происходит по нарастающему фронту. Защелкивание данных выполняется по падающему фронту.

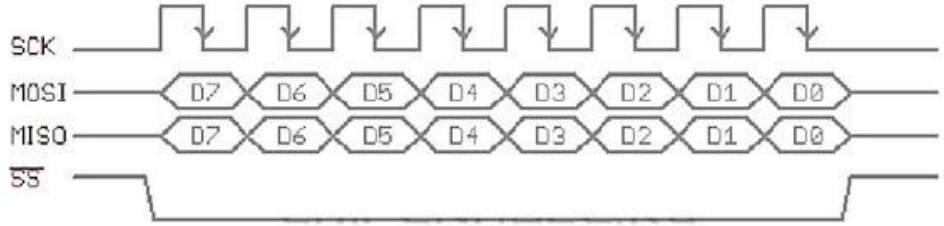



Рисунок 7 – Временная диаграмма работы SPI-интерфейса в режиме mode 1.

**SPI mode 2**

CPOL = 1, CPHA=0. Тактовый сигнал начинается с уровня логической единицы. Защелкивание данных выполняется по падающему фронту. Смена данных выполняется по нарастающему фронту тактового сигнала.

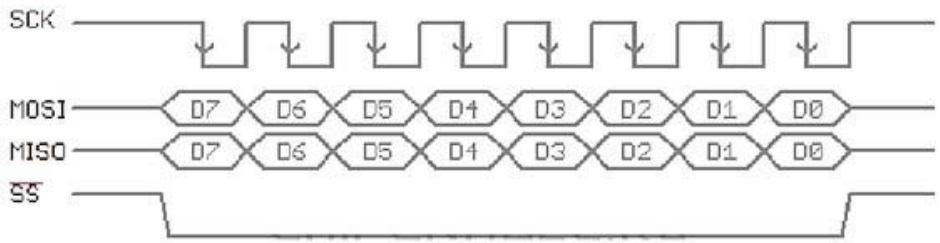



Рисунок 8 – Временная диаграмма работы SPI-интерфейса в режиме mode 2.

**SPI mode 3**

CPOL = 1, CPHA=1. Тактовый сигнал начинается с уровня логической единицы. Смена данных выполняется по падающему фронту тактового сигнала. Защелкивание данных выполняется по нарастающему фронту.

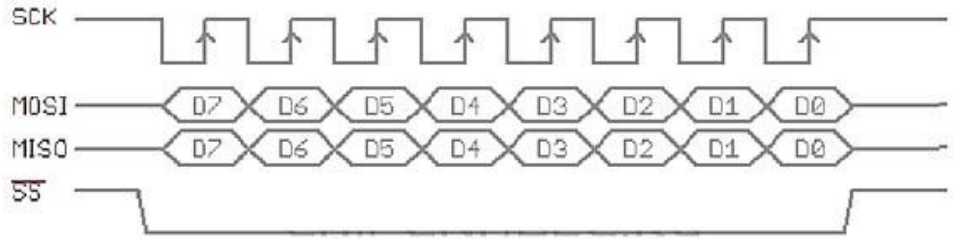



Рисунок 9 – Временная диаграмма работы SPI-интерфейса в режиме mode 3.

Современные микроконтроллеры поддерживают все четыре режима работы SPI.

Стоит отметить, что:

- передача данных по SPI может происходить как старшим битом вперед, так и младшим.
- количество байт передаваемых за время удержания сигнала выбора (SS) ничем не ограничено и определяется спецификацией используемого ведомого устройства.
- в спецификации на ведомое устройство указываются:
  - поддерживаемые режимы работы SPI,
  - максимальная частота тактового сигнала,
  - содержимое передаваемых или принимаемых данных.

# Цепи SPI в лабораторном стенде UNI-DS3

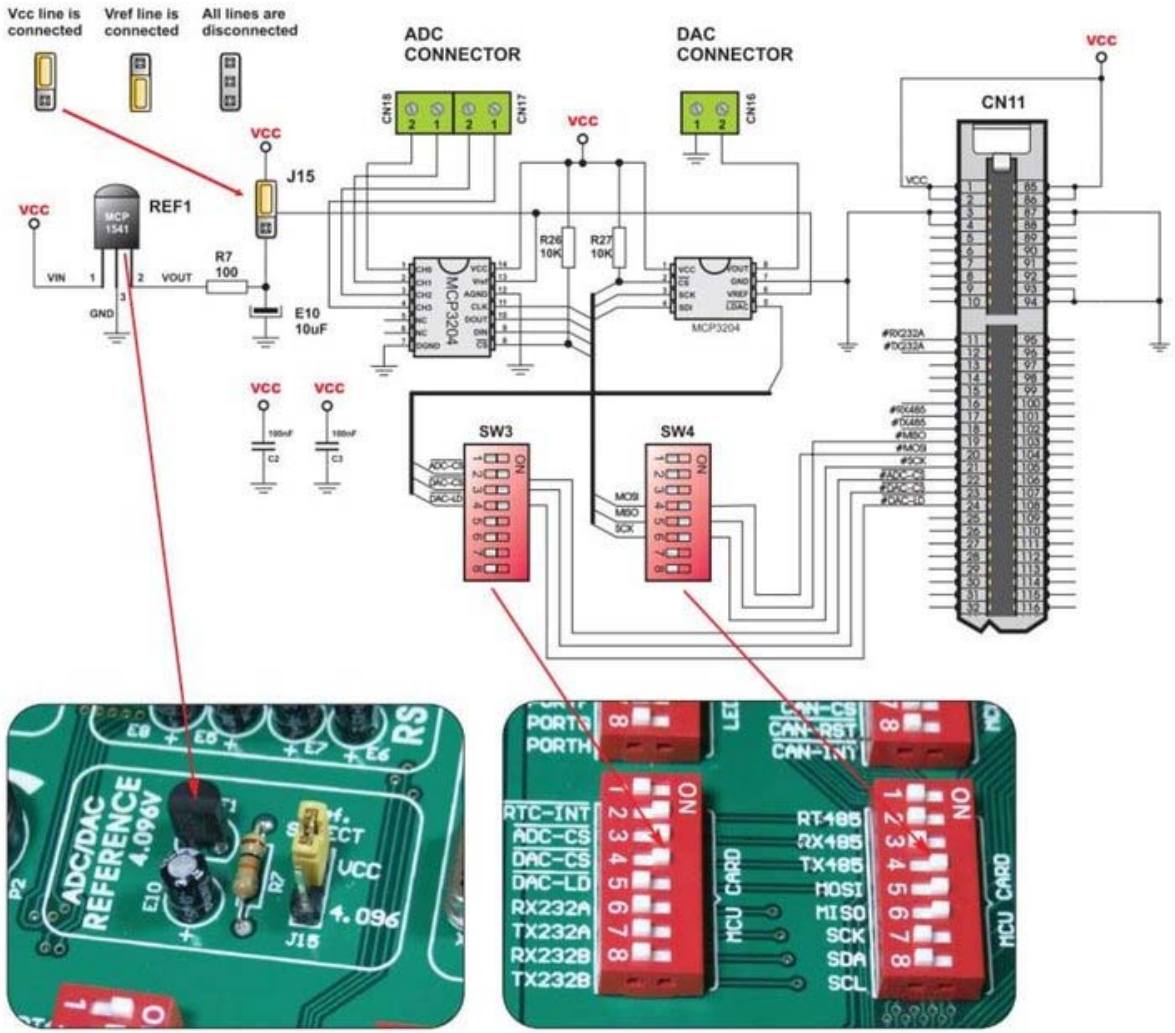



Рисунок 10 – Использование интерфейса SPI в лабораторном стенде UNI-DS3.

# Интерфейсная шина

I2C

## Двухпроводный последовательный интерфейс

**Двухпроводный последовательный интерфейс** $I^2C$ (Inter-Integrated Circuit) — последовательная шина данных для связи интегральных схем. Протокол передачи данных по шине $I^2C$ гарантирует надежную и качественную прием/передачу данных. Интерфейс разработан корпорацией Philips в начале 1980-х как простая шина внутренней связи для создания управляющей электроники. Существует 3 версии интерфейса в основном различающиеся по скорости передачи данных и режимом разрядной адресации:

- режим медленной передачи (low-speed) 100 кбит/с, режим быстрой передачи (fast-speed) со скоростью до 400 кбит/с и быстродействующий режим (Hs-mode) со скоростью передачи до 3,4 Мбит/с. Также имеются режимы 7-и и 10-и битной адресации.

По всему миру интерфейс $I^2C$ стал международным стандартом. Было разработано более 1000 интегральных схем, лицензия на официальное использование спецификации приобретена более чем 50 фирмами. Полная гамма микросхем с интерфейсом $I^2C$, выпускаемых фирмой, насчитывает в настоящее время более 150 наименований, выполненных с применением как перспективной КМОП-технологии, так и с уже ставшей традиционной — биполярной.

Важнейшим критерием, определяющим возможность использования той или иной коммуникационной шины, является спектр её технических характеристик.

Шина $I^2C$ относится к классу **двунаправленных асинхронных шин с последовательной передачей данных** и, как следствие, обладает достаточно низкой пропускной способностью. Поэтому её почти не используют в составе персональных компьютеров, а только как вспомогательную для идентификации установленных устройств.

Наибольшее применение шина $I^2C$ нашла для согласования работы устройств, наполняющих изделия бытовой и встроенной техники, где она вполне годится.

Основные технические характеристики шины $I^2C$ по спецификации 1.0 приведены в таблице 1.

Таблица 1 – Основные технические характеристики шины $I^2C$ (спецификация 1.0)

| Наименование параметра | Значение параметра |
|---|---|
| Скорость обмена low-speed | не более 100 кбит/с |
| Скорость обмена fast-speed | не более 400 кбит/с |
| Число адресуемых устройств (7 бит) | до 128 |
| Суммарная длина линий SCL и SDA | не более 4 м |
| Суммарная паразитная емкость относительно общего провода | не более 400 пФ |
| Входная емкость на каждый вывод абонента | не более 10 пФ |

В таблице 2 представлены основные термины, связанные с шиной $I^2C$.

Таблица 2 – Основные термины, связанные с шиной $I^2C$

| Термин | Описание |
|---|---|
| Передатчик | Устройство, передающее данные по шине $I^2C$. |
| Приемник | Устройство, принимающее данные с шины $I^2C$. |
| Ведущий | Устройство, инициирующее передачу данных и формирующее тактовый сигнал. |
| Ведомый | Устройство, к которому обращается ведущий. |
| Конкуренция | Несколько ведущих на шине могут пытаться передать данные без разрушения текущего сообщения. |
| Арбитраж | Процедура, гарантирующая, что только один ведущий управляет шиной. |
| Синхронизация | Процедура синхронизации тактовых сигналов от двух или более устройств. |

При передаче данных одно устройство является "Ведущим" (Master), которое инициирует передачу данных и формирует сигналы синхронизации. Другое устройство "Ведомое" (Slave), которое может начать передачу данных только по команде ведущего шины (рисунки 11, 12).

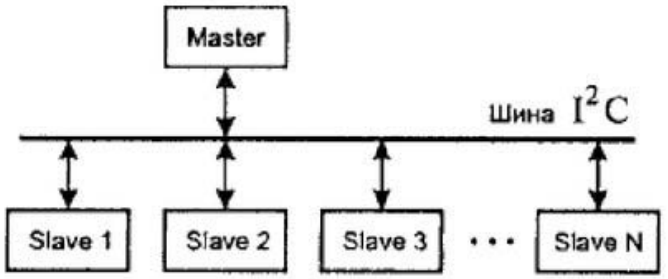



Рисунок 11 – Master-организация шины I²C

Спецификация I²C в принципе поддерживает режим multi-master (рисунок 13), когда к одной шине подключено несколько master-устройств. Но этот режим используется в аппаратуре нечасто из-за сложности доступа к шине, так как на шине может совершать операции только одно master-устройство, остальные «мастера» обязаны отключаться. В противном случае возникает ситуация, называемая **шинным конфликтом**. Из-за которого информация может попросту не дойти до адресата и нарушится работа устройства.

Для того чтобы исключить шинные конфликты, в режиме multi-master должны содержаться процедуры **арбитража** и **синхронизации,** устанавливающие порядок работы master-устройств.

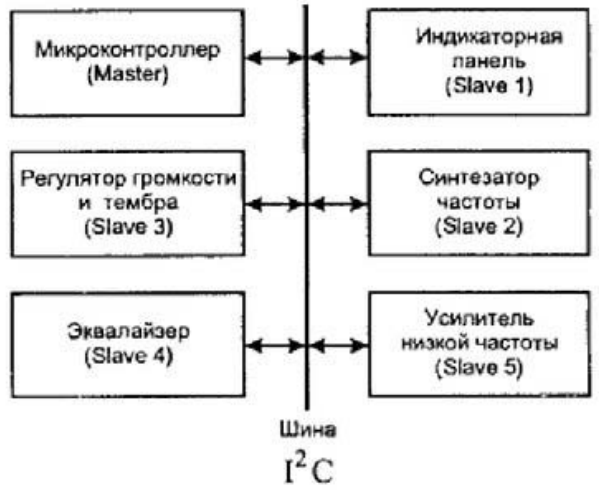



Рисунок 12 – Пример взаимодействия устройств на шине I²C

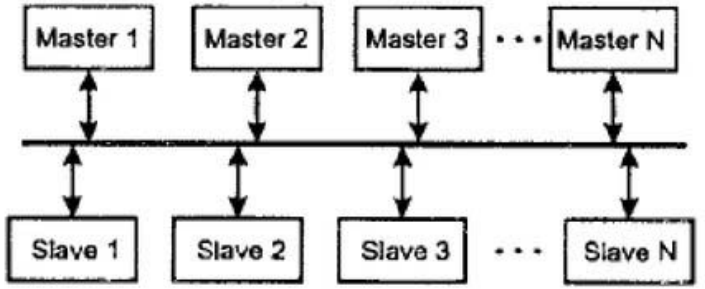



Рисунок 13 – Multi-master организация шины I²C

## Физический уровень.

Данные передаются по двум проводам — провод данных SDA и провод синхронизации SCL.
Выходы устройства формирующего сигналы данных (SDA) и синхронизации (SCL) должны быть типа «открытый коллектор», чтобы выполнять требования "монтажного И" на шине.
Для формирования высокого уровня сигнала на линиях к ним подключаются подтягивающие резисторы.
Число устройств, которые могут быть подключены к шине $I^2C$, ограничивается только максимальной емкостью шины (400 пФ) и способностью адресации этих устройств.

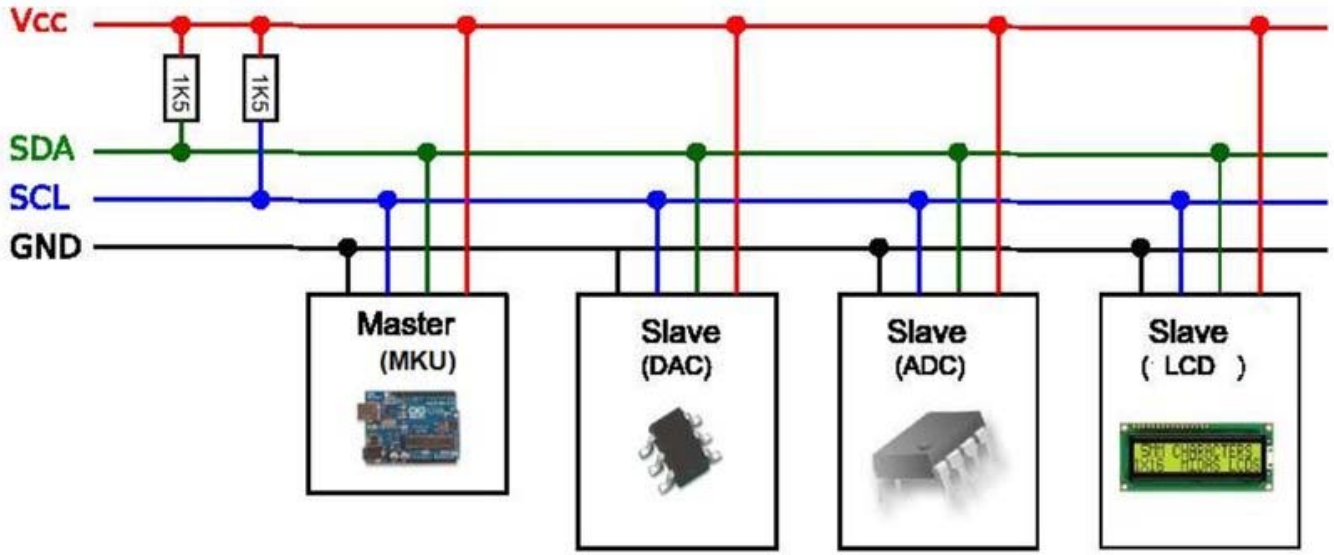



Рисунок 14 – Схема подключения устройств в шине I2C

Такты синхронизации генерирует ведущий (master), ведомый (slave) выдает сигнал подтверждения при приеме байта.
Всего на одной двухпроводной шине может быть до 127 устройств.

## Совместимость

**Совместимость** является важной технической характеристикой шины. Ранее разработанные элементы, обладающие только возможностями низкоскоростного обмена, должны без проблем связываться с высокоскоростными, и наоборот. К шине I²C могут быть подключены интерфейсы трех типов: low-speed, fast-speed, Hs-mode. Обмен данными может быть осуществлен со скоростью, доступной самому медленному интерфейсу. В таблице 3 приведены возможные предельные скорости обмена по совмещенной шине.

Таблица 3. Скорость обмена данными в совмещенных шинах

| Направление передачи | Конфигурация шины I²C | | | |
|---|---|---|---|---|
| | Hs + fast + low | Hs + fast | Hs + low | fast + low |
| Hs – Hs | 0...3,4 Мбит/с | 0...3,4 Мбит/с | 0...3,4 Мбит/с | – |
| Hs – fast | 0...100 кбит/с | 0...400 кбит/с | – | – |
| Hs – low | 0...100 кбит/с | – | 0...100 кбит/с | – |
| fast – low | 0...100 кбит/с | – | – | 0...100 кбит/с |
| fast – fast | 0...100 кбит/с | 0...400 кбит/с | – | 0...100 кбит/с |
| low – low | 0...100 кбит/с | – | 0...100 кбит/с | 0...100 кбит/с |

Шина I²C также позволяет совмещать устройства с разными напряжениями питания как показано на рисунке 15.

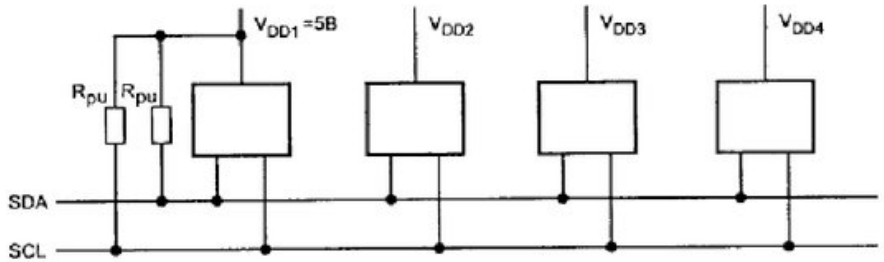



Рисунок 15 – Соединение элементов с разными напряжениями питания

## Протокол обмена данными

Передача любого бита по шине происходит при условии **стробирования** данных SDA по линии SCL.
Master-устройство выставляет бит данных «0» или «1» на линию SDA.
Slave-устройство фиксирует этот бит только тогда, когда на линии SCL произойдет перепад сигнала из низкого уровня в высокий (так называемый положительный перепад).
Отсюда следует первое правило организации протокола шины: **смена информации на линии SDA может быть произведена только при нулевом состоянии линии SCL.**

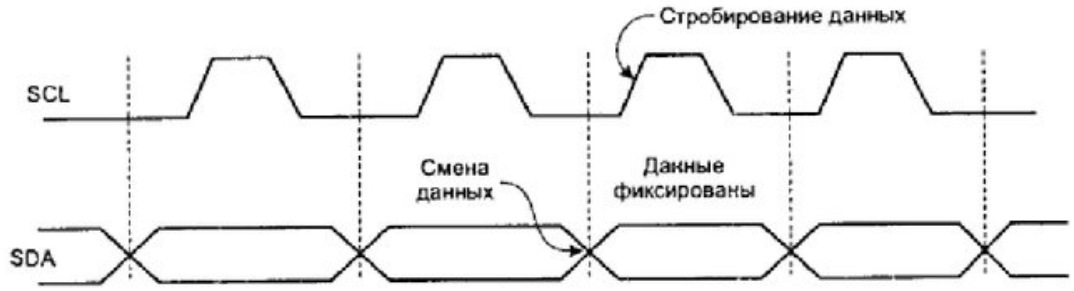



Рисунок 16 – Передача бита по шине 12C

В исходном состоянии оба сигнала SDA и SCL находятся в высоком состоянии.
Для распознавания начала и конца передачи в спецификацию шины были введены специальные условия СТАРТ (Start) и СТОП (Stop). В фирменной документации условие Start имеет условное сокращение «S», условие Stop — «P».

## Состояние СТАРТ и СТОП

Процедура обмена начинается с того, что ведущий формирует состояние СТАРТ — ведущий генерирует переход сигнала линии SDA из ВЫСОКОГО состояния в НИЗКОЕ при ВЫСОКОМ уровне на линии SCL (рисунок 17). Этот переход воспринимается всеми устройствами, подключенными к шине как признак начала процедуры обмена.

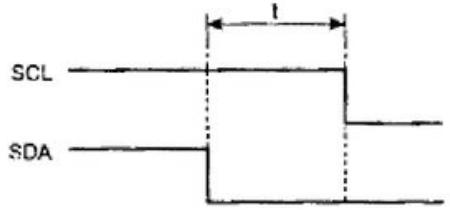



Рисунок 17 – Условие START

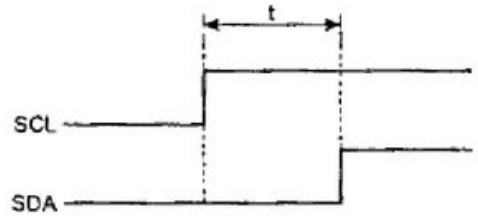



Рисунок 18 – Условие STOP

Условие СТОП возникает при положительном перепаде линии SDA при единичном состоянии линии SCL (рисунок 18).

Состояния СТАРТ и СТОП всегда должны генерироваться master-устройствами.

Укрупненно информационный пакет, передаваемый по шине $I^2C$, выглядит так, как показано на рисунок 19.


**Рисунок** 19 – Информационный пакет данных на шине I2C

## Передача данных

После отработки состояния Start передатчик последовательно выставляет на линии SDA данные, начиная со старшего бита (MSB) и заканчивая младшим (LSB). Передача данных по шине производится по 8 битов. Данные стробируются по линии SCL импульсами 1...8.

На 9-м такте следует *сигнал подтверждения* (acknowledge). Сигнал подтверждения свидетельствует о том, что данные нормально приняты и обработаны (рисунок 20).

В фирменной документации состояние *acknowledge* условно именуется буквой «A», а в другой литературе ACK.

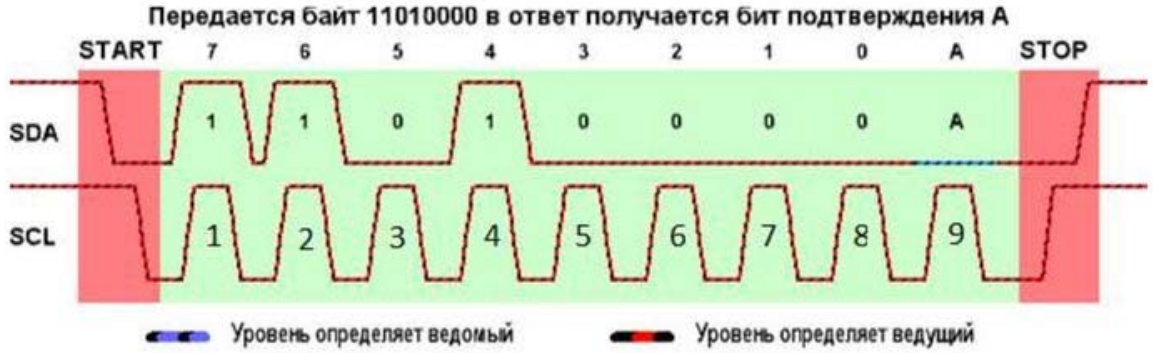



Рисунок 20 – Передача байта по шине $I^2C$ с приемом сигнала подтверждения

Когда ведущий шины принимает данные, то на каждый принимаемый байт формируется бит подтверждения, если принятый байт не последний.

Для сообщения ведомому о том, что ведущий прекращает принимать данные по приему последнего байта ACK не формируется. Ведомый отпускает SDA, чтобы ведущий смог передать бит STOP. Ведущий может формировать бит STOP на месте бита подтверждения.

В случае неподтверждения нормального приема (сигнал АСК имеет высокий уровень) передатчику желательно выполнить условие Stop и повторить передачу (рисунок 21).

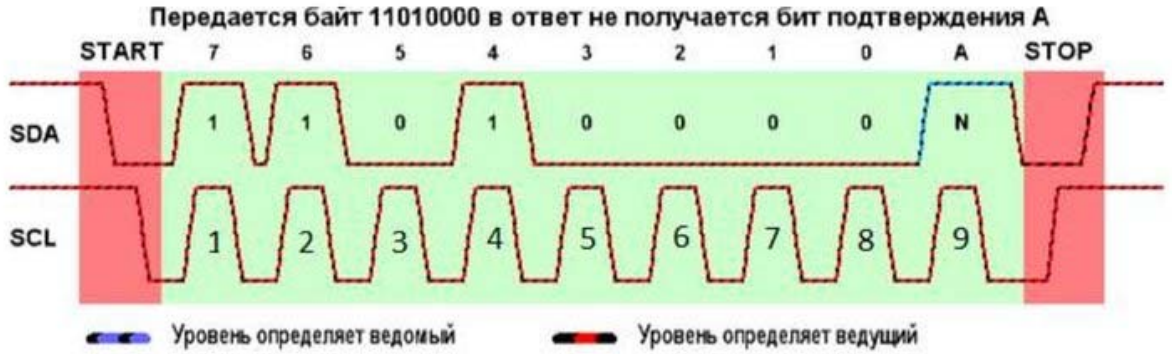



Рисунок 21 – Передача байта по шине I²C с отсутствием сигнала подтверждения

Также может возникнуть ситуация связанная с задержкой обработки данных у slave-абонента при получении сигнала подтверждения (АСК), как показано на рисунок 22.

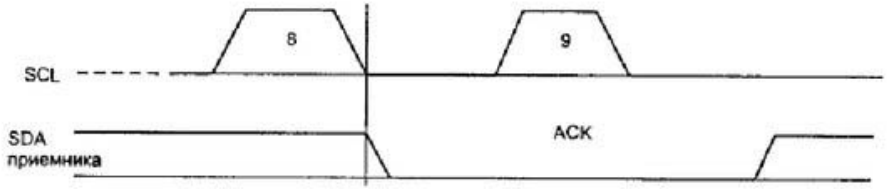



Рисунок 22 – Задержка состояния acknowledge

Задержка освобождения линии SDA не свидетельствует о неправильном обмене информацией, поэтому master-абоненту достаточно дождаться окончания АСК и продолжить передачу.
Если ведомому необходимо задержать передачу данных, то он может удерживать SCL в низком логическом уровне. Передача данных продолжится, когда ведомый отпустит SCL. Это позволяет ведомому подготовить новые данные для передачи.

## Форматы передачи данных на шине $I^2C$

**Формат (протокол) передачи данных** — *набор соглашений интерфейса логического уровня, которые определяют обмен данными между различными программами.*
*Эти соглашения задают единообразный способ передачи сообщений и обработки ошибок при взаимодействии программного обеспечения разнесённой в пространстве аппаратуры, соединённой тем или иным интерфейсом.*

Каждое устройство на шине $I^2C$ должно иметь свой **уникальный адрес,** по которому к нему можно обратиться.
Адрес устройства всегда располагается в определенном месте (в определенных ячейках внутренней памяти).
Адрес может быть **назначен,** а может быть **жёстко определен.**

На шине $I^2C$ адреса устройств, называемые SAVE-адресами, **жёстко определены** при изготовлении микросхем и не подлежат переназначению, то есть модификации. Иногда, для некоторых микросхем переназначить адрес можно (это делается с помощью 2—3 бит, которые нужно предварительно определить **аппаратно**, то есть установить перемычки на соответствующие выводы).
Значение SLAVE-адреса можно узнать из документации на соответствующую микросхему.

Так как все абоненты шины обмениваются данными только по линиям SDA и SCL, то в момент начала передачи все SLAVE-абоненты «слушают» линию на предмет опознавания своего SLAVE-адреса. Опознавший свой адрес абонент продолжает прием данных и выдачу сигналов АСК, остальные только следят за моментом выдачи состояния STOP.

Для адресации устройств на шине $I^2C$ используется два формата адреса:

- **7-разрядный формат** с битом чтения/записи R/W;
- **10-разрядный формат** (передаются два байта)

## Адресация устройств на шине I2C

Возможны три формата передачи:

- MASTER транслирует данные на SLAVE;
- MASTER читает данные от SLAVE;
- комбинированный формат передачи(записи)/чтения.


S - Start
R/W - Чтение/запись
ACK - Подтверждение

Рисунок 23 – Формат протокола шины $I^2C$

## 7-и разрядный формат

Формат кадра при 7-битной адресации, предполагает передачу SLAVE-адреса 7 битами, а восьмой бит должен содержать признак операции «чтение/запись» (R/W).

При 7-битной адресации на шине может присутствовать только 128 устройств с уникальными адресами. Фактически устройств меньше, так как некоторые адреса зарезервированы под некоторые служебные функции.

На рисунке 24 представлен формат передачи данных от MASTER-устройства к SLAVE-абоненту.


Рисунок 24 - MASTER транслирует данные на SLAVE


Рисунок 25 - MASTER читает данные от SLAVE

Обратите внимание: после чтения информационного байта DATA MASTER-абонент обязан подтвердить получение байта сигналом АСК.

Комбинированные форматы используются для работы с последовательной памятью (для сокращения времени доступа к данным).

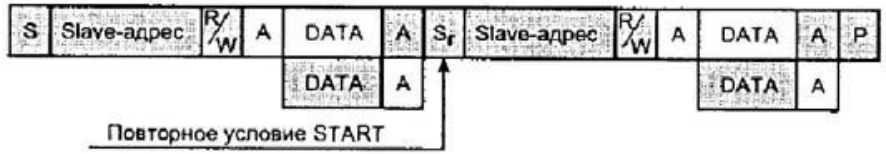



Рисунок 26 – Комбинированный формат

Единственное новшество, введенное в комбинированный формат — **повторное условие START** (*repeated Start condition*), обозначаемое на рисунке 4 сокращением «Sr». После него можно обратиться к другому устройству не освобождая шину.

Общий вызов может адресовать все устройства, подключенные к шине $I^2C$.

При получении общего вызова устройства должны подтверждать прием выдачей сигнала АСК. Устройства, которые не нуждаются в сведениях, передаваемых при общем вызове, могут игнорировать этот адрес, не выставляя сигнал АСК.

Если же устройство выполнено так, что обязательно требует данных общего вызова, оно ведет себя точно так же, как обыкновенный SLAVE-абонент. Второй байт, следующий за общим вызовом, обычно содержит информационную часть. На рисунке 6 представлен формат общего вызова.

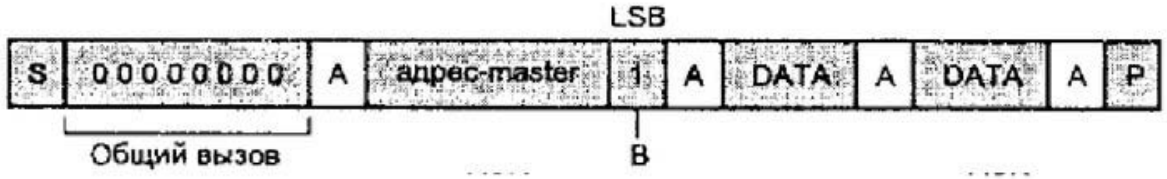



Рисунок 27 – Структура кадра общего вызова

Обратите внимание — информационная часть содержит бит B, который в сочетании с битами 7...1 (обозначенными на рис. 6 символом «X») регламентирует назначение информации, передаваемой в данном случае. Регламентируются следующие комбинации:

- 0000 011 0 (06h) — аппаратный сброс. При получении этой комбинации все абоненты, реагирующие на общий вызов, производят внутренний сброс (рестарт), но аппаратный сброс этих абонентов не должен блокировать шину;
- 0000 010 0 (04h) — запись программируемой части адреса SLAVE-абонента с помощью аппаратных средств. Все абоненты, ответившие на этот адрес, будут блокированы и не смогут быть сброшены;
- 0000 000 0 (00h) — этот код запрещается использовать;
- XXXX XXX 1 — используется, когда на шине присутствует много совмещенных MASTER/SLAVE-абонентов. В таких случаях часто необходимо генерировать запрос об адресе устройства, которому должна быть передана информация. К примеру, устройство «А», получив от устройства «Б» данные, обработав их, передает информацию устройству «В» вместе с собственным адресом (обычно адреса «MASTER/SLAVE» в совмещенных абонентах совпадают).

## 10-и разрядный формат

При использовании 10-и разрядного формата передается два байта.


Рисунок 28 – 10-и битный формат протокола $I^2C$

В первом байте передается: пять битов, определяющих, что это 10-разрядный адрес; два старших бита адреса; бит записи/чтения. Во втором байте передается 8 младших бит адреса (рисунок 28).

Адресация с помощью 10 разрядов аналогична 7-и разрядной адресации.

Устройство, получив служебный код в первом байте и опознав возможность приема 10-разрядного адреса, подтверждает это и принимает второй байт.
При совпадении принятого адреса с содержащимся внутри устройства собственным адресом выдается подтверждение АСК и ведется прием данных в обычном режиме до появления состояния STOP.
На рисунке 29 показан формат передачи данных от MASTER-абонента к SLAVE-устройству.


Рисунок 29 – Передача данных от MASTER-абонента к SALVE-устройству

На рисунке 30 приведен формат передачи данных от SLAVE-абонента к MASTER-устройству.

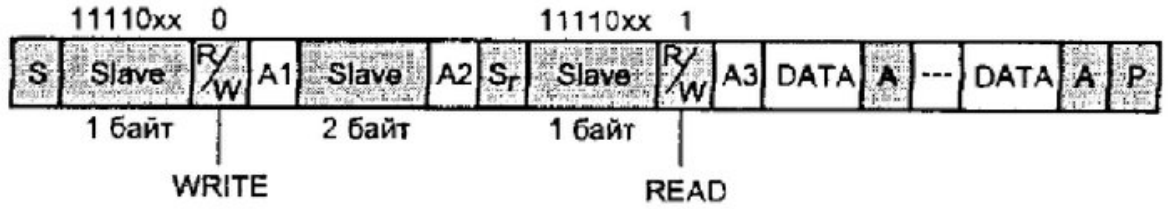



Рисунок 30 – Передача данных от SLAVE-абонента к MASTER-устройству

Вначале MASTER-устройство первым байтом адресует все SLAVE-абоненты соответствующим кодом, и они подтверждают его получение сигналом АСК (A1).

Затем вторым байтом адресуется конкретное устройство с выдачей сигнала АСК (A2).

После выполнения условия «**повторный START**» (Sr) адресованное SLAVE-устройство сохраняет возможность обращаться к нему, поэтому достаточно повторить первый байт адреса, но уже с другим значением бита R/W и получить АСК (A3).

## Комбинированный 10-и разрядный режим

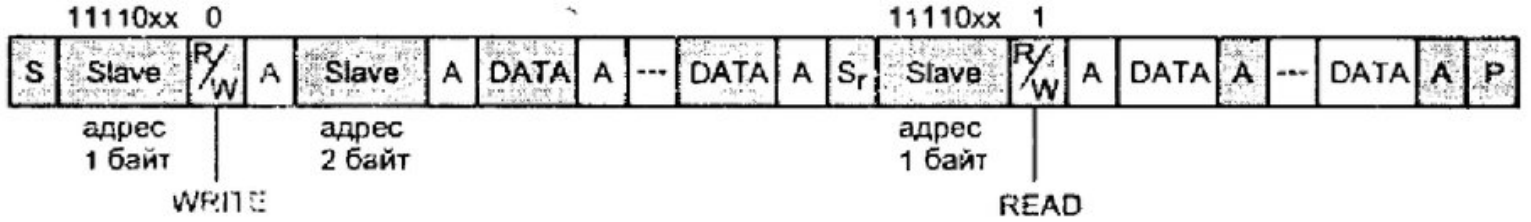



Рисунок 31 – Комбинированный формат. MASTER-абонент адресует SLAVE-устройство с 10-разрядным адресом, затем передает данные и читает данные

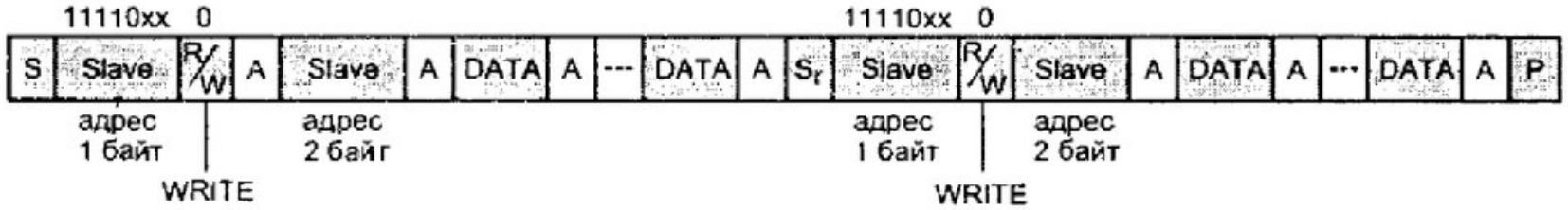
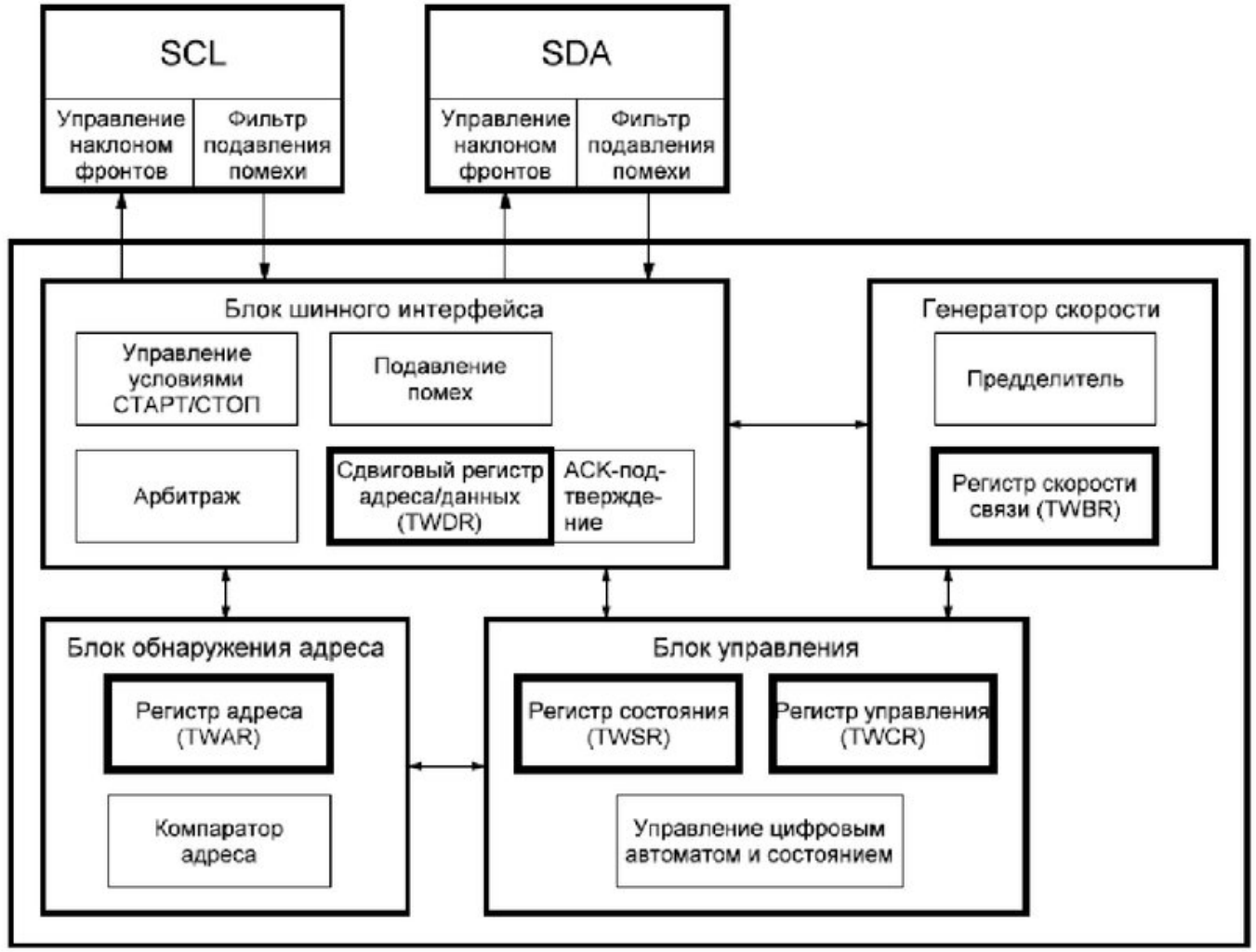


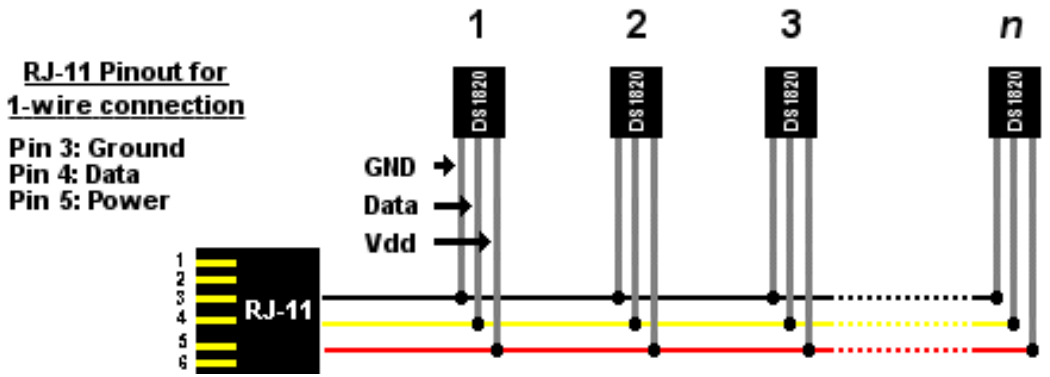

Рисунок 32 – Комбинированный формат. MASTER-устройство передает данные двум SLAVE-абонентам с 10-разрядным адресом


Рисунок 33 – Комбинированный формат. MASTER-абонент передает данные двум SLAVE-устройствам: одному с 7-разрядиым адресом, а другому с 10-разрядным адресом

## Функциональная схема блока шины I²C (TWI) микроконтроллера AVR



# Интерфейсная шина 1-Wire



# Интерфейсная шина 1-Wire

**1-Wire (англ. один провод)** — двунаправленная шина связи для устройств с низкоскоростной передачей данных (обычно 15,4 Кбит/с, максимум 125 Кбит/с в режиме overdrive), в которой данные передаются по цепи питания (то есть всего используются два провода — один для заземления, а второй для питания и данных; в некоторых случаях используют и отдельный провод питания).

1-Wire разработан в 90-х корпорацией Dallas Semiconductor (с 2001 года — Maxim Integrated) и является её зарегистрированной торговой маркой.

1-Wire обычно используется для того, чтобы связываться с недорогими простыми устройствами, такими, как, например, цифровые термометры и измерители параметров внешней среды.

На 1-Wire работает большинство "таблеток" — домофонных чипов (DS1990A), карточек доступа, а также через 1-Wire общаются популярные датчики температуры (DS18S20 и DS18B20), транзисторные ключи (DS2405, DS2406), программируемые порты ввода-вывода (DS2408), АЦП и ЦАП, часы реального времени (DS2417) и многое другое.

# Интерфейсная шина 1-Wire

***Интерфейс 1-Wire***, регламентирован разработчиками для применения в трех основных сферах-приложениях:
• приборы в специальных корпусах MicroCAN для решения проблем идентификации, переноса или преобразования информации (технология iButton),
• программирование встроенной памяти интегральных компонентов,
• системы автоматизации (технология сетей 1-Wire-сетей).

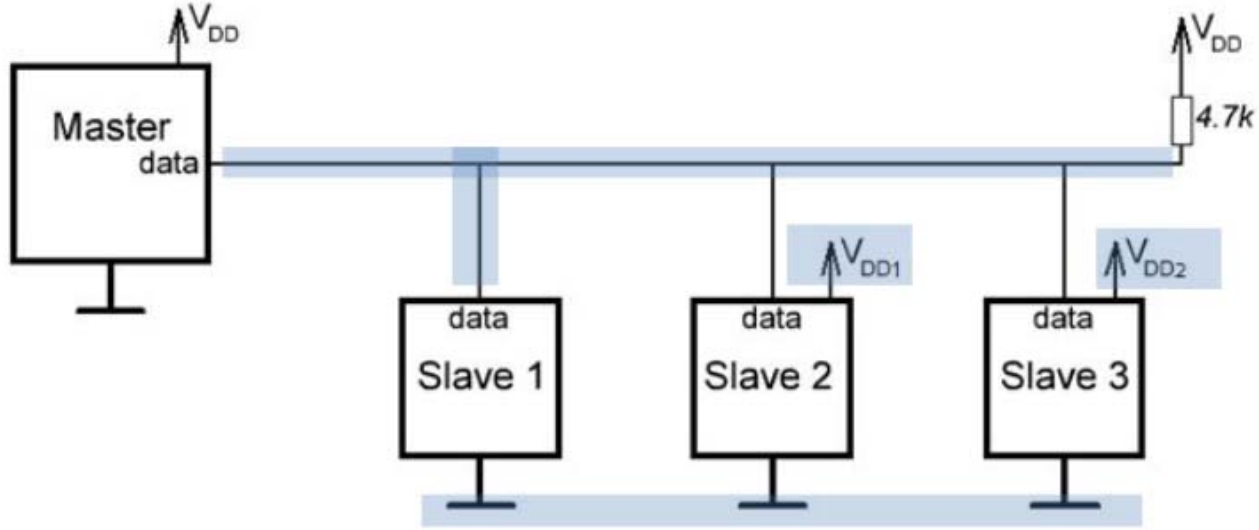
*Преимущества 1-Wire:*
• простое и оригинальное решение адресуемости абонентов;
• несложный протокол;
• простая структура линии связи;
• малое потребление компонентов;
• легкое изменение конфигурации сети;
• значительная протяженность линий связи;
• исключительная дешевизна всей технологии в целом.

Однопроводные компоненты 1-Wire являются самотактируемыми полупроводниковыми устройствами, в основе обмена информацией между которыми, лежит управление изменением длительности временных интервалов импульсных сигналов в однопроводной среде и их измерение.

Вся информация, циркулирующая в сети, воспринимается абонентами либо как команды, либо как данные. Команды сети генерируются мастером и обеспечивают различные варианты поиска и адресации ведомых устройств, определяют активность на линии даже без непосредственной адресации отдельных компонентов, управляют обменом данными в сети.

# Интерфейсная шина 1-Wire

*Топология сети — общая шина*. Сеть устройств 1-Wire со связанным основным стройством названа «MicroLan», это также торговая марка Dallas Semiconductor.
Режим связи в этом интерфейсе – асинхронный и полудуплексный.
При отсылке многобайтовых данных передача идёт от младшего байта к старшему.




Для организации интерфейса необходимы как минимум линия данных и «общий провод».
Может быть вариант с выделенной линией питания (Slave 2, Slave 3, например, 3В и 5 В).
Однако некоторые ведомые устройства могут питаться и «паразитно» – получать «подпитку» через шину данных (Slave 1).
Ограничение максимальной длины однопроводной линии регламентировано разработчиками на уровне 300 м.

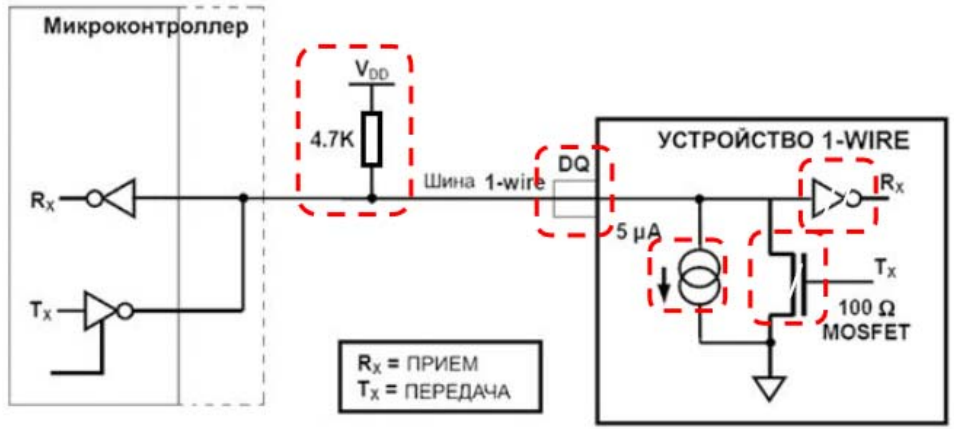



Рисунок 2 - Упрощенная схема аппаратной реализации интерфейса 1-Wire

Упрощенная схема аппаратной реализации интерфейса 1-Wire показана на рисунке 2.

1. Вывод DQ устройства представляет собой вход КМОП-логического элемента, который может быть зашунтирован (замкнут на общий провод) полевым транзистором. Сопротивление канала этого транзистора в открытом состоянии – около 100 Ом. Когда транзистор заперт – имеется небольшой ток утечки (примерно 5 мкА) на общий провод.
2. Шина 1-Wire должна быть подтянута отдельным резистором к напряжению питания устройств (от 3 до 5В – уточняется по характеристикам конкретного устройства). Сопротивление этого резистора 4.7 кОм, однако, это значение рекомендовано только для достаточно коротких линий.
3. Если шина 1-Wire используется для подключения удаленных на большое расстояние устройств, то сопротивление этого резистора следует уменьшить.
4. Минимально допустимое сопротивление – около 300 Ом, а максимальное – около 20-30 кОм. Данные величины – ориентировочные и всегда должны быть уточнены по характеристикам конкретного устройства 1-Wire его максимальный втекающий ток линии DO, который и определяет минимум внешнего сопротивления.

## Канальный уровень

Обмен информацией ведётся так называемыми *тайм-слотами* длительностью 60 мкс.
Один тайм-слот служит для обмена одним битом информации.
*Данные передаются бит за битом*, начиная с *младшего бита младшего байта*.
При передаче по 1-Wire, например, двухбайтового числа порядок передачи будет таким:
Например, десятичное число $10\,234_{10}$ – в двоичном виде выглядит так: $0010\ 0111\ 1111\ 1010_2$
В памяти число размещено так: 1111 1010 0010 0111.
Передача по 1-Wire будет выглядеть так: 0→1→0→1→1→1→1→1→1→1→1→0→0→1→0→0
При обмене информацией ведущий инициирует каждую связь на битовом уровне. Это означает, что *передача каждого бита*, независимо от направления (передача или приём), *должна быть инициирована ведущим*.
Шина данных по умолчанию подтягивается к "единице", поэтому для начала как приёма, так и для передачи ведущий опускает линию в "ноль" на некоторое время.
**Внимание**: ни ведущий, ни ведомые не выставляют на шине "единицу" – это черевато коротким замыканием если одно устройство выставит на шине "1", а другое – "0".
Поэтому как ведущий, так и ведомый могут использовать только два состояния:

- "выход в ноль" и
- "z-состояние" (на вход без подтяжки).

Подтяжка к питанию осуществляется резистором.

В 1-Wire есть *5 основных команд* для связи по шине :

- "Запись 1";
- "Запись 0";
- "Чтение";
- "Сброс" ;
- "Присутствие".

На рисунках выделено: **красным** – управление линией от ведущего, синим – управление линией от ведомого, **черным** – освобожденная линия (с помощью подтяжки шина автоматически переходит в "единицу").

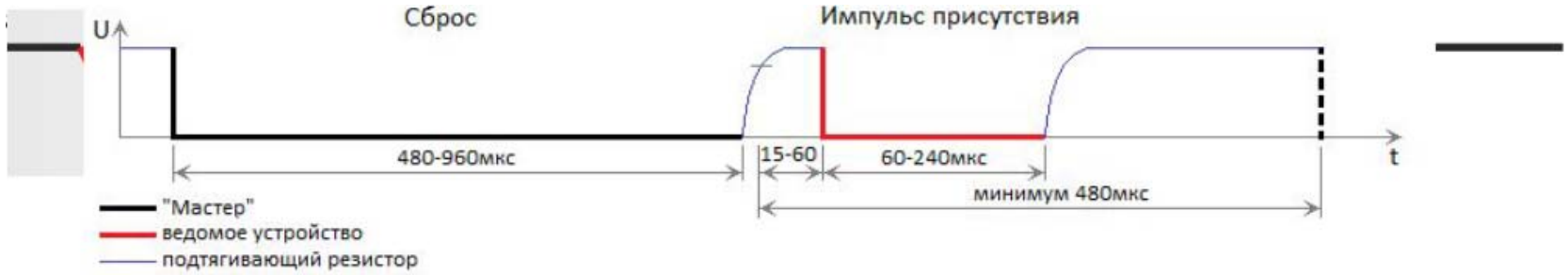
**Сигнал "Запись 0".** Ведущий формирует низкий уровень в течение не менее 60 мкс, но не дольше 120 мкс.


**Сигнал "Запись 1".** Ведущий устанавливает низкий уровень в течение 1…15 мкс. После этого, в течение оставшейся части временного слота он освобождает шину.

**Сигнал "Чтение".** Ведущий устанавливает низкий уровень в течение 1…15 мкс. После этого подчинённый, если хочет передать 0, удерживает шину в низком состоянии до 60 мкс; если же подчинённый хочет передать 1, то он просто освобождает линию. Ведущий обычно сканирует состояние шины по истечении 15 мкс после установки низкого уровня на шине.


Так, ведомый удерживает линию к земле, если хочет передать "0", и просто отпускает линию, если хочет передать "1". Таким образом при чтении получаем следующие диаграммы.

**Сигнал "Сброс/присутствие".** Для этого типа сигнала временные интервалы импульсов отличаются.
Ведущий устанавливает низкий уровень в течение 8 временных слотов (480 мкс), а затем освобождает шину.
Данный длительный период низкого состояния называется *сигнал "Сброс".*
Если на шине присутствует подчинённый, то он должен в течение 60 мкс после освобождения ведущим шины установить низкий уровень длительностью не менее 60 мкс.
Данный отклик носит название сигнал "Присутствие".
Сброс Присутствие




Если такой сигнал не обнаруживается, то ведущий должен полагать, что нет подключённых устройств к шине и дальнейшая связь невозможна.
*Данная связка сигналов всегда начинает любой обмен информацией между устройствами.*
Любое ведомое устройство после получения питания сразу же выдаёт сигнал присутствия.
Сигнал "Сброс" позволяет ведущему досрочно завершить обмен информацией – например, если датчик температуры передаёт всю свою память, а нужны, например, только первые два байта, которые собственно содержат значение температуры, то после получения этих двух байт микросхема просто может опустить линию в ноль на нужное количество времени – датчик поймет, что больше ничего пересылать не нужно.

# Интерфейсная шина 1-Wire

Весь обмен на шине 1-Wire происходит посредством специальных команд. Их число для каждого типа устройств различно. Есть минимальный набор стандартных команд, которые поддерживают все 1-Wire-устройства – так называемые ROM-команды.

***Алгоритм взаимодействия:***

1. Ведущий посылает на линию сигнал "Сброс". Затем линия освобождается для отклика ведомых. Если на шине присутствует ведомый, то в течение 60 мкс он сообщает о своём "присутствии".
Если же ведущий не получает отклика – "присутствия", то он считает, что подключённых к шине устройств нет.
2. Сетевой уровень: ведущий определяет устройство на шине данных к которому он будет дальше обращаться. Данный выбор обеспечивается отсылкой одной из ROM-команд (длиной в 1 байт), которые работают с уникальными кодами устройств:

- **Read ROM (0x33)** – Чтение адреса устройства. Используется для определения адреса единственного устройства на шине, если точно известно, что есть только одно подчинённое устройство (например, только один датчик температуры или один домофонный ключ), тогда для считывания его кода не нужно выполнять команду поиск других ведомых. При получении данной команды все ведомые устройства на шине отсылают свой уникальный код.
- **Skip ROM (0xCC)** – Игнорировать адрес. Используется для обращения к единственному устройству на шине, при этом адрес устройства игнорируется (можно обращаться к неизвестному устройству или дать команду всем устройствам на шине (широковещательная рассылка) – например, нужно, чтобы все подключённые датчики одновременно считали температуру.
- **Match ROM (0x55)** – Выбор адреса. Используется для обращения к конкретному адресу устройства из многих подключенных. После отсылки команды ведущий передаёт 64-разрядный код адреса ведомого устройства. По завершении передачи разрешается отвечать только тому подчинённому устройству (после приёма следующего импульса...

• **Search ROM (0xF0)** – Поиск адресов. Используется при универсальном алгоритме определения количества подключенных устройств и их адресов. Эта команда используется когда *адресные коды подключённых устройств* не известны.

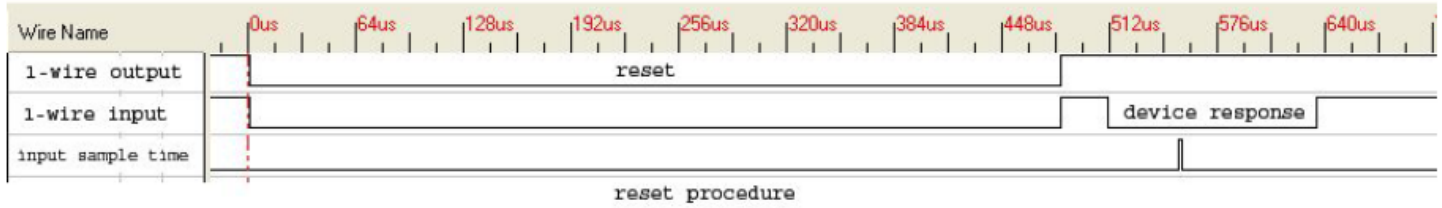
После формирования Ведущим устройством (Мастер) команды "Search ROM" все ведомые устройства посылают на шину значение своего младшего бита; при этом сначала в течение одного такта ведомыми посылается "прямое" значение бита, а в течение следующего такта – "обратное" (логическое НЕ, дополнение) значение этого же бита. Тогда Мастер может считать следующие последовательности:

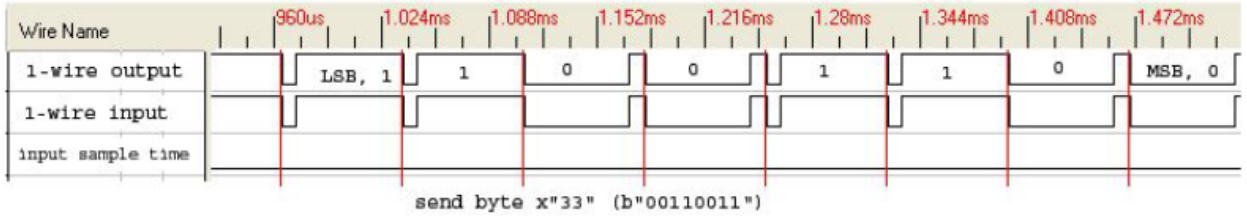
• **"01"** – если у всех ведомых устройств младший бит равен "0" (сначала они отсылают "0", потом (НЕ "0“) = "1")
• **"10"** – если у всех ведомых устройств младший бит равен "1"
• **"00"** – если есть так называемый конфликт – у некоторых устройств младший бит равен "1", у остальных – "0". Это когда при передаче "прямого" значения устройства с "0" опускают линию, при передаче "обратного" – устройства с "1" также опускают линию. Далее Мастер в следующем временном слоте отсылает "0" или "1", таким образом определяя, с какими устройствами дальше будет общаться – все устройства, у которых младший бит не соответствует биту, сформированному мастером на этом этапе, перейдут в состояние ожидания и будут находиться в нём, пока не получат импульс сброса.

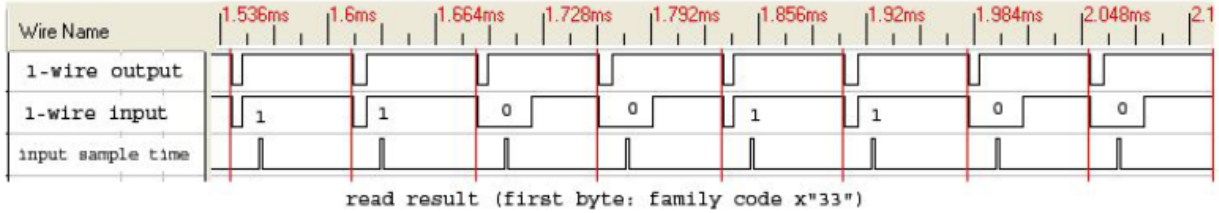
Затем происходят аналогичные 63 цикла чтения-выбора следующих бит адреса, пока, наконец, ведущее устройство не определит код ROM одного из подчинённых устройств и не обратится к нему без его специальной адресации командой Macth ROM (0x55).

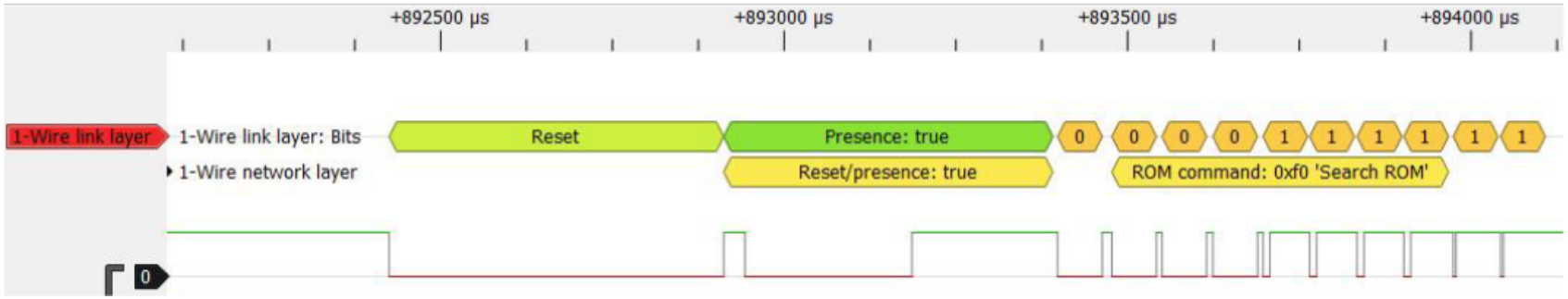
Когда закончен поиск одного идентификатора, то все подчиненные устройства, кроме одного, должны находиться в состоянии ожидания (idle). Для Мастера каждая *стадия выбора состоит из двух тайм-слотов чтения и одного тайм-слота записи*. Стандартная скорость отработки команды составляет:

1 Wire reset, write and read example with DS2432



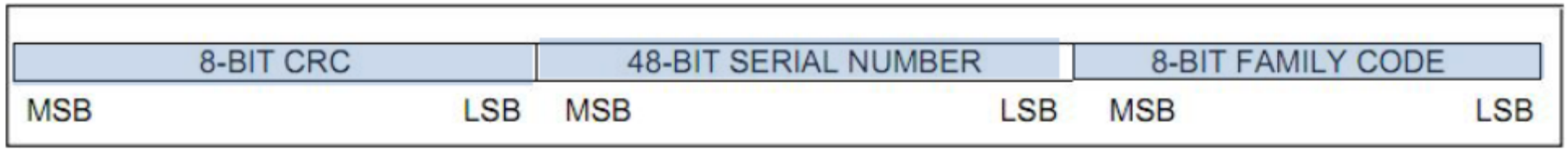



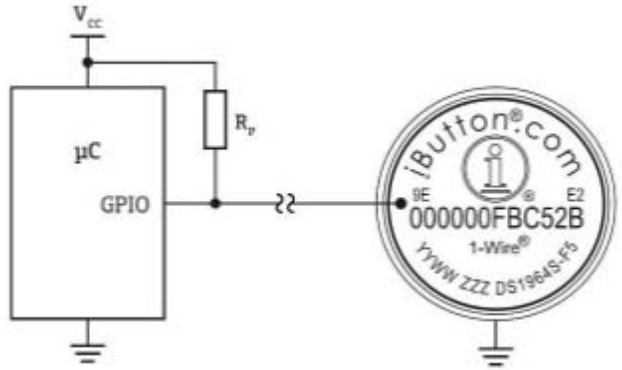



## Уникальные коды устройств 1-Wire

Каждое 1-Wire устройство имеет свой 64-х битный код, состоящий из трёх частей:




- Младший байт – это код семейства, к которому относится устройство,
- 6 следующих байт – уникальный в семействе серийный номер.
- Старший байт – это контрольная сумма (CRC код), который служит для проверки правильности приёма всего кода. Фирма Maxim гарантирует, что один раз использованный адрес никогда не повторится в другом устройстве. В самом деле, 48 бит – это 2,81•1014 различных чисел. Если производить 1000 миллиардов (1012 ) различных устройств ежегодно, то все серийные номера можно использовать не ранее чем через 281 год – и это только для одного семейства. Так, например, на родных даллосовских (сейчас максимовских) "таблетках" часть уникального кода – а именно, 48-битный серийный номер – пишется на металле в шестнадцатиричном виде.

**Транспортный уровень** – если были отправлены команды "Match ROM" (0x55) или "Skip ROM" (0xCC), то далее ведущий отправляет какую-либо функциональную команду. Набор функциональных команд и дальнейшее поведение (например, должен ли ведущий быть готов принимать данные от выбранного ведомого устройства) зависит от конкретного 1-Wire устройства.

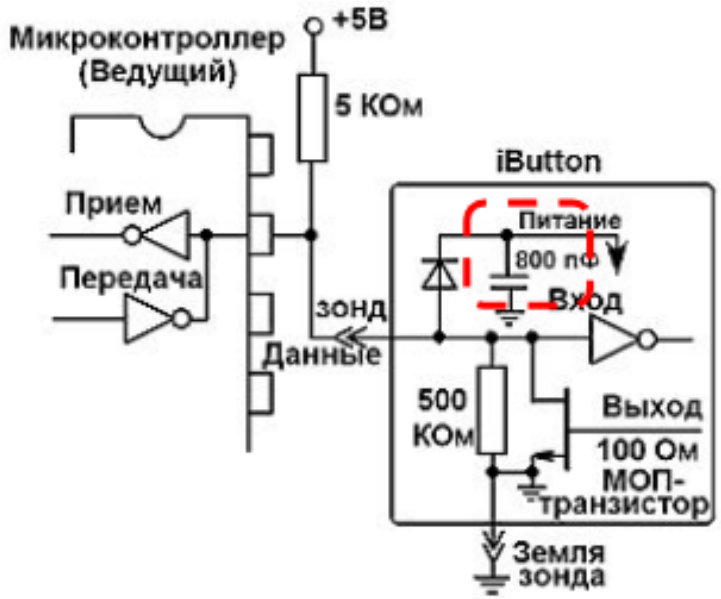
Например, если надо микроконтроллеру (МК) получить информацию от датчика температуры, например, DS18S20.

Алгоритм работы будет следующим:

- МК отсылает импульс «Сброс».
- Датчик отвечает импульсом «Присутствие».
- МК отсылает адресную команду "Skip ROM" – так как заранее известно, что датчик на линии один, то нет необходимости работать с "адресами" (идентификаторами).
- МК отсылает функциональную команду "Convert T" – по этой команде датчик температуры начнёт однократное температурное преобразование. Результат этого преобразования будет записан в память датчика.
- МК ждёт, пока датчик закончит преобразование (ведомое устройство никоим образом не может само сообщить, что оно "освободилось", поэтому микроконтроллер просто ждёт время, указанное в техдокументации).
- МК отсылает импульс «Сброс».
- Датчик отвечает импульсом «Присутствие».
- МК снова отсылает адресную команду «Skip ROM».
- МК отсылает функциональную команду "Read Scratchpad" – по этой команде датчик отсылает 9 байт своей памяти.
- МК считывает нужное количество байт (значение температуры содержится в первых двух)
- При необходимости МК завершает (прерывает) сеанс связи, отсылая импульс «Сброс»

## «Паразитное» питание

Альтернативой применению внешнего питания служит, так называемый, механизм "паразитного питания", действие которого заключается в использовании каждым из ведомых компонентов 1-Wire-линии электрической энергии импульсов, передаваемых по шине данных, которая аккумулируется специальной, встроенной в прибор ёмкостью. Кроме того, отдельные компоненты однопроводных сетей могут использовать режим питания по шине данных, когда энергия к приемнику поступает непосредственно от Мастера по линии связи, при этом обмен информацией в сети принудительно прекращается.




Каждое устройство имеет свои требования к «паразитному» питанию, но обычно это требование не опускать линию данных в ноль в течение выполнения ведомым определённых функциональных команд. Например, для датчика температуры DS18S20, использующего протокол 1-Wire, необходимо гарантировать, что на шине данных будет достаточный уровень напряжения при выполнении температурного преобразования или при копировании данных из памяти EEPROM. Согласно технической документации, при этом рекомендуется подтягивать линию данных к питанию с помощью полевого транзистора; на деле же при использовании напряжения +5-+5.5 В микроконтроллеру достаточно просто не опускать шину данных в ноль (в случае именно с датчиком DS18S20 – другие датчики могут потреблять больше тока). Также именно «паразитное» питание является одной из причин, почему микроконтроллеру стоит передавать сигналы (то есть опускать линию данных в ноль) по минимальному времени – это позволяет устройствам на «паразитном» питании нормально функционировать

## Контрольная сумма

Для того, чтобы гарантировать целостность данных устройствами 1-Wire используется специальная проверка контрольной суммы основанная на циклическом избыточном коде CRC – (Cyclic Redundancy Check ).
В устройствах 1-Wire можно найти два разных варианта CRC.

- CRC8 – 8-битный (называется Dallas One Wire CRC или DOW-CRC),
- CRC16 – 16-битный.

CRC8 используется в секции ROM всех устройств. Также CRC8 используется в некоторых устройствах для проверки других данных, наподобие команд, выданных на шине.
CRC16 используется в некоторых устройствах для проверки на ошибки в достаточно больших наборах данных.
Аппаратный эквивалент 8-битной CRC, используемой на 64-битном идентификаторе, показан на рисунке 3.
Блоками представлены отдельные биты регистра сдвига.
Эквивалентный полином для этой CRC будет X8 + X5 + X4 + 1.
Контрольная сумма

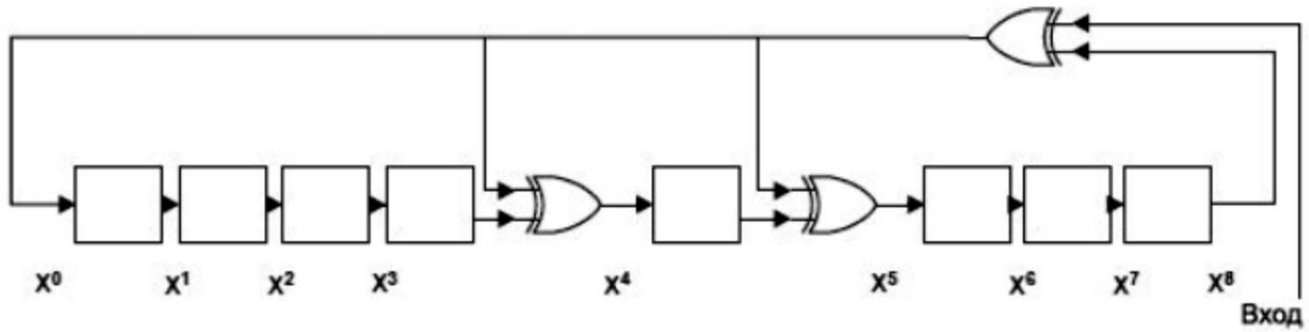
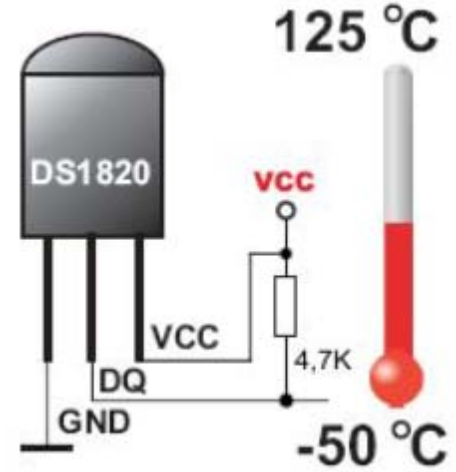



Рисунок 3 - Аппаратный эквивалент CRC8, используемой в 1-Wire устройствах

## Цифровой термометр с однопроводным интерфейсом

Микросхема DS18B20 представляет собой температурный датчик в корпусе TO92. Первый вывод (GND) подключается к шине RETURN, второй (DQ) DATA, а третий используется для подачи внешнего питания. Внешнее питание может быть организовано от отдельного источника +5 V.
DS18B20 – высокоточный цифровой термометр с однопроводным интерфейсом в стандарте MicroLAN. Диапазон измеряемых температур от -55° C до +125° C.
Считываемый с прибора цифровой код является прямым непосредственным кодом измеренного значения температуры и не нуждается в дополнительных преобразованиях.




DS18B20 имеет встроенный АЦП имеющий 12 разрядов выходного кода.
Абсолютная погрешность преобразования меньше 0,5° C в диапазоне контролируемых температур – от +10° C до +85° C.
Максимальное время полного 12-ти разрядного преобразования ~750мс.
Внутренняя энергонезависимая память EEPROM обеспечивает запись произвольных значений верхней и нижней границы температурных уставок. Кроме того, микросхема содержит встроенный логический механизм приоритетной сигнализации в линию о факте выхода температуры за один из выбранных порогов.
Термометр имеет индивидуальный 64-разрядный регистрационный номер (групповой код 0x28) и обеспечивает возможность работы без внешнего источника питания, только за счет «паразитного» питания однопроводной линии. Питание прибора через отдельный внешний вывод производится напряжением от 3,0 В до 5,5 В.

## Опрос температурных датчиков DS18B20, DS18S20, DS1822

Получение значений температуры с датчиков выполняется в два этапа:
1) Запрос на измерение температуры (команда 0x44). Для устройств с паразитным питанием в течение 10 мкс после этого должен быть включен относительно низкоомный подтягивающий резистор на всё время замера (до 750 мс).
2) Запрос на считывание измеренных значений (команда 0xBE).
Любому запросу к устройству обязательно предшествует сигнал «Сброс» и одна из команд «выбора устройства», в соответствии с сетевым протоколом 1-wire.
Результаты измерений сохраняются в устройстве в буферной энергонезависимой памяти размером 9 байт, называемой скратчпад (scratchpad – электронный блокнот).
У всех трёх рассматриваемых моделей термодатчиков структура скратчпада схожа между собой:

| Позиция | Значение |
|---|---|
| 0 | младший байт значения температуры |
| 1 | старший байт значения температуры |
| 2,3 | граничные контрольные значения, или пользовательские данные |
| 4 | регистр конфигурации (DS18B20, DS1822) / зарезервирован (принимает значение 0xFF для DS18S20) |
| 5 | зарезервирован (принимает значение 0xFF) |
| 6 | зарезервирован (DS18B20, DS1822)/остаток COUNT_REMAIN (DS18S20) |
| 7 | зарезервирован (принимает значение 0x10 для DS18B20, DS1822) /множитель = отсчётов на градус (всегда 0x10 = 16 для DS18S20) |
| 8 | контрольная сумма CRC, рассчитывается для всего скратчпада по тем же правилам, что и для адреса. |

Измеренное значение температуры сохраняется в виде знакового 16-и битного целого в первых двух байтах скратчпада, с начала младший байт.
Датчики DS18B20 и DS1822 имеют программируемое разрешение от 9 до 12 бит, и сохраняют результат всегда в 1/16 градусах Цельсия. Таким образом, значения

- 0x50 0x05 – соответствуют +85°C,
- 0xA2 0x00 – соответствуют +10,125°C,
- 0x6F 0xFE – соответствуют –25,0625°C.

По умолчанию разрешение АЦП задано 12 бит. При выборе меньшего разрешения сокращается время замера температуры, при этом недостающее число младших разрядов результата будет содержать неопределённые данные.
В отличие от них, DS18S20 сохраняет результат с разрешением в половину градуса. То есть

- 0xAA 0x00 – соответствует +85°C,
- 0x32 0x00 – соответствует +25°C,
- 0xCE 0xFF – соответствует –25°C.

Однако, можно вычислить значение с точностью до 1/16 градуса, используя значение COUNT_REMAIN, сохраняемое в ячейке 6 скратчпада по следующей формуле:

**ТЕМПЕРАТУРА = ПРОЧИТАННАЯ ТЕМПЕРАТУРА – 0,25°C + COUNT_REMAIN / 16,**

где 16 – это количество отсчётов на градус, значение, возвращаемое в ячейку 7 скратчпада и всегда равное 16 для DS18S20.

Рисунок 4 - Схема подключения нескольких датчиков DS18B20 с внешним питанием.

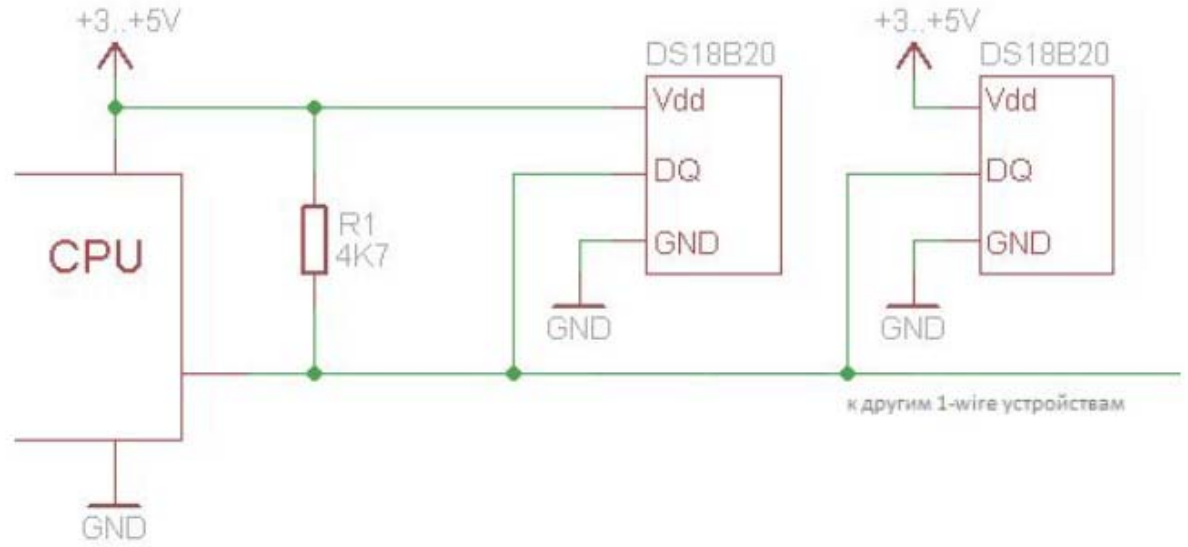
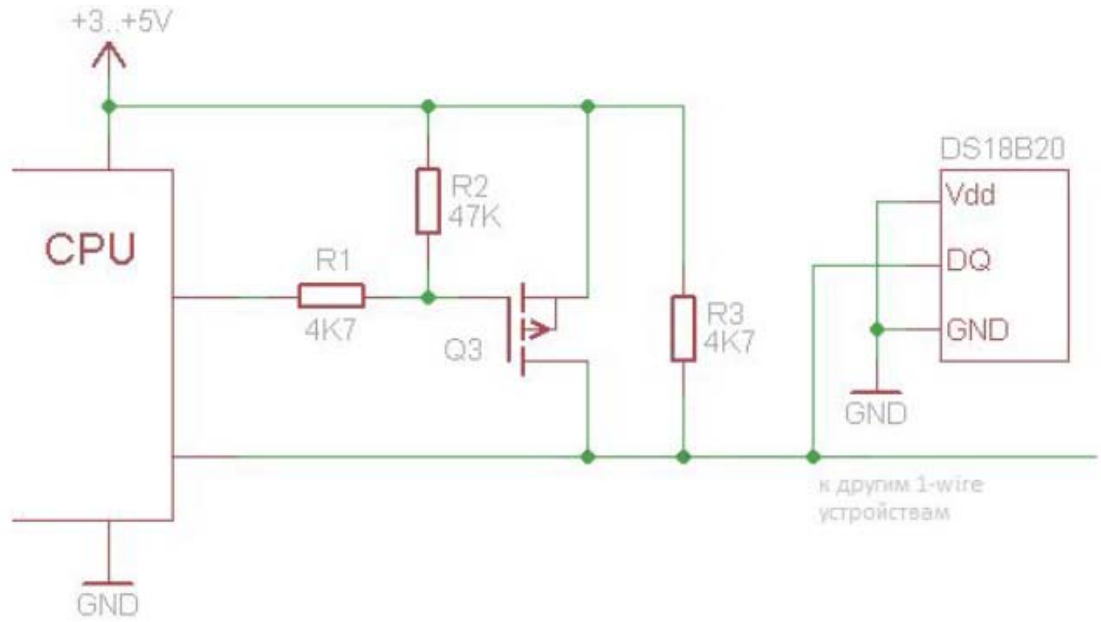



Рисунок 5 - Схема подключения датчика DS18B20 в режиме «паразитного» питания



# Спасибо за внимание

https://theslide.ru/uncategorized/interfeys-1-wire

ЧУ ПО «Социально-технологический колледж»

Преподаватель: Борисов Алексей Альбертович