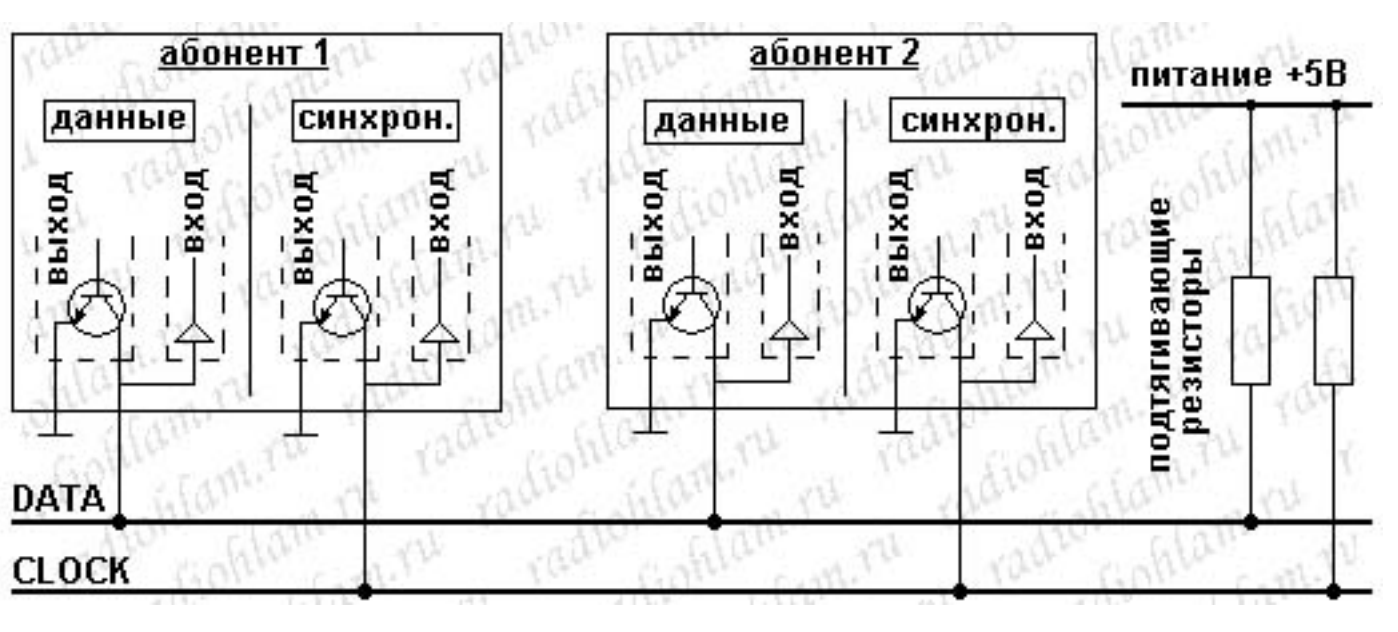
Подробное описание интерфейса I2C

26.10.2010 Интерфейсы, Теория Комментарии: 0 rhf-admin Метки: I2C, интерфейс

Интерфейс I2C (или по другому IIC) — это достаточно широко распространённый сетевой последовательный интерфейс, придуманный фирмой Philips и завоевавший популярность относительно высокой скоростью передачи данных (обычно до 100 кбит/с, в современных микросхемах до 400 кбит/с), дешевизной и простотой реализации.

1) *Физика*.



Физически сеть представляет собой двухпроводную шину, линии которой называются **DATA** и **CLOCK** (необходим ещё и третий провод — земля, но интерфейс принято называть двухпроводным по количеству сигнальных проводов). Соответственно, по линии DATA передаются данные, линия CLOCK служит для тактирования. К шине может быть подключено до 128 абонентов, каждый со своим уникальным номером. В каждый момент времени информация передаётся только одним абонентом и только в одну сторону.

Устройства I2C имеют выход с "открытым коллектором". Когда выходной транзистор закрыт — на соответствующей линии через внешний подтягивающий резистор устанавливается высокий уровень, когда выходной транзистор открыт — он притягивает соответствующую линию к земле и на ней устанавливается низкий уровень (смотрите рисунок). Резисторы имеют номинал от нескольких килоОм до нескольких десятков килоОм (чем выше скорость — тем меньше номинал резисторов, но больше энергопотребление). На рисунке треугольниками на входе показано, что входы высокоомные и, соответственно, влияния на уровни сигналов на линиях они не оказывают, а только "считывают" эти уровни. Обычно используются уровни 5В или 3,3В.

2) *Логика*.

Любое устройство на шине I2C может быть одного из двух типов: Master (ведущий) или Slave (ведомый). Обмен данными происходит сеансами. "Мастер"-устройство полностью управляет сеансом: инициирует сеанс обмена данными, управляет передачей, подавая тактовые импульсы на линию Clock, и завершает сеанс.

Кроме этого, в зависимости от направления передачи данных и "Мастер" и "Слэйв"-устройства могут быть "Приёмниками" или "Передатчиками". Когда "Мастер" принимает данные от "Слэйва" — он является "Приёмником", а "Слэйв" — "Передатчиком". Когда же "Слэйв" принимает данные от "Мастера", то он уже является "Приёмником", а "Мастер" в этом случае является "Передатчиком".

Не надо путать тип устройства "Мастер" со статусом "Передатчика". Несмотря на то, что при чтении "Мастером" информации из "Слэйва", последний выставляет данные на шину Data, делает он это только тогда, когда "Мастер" ему это разрешит, установкой соответствующего уровня на линии Clock. Так что, хотя "Слэйв" в этом случае и управляет шиной Data, — самим обменом всё равно управляет "Мастер".

В режиме ожидания (когда не идёт сеанс обмена данными) обе сигнальные линии (Data и Clock) находятся в состоянии высокого уровня (притянуты к питанию).

Каждый сеанс обмена начинается с подачи "Мастером" так называемого Start-условия. "Старт-условие" — это изменение уровня на линии Data с высокого на низкий при наличии высокого уровня на линии Clock.

После подачи "Старт-условия" первым делом "Мастер" должен сказать с кем он хочет пообщаться и указать, что именно он хочет — передавать данные в устройство или читать их из него. Для этого он выдаёт на шину 7-ми битный адрес "Слэйв" устройства (по другому говорят: "адресует "Слэйв" устройство"), с которым хочет общаться, и один бит, указывающий направление передачи данных (0 — если от "Мастера" к "Слэйву" и 1 — если от "Слэйва" к "Мастеру"). Первый байт после подачи "Старт"-условия всегда всеми "Слэйвами" воспринимается как адресация.

Поскольку направление передачи данных указывается при открытии сеанса вместе с адресацией устройства, то для того, чтобы изменить это направление, необходимо открывать ещё один сеанс (снова подавать "Старт"-условие, адресовать это же устройство и указывать новое направление передачи).

После того, как "Мастер" скажет, к кому именно он обращается и укажет направление передачи данных, — начинается собственно передача: "Мастер" выдаёт на шину данные для "Слэйва" или получает их от него. Эта часть обмена (какие именно данные и в каком порядке "Мастер" должен выдавать на шину, чтобы устройство его поняло и сделало то, что ему нужно) уже определяется каждым конкретным устройством.

Заканчивается каждый сеанс обмена подачей "Мастером" так называемого Stop-условия, которое заключается в изменении уровня на линии Data с низкого на высокий, опять же при наличии высокого уровня на линии Clock. *Если на шине сформировано Stop-условие, то закрываются все открытые сеансы обмена*.

Внутри сеанса любые изменения на линии Data при наличии высокого уровня на линии Clock запрещены, поскольку в это время происходит считывание данных "Приёмником". Если такие изменения произойдут, то они в любом случае будут восприняты либо как "Старт"-условие (что вызовет прекращение обмена данными), либо как "Стоп"-условие (что будет означать окончание текущего сеанса обмена). Соответственно, во время сеанса обмена установка данных "Передатчиком" (выставление нужного уровня на линии Data) может происходить только при низком уровне на линии Clock.

Несколько слов по поводу того, в чём в данном случае разница между "прекращением обмена данными" и "окончанием сеанса обмена". В принципе "Мастеру" разрешается, не закрыв первый сеанс обмена, открыть ещё один или несколько сеансов обмена с этим же (например, как было сказано выше, для изменения направления передачи данных) или даже с другими "Слэйвами", подав новое "Старт"-условие без подачи "Стоп"-условия для закрытия предыдущего сеанса. Управлять линией Data, для того, чтобы отвечать "Мастеру", в этом случае будет разрешено тому устройству, к которому "Мастер" обратился последним, однако старый сеанс при этом нельзя считать законченным. И вот почему. Многие устройства (например те же eeprom-ки 24Cxx) для ускорения работы складывают данные, полученные от "Мастера" в буфер, а разбираться с этими полученными данными начинают только после получения сигнала об окончании сеанса обмена (то есть "Стоп-условия").

То есть, например, если на шине висит 2 микросхемы eeprom 24Cxx и вы открыли сеанс записи в одну микросхему и передали ей данные для записи, а потом, не закрывая этот первый сеанс, открыли новый сеанс для записи в другую микросхему, то реальная запись и в первую и во вторую микросхему произойдёт только после формирования на шине "Стоп-условия", которое закроет оба сеанса. После получения данных от "Мастера" eeprom-ка складывает их во внутренний буфер и ждёт окончания сеанса, для того, чтобы начать собственно процесс записи из своего внутреннего буфера непосредственно в eeprom. То есть, если вы после после передачи данных для записи в первую микруху не закрыли этот сеанс, открыли второй сеанс и отправили данные для записи во вторую микруху, а потом, не сформировав "Стоп-условие", выключили питание, то реально данные не запишутся ни в первую микросхему, ни во вторую. Или, например, если вы пишете данные попеременно в две микрухи, то в принципе вы можете открыть один сеанс для записи в первую, потом другой сеанс для записи во вторую, потом третий сеанс для записи опять в первую и т.д., но если вы не будете закрывать эти сеансы, то в конце концов это приведёт к переполнению внутренних буферов и в итоге к потере данных.

Здесь можно привести такую аналогию: ученики в классе ("слэйвы") и учитель ("мастер"). Допустим учитель вызвал какого-то ученика (пусть будет Вася) к доске и попросил его решить какой-то пример. После того как Вася этот пример решил, учитель вызвал к доске Петю и начал спрашивать у него домашнее задание, но Васю на место не отпустил. Вот в этом случае вроде бы разговор с Васей закончен, — учитель разговаривает с Петей, но Вася стоит у доски и не может спокойно заниматься своими делами (сеанс общения с ним не закрыт).

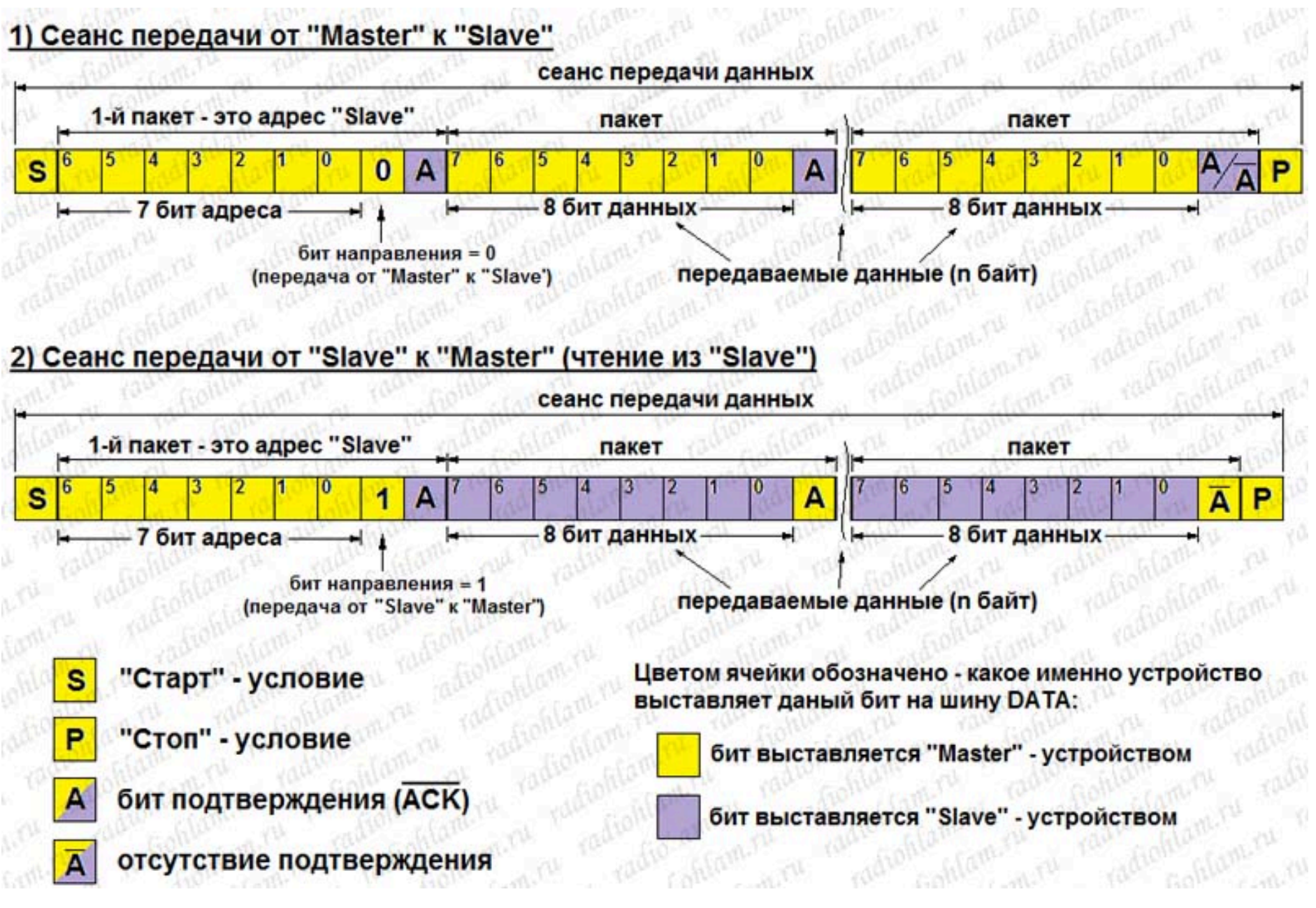
В случае, если "Слэйв" во время сеанса обмена не успевает обрабатывать данные, — он может растягивать процесс обмена, удерживая линию Clock в состоянии низкого уровня, поэтому "Мастер" должен проверять возврат линии Clock к высокому уровню после того, как он её отпустит. *Хотелось бы подчеркнуть, что не стоит путать состояние, когда "Слэйв" не успевает принимать или посылать данные, с состоянием, когда он просто занят обработкой данных, полученных в результате сеанса обмена. В первом случае (во время обмена данными) он может растягивать обмен, удерживая линию Clock, а во втором случае (когда сеанс обмена с ним закончен) он никакие линии трогать не имеет права. В последнем случае он просто не будет отвечать на "обращение" к нему от "Мастера".*

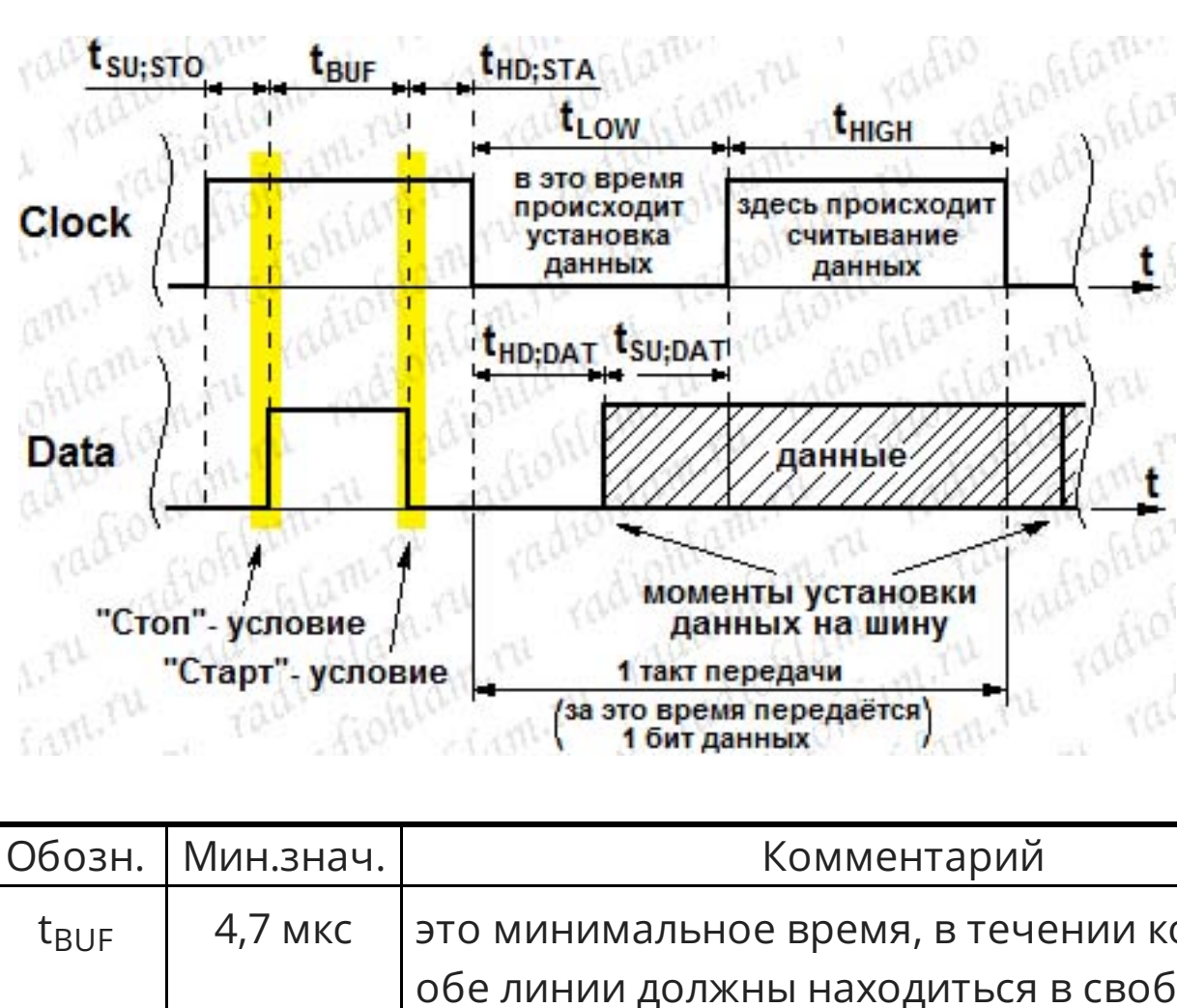
Внутри сеанса передача состоит из пакетов по девять бит, передаваемых в обычной положительной логике (то есть высокий уровень — это 1, а низкий уровень — это 0). Из них 8 бит передаёт "Передатчик" "Приёмнику", а последний девятый бит передаёт "Приёмник" "Передатчику". Биты в пакете передаются старшим битом вперёд. Последний, девятый бит называется битом подтверждения ACK (от английского слова acknowledge — подтверждение). Он передаётся в инвертированном виде, то есть 0 на линии соответствует наличию бита подтверждения, а 1 — его отсутствию. Бит подтверждения может сигнализировать как об отсутствии или занятости устройства (если он не установился при адресации), так и о том, что "Приёмник" хочет закончить передачу или о том, что команда, посланная "Мастером", не выполнена.

Каждый бит передаётся за один такт. Та половина такта, во время которой на линии Clock установлен низкий уровень, используется для установки бита данных на шину передающим абонентом (если предыдущий бит передавал другой абонент, то он в это время должен отпустить шину данных). Та половина такта, во время которой на линии Clock установлен высокий уровень, используется принимающим абонентом для считывания установленного значения бита с шины данных.

Вот собственно и всё. На рисунках ниже всё это описание показано в графической форме.

3) Диаграммы и тайминги.





Параметр	Обозн.	Мин.знач.	Комментарий
Свободная шина	t_{BUF}	4,7 мкс	это минимальное время, в течении которого обе линии должны находиться в свободном состоянии перед подачей "Старт"-условия
Фиксация "Старт"-условия	$t_{HD;STA}$	4,0 мкс	минимальное время от подачи "Старт"-условия до начала первого такта передачи
Готовность "Стоп"-условия	$t_{SU;STO}$	4,0 мкс	минимальное время, через которое можно подавать "Стоп"- условие после освобождения шины Clock
Длительность LOW полупер. шины Clock	t_{LOW}	4,7 мкс	минимальная длительность полупериода установки данных (когда на шине Clock низкий уровень)
Длительность HIGH полупер. шины Clock	t_{HIGH}	4,0 мкс	минимальная длительность полупериода считывания данных (когда на шине Clock высокий уровень)
Удержание данных	$t_{HD;DAT}$	0	то есть данные на шину Data можно выставлять сразу после спада на линии Clock
Готовность данных	$t_{SU;DAT}$	250 нс	то есть поднимать уровень на шине Clock можно не ранее 250 нс после установки данных на шине Data

Минимальные значения времени в таблице указаны для максимальной скорости передачи 100 кбит/с.

Программная реализация мастер-абонента шины I2C в режиме single-master, библиотеки процедур: для PIC, для AVR

Программа для устройства копирования микросхем памяти 24Cxx (здесь можно посмотреть пример использования приведённых выше библиотек для реализации режима I2C-Master на PIC-контроллере)

Программа 2 для контроллера I2C-шлюза, режим Slave из терминалки ПК (а тут посмотреть пример того, как можно сделать I2C-Slave на контроллере AVR)

Понравилась статья? Поделись с друзьями!

Post Views: 11 394

Метки: I2C, интерфейс

Поиск »

- Регистрация
- Войти
- Лента записей
- Лента комментариев

РУБРИКИ

- Контроллеры
  - ARM
  - MIPS
  - PIC
  - AVR
  - Программаторы / средства разработки
- Питание
  - Блоки питания
  - Импульсные преобразователи
  - Линейные стабилизаторы
- Интерфейсы
- Домашняя автоматизация
- Светотехника
- Авто/мото/вело
- Радио
- Технологии
- Маркировка
- Полезные программы для ПК
- Онлайн-калькуляторы
- Словарь терминов, аббревиатур, формул и законов электроники

ОБЛАКО ТЕГОВ

1-Wire ARM avr buck C++ Cortex-M3 dc/dc EEPROM I2C Keil uVision linux MC34063 omega2 OpenWRT pic RS-232 spi step-down step-up stm32 Telegram usb web ИК ИК-приёмник ИК-пульт ОУ боты импульсный преобразователь интерфейс контроллер манчестерский код мотор операционник операционный усилитель понижающий преобразователь преобразователь интерфейсов преобразователь напряжения программатор программирование протокол светодиод светодиодный драйвер фильтр шлюз

Добавить комментарий

Для отправки комментария вам необходимо авторизоваться.