ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Программирование микроконтроллеров.

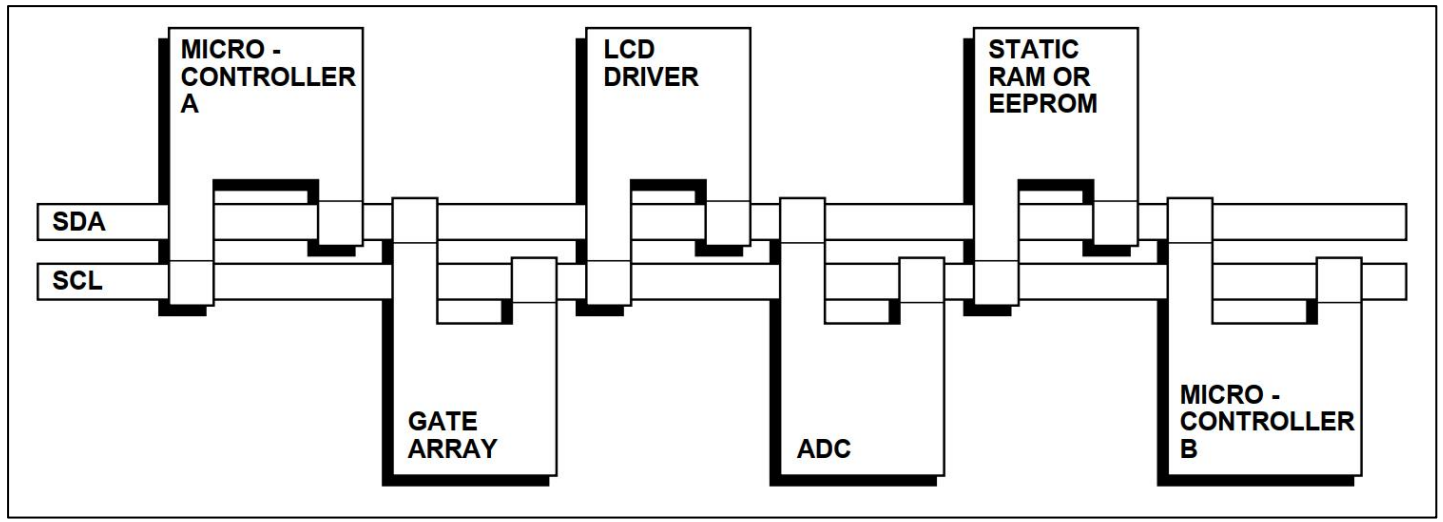
Основные периферийные модули. Теория и практика.

Модуль последовательного интерфейса I2C.

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Обзор интерфейса I2C

I2C = **IIC** = «ай ту си» = Inter-Integrated Circuit



Скорость передачи битов:

Standard Mode (SM) = 100 кбит/сек

Fast Mode (FM) = 400 кбит/сек

Варианты адресации:

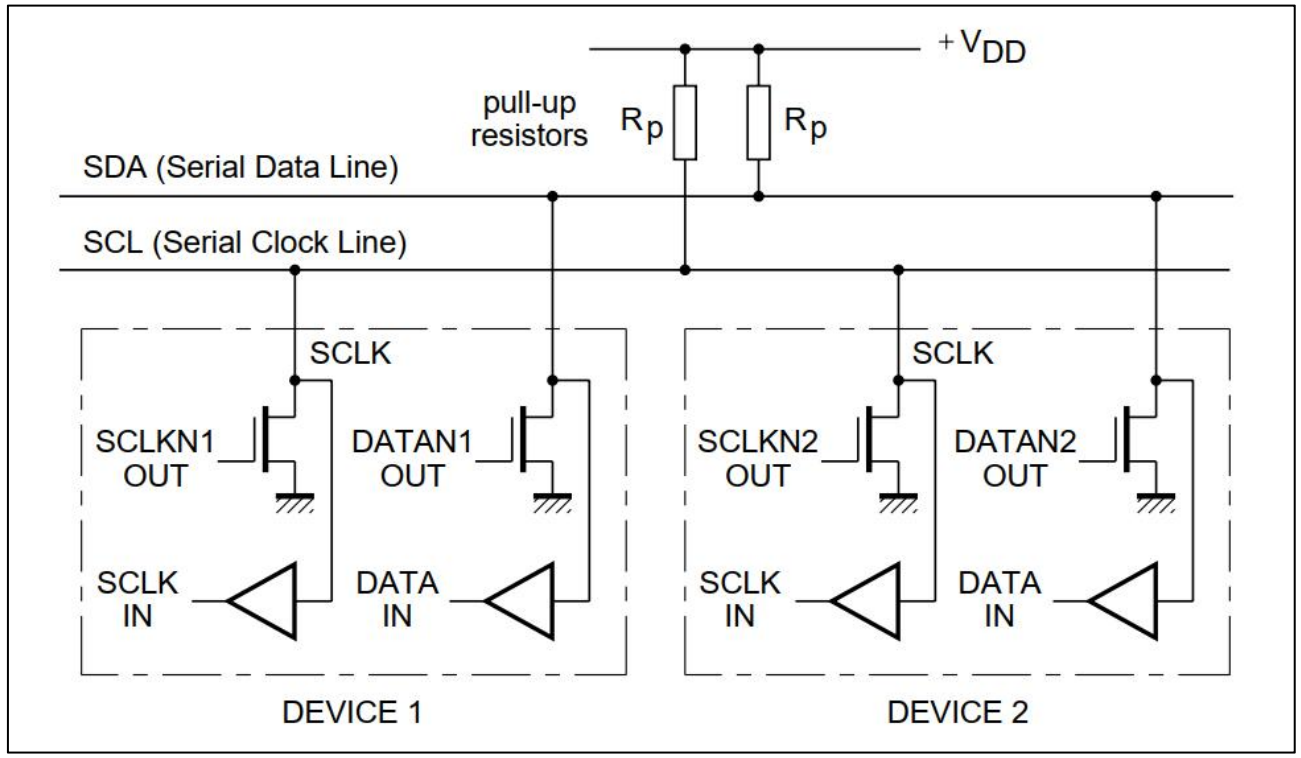
7-битная: до 112 устройств на шине

10-битная: до 1008 устройств на шине

16 адресов зарезервированы

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Аппаратная часть шины I2C

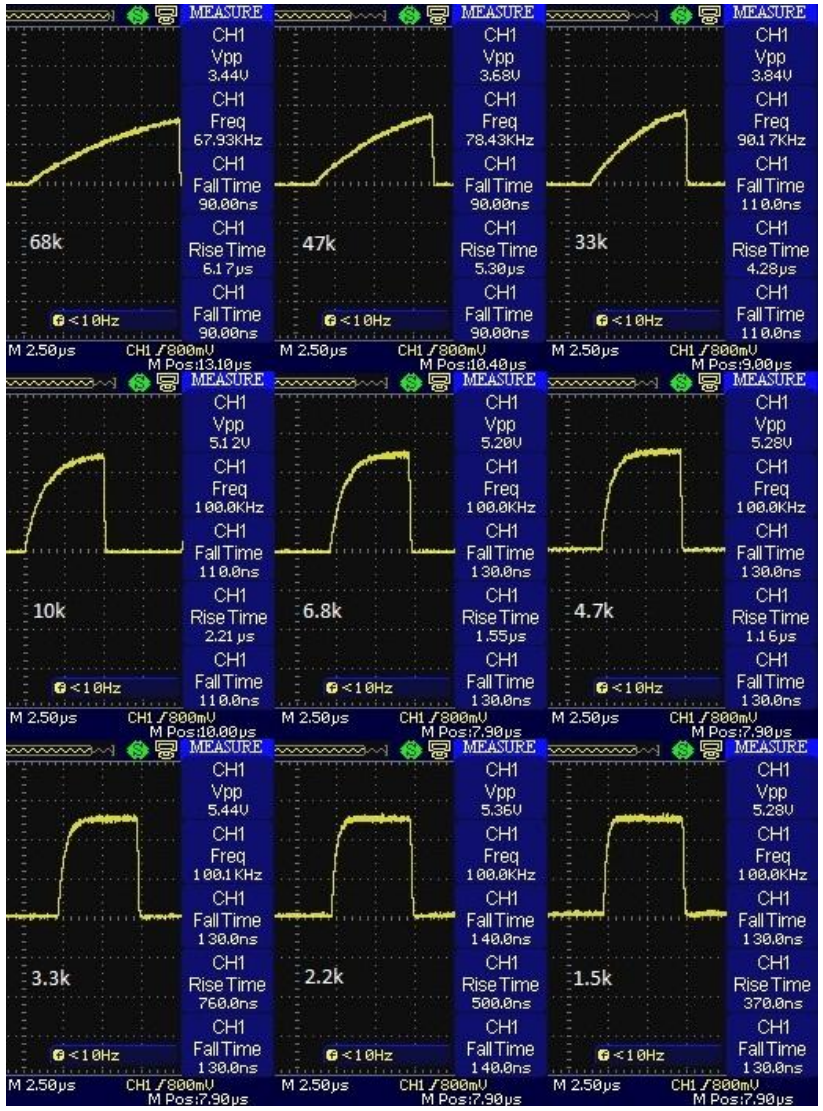


- Vdd = 3,3V – 5V
- максимальное количество устройств на шине ограничивается не только разрядностью адреса, но и максимальной емкостью шины 400 пФ

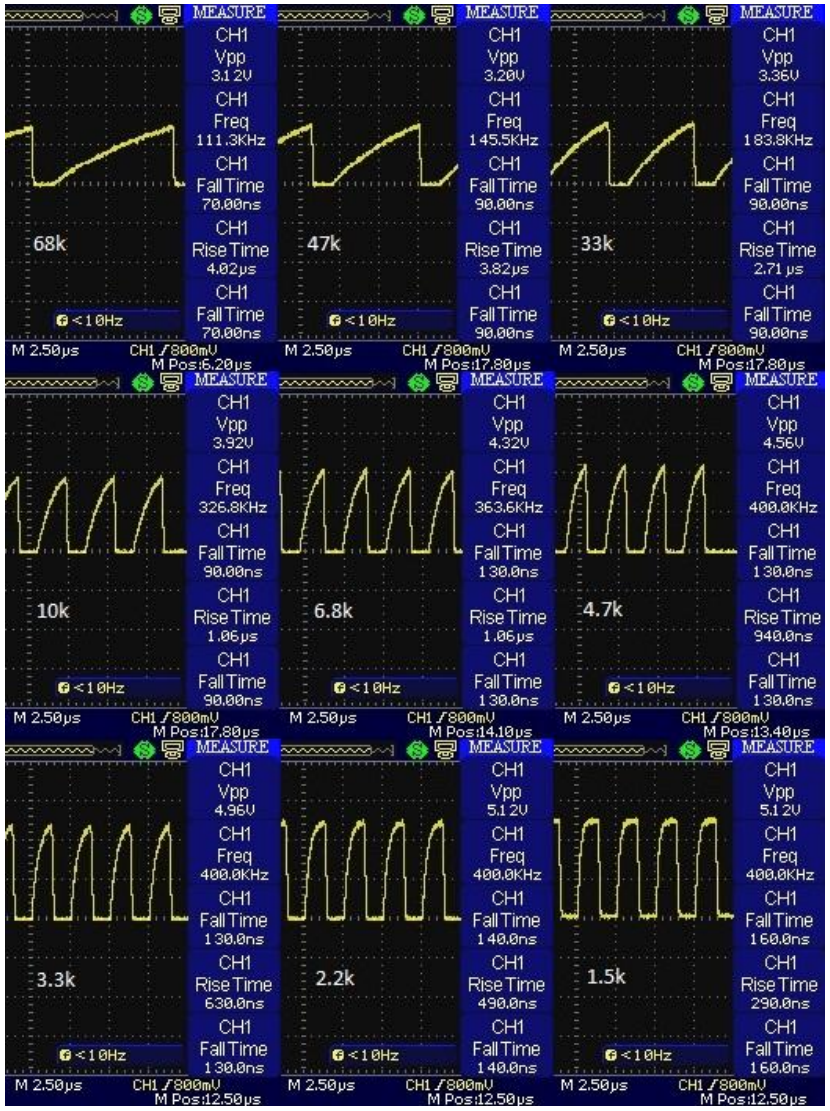
ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Влияние подтягивающих резисторов

100 кГц

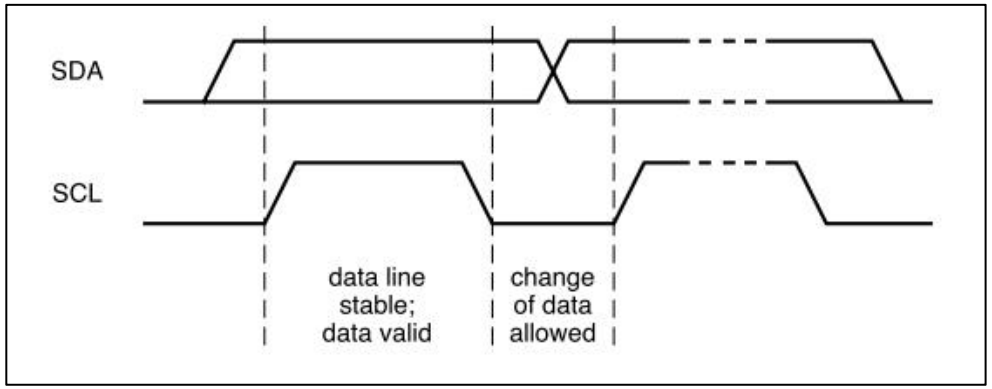


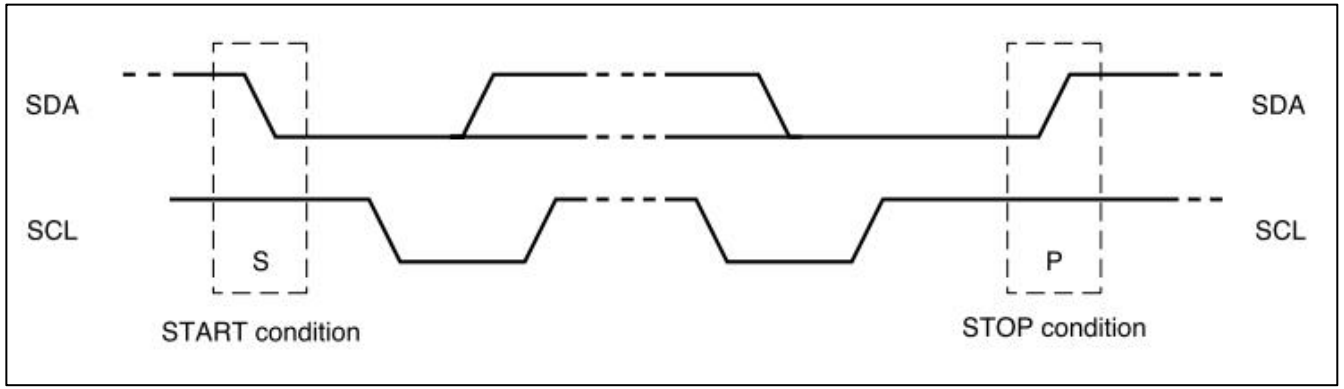
400 кГц

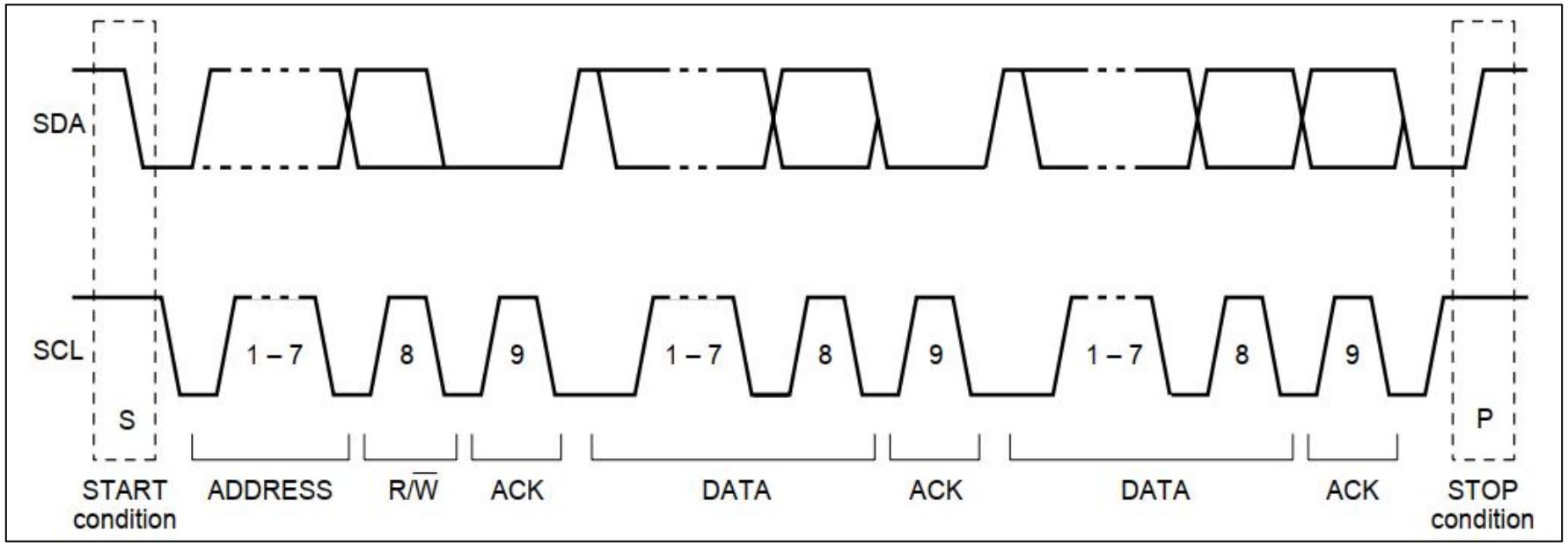


ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Временные диаграммы интерфейса I2C



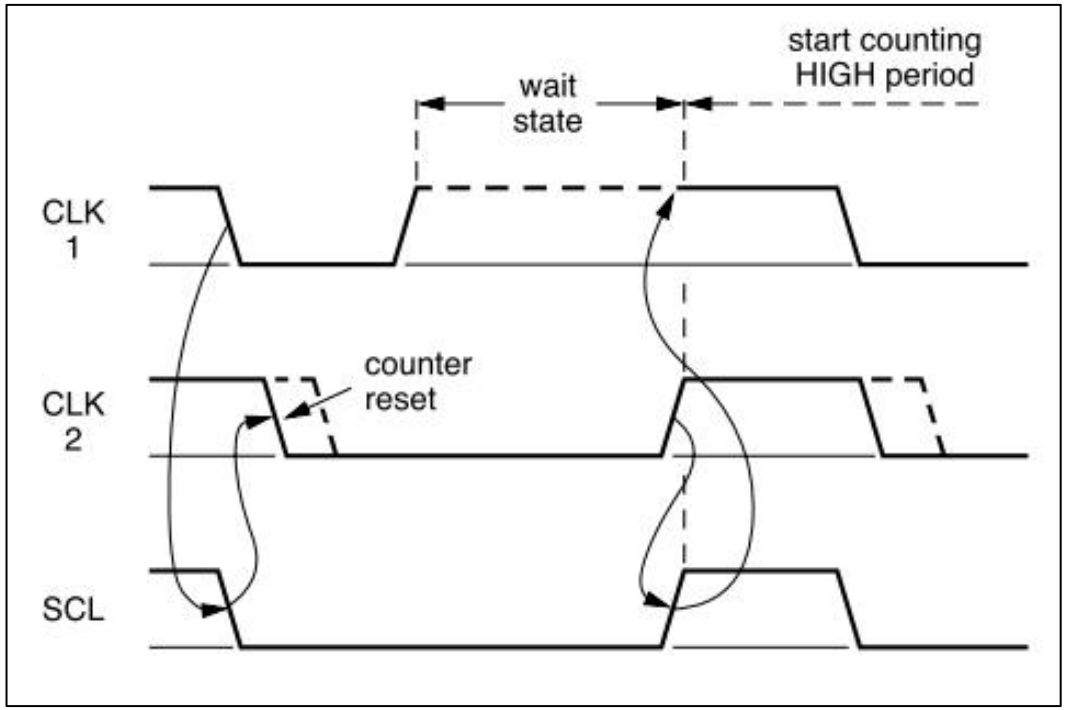




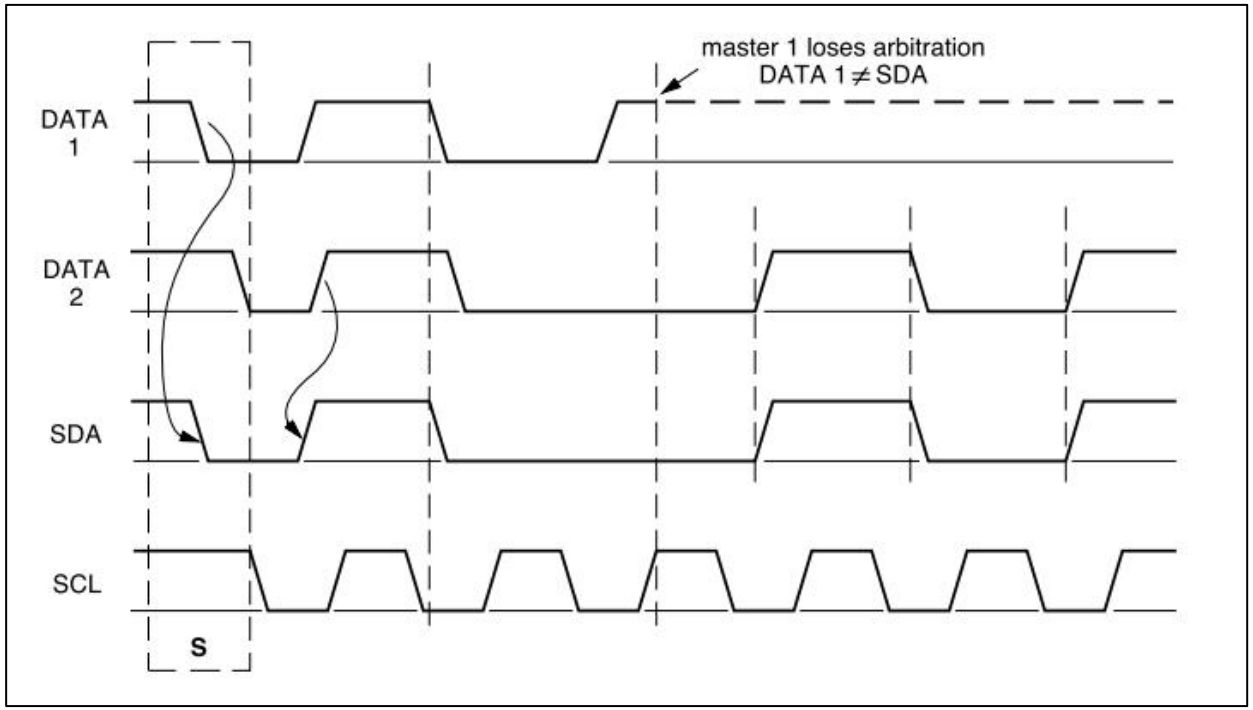
ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Работа нескольких Мастеров на шине I2C

Синхронизация SCL

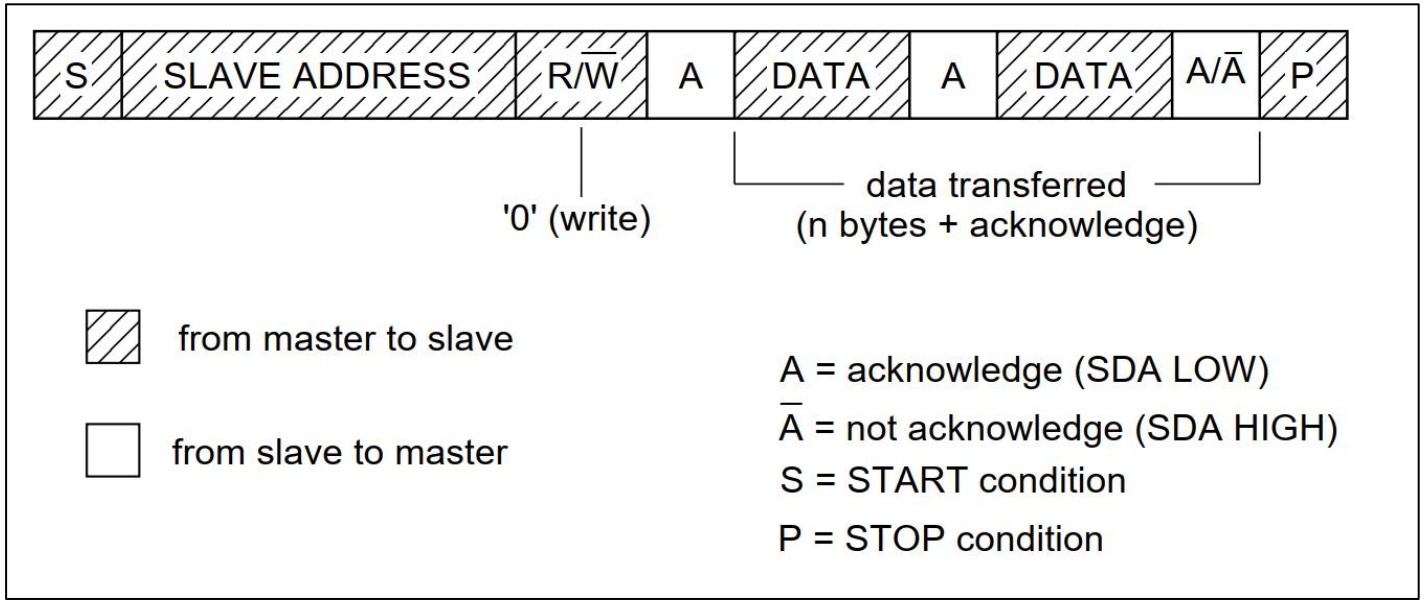


Арбитраж по линии SDA

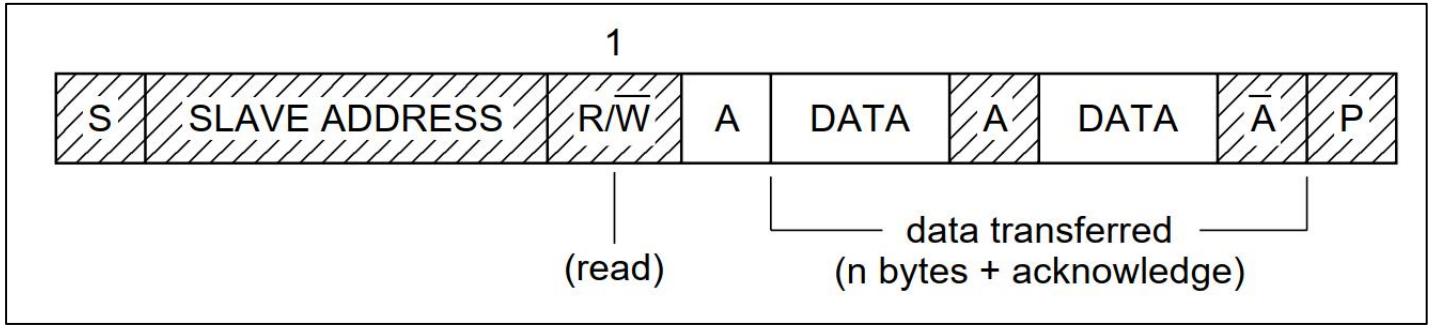


ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Структуры пакетов интерфейса I2C



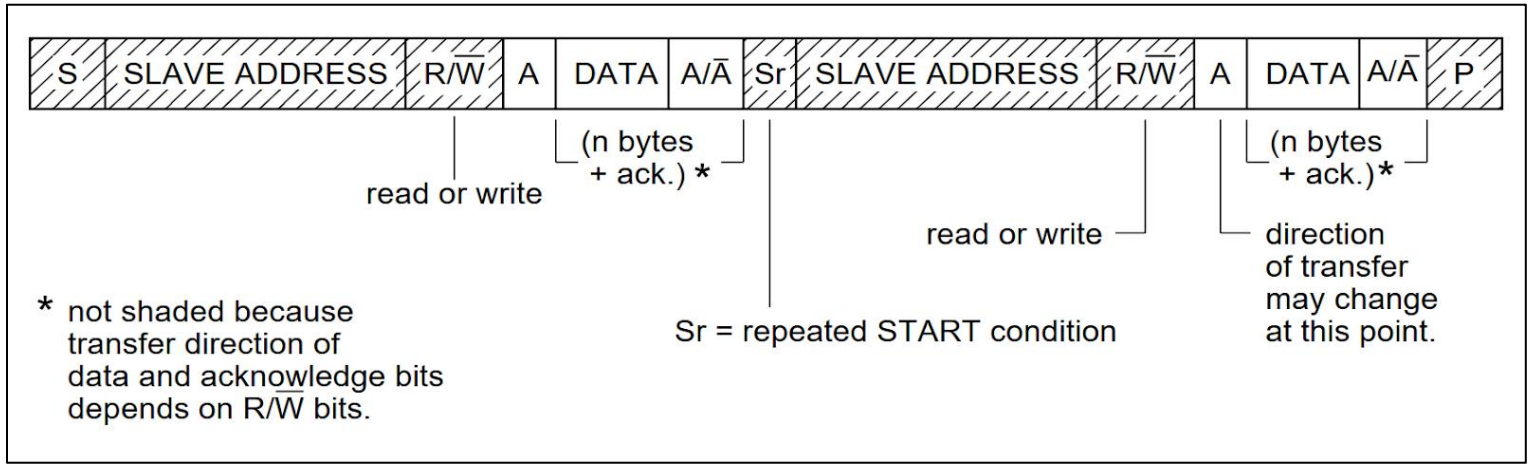
Запись в slave
7-ми битный адрес

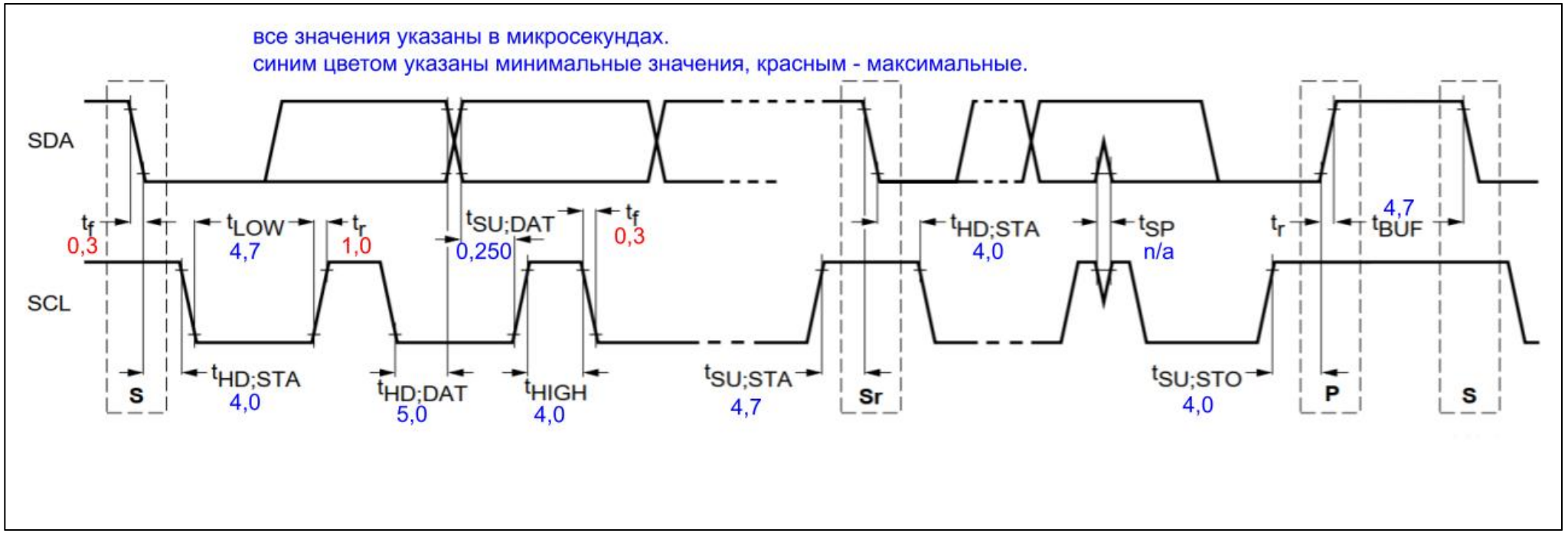


Чтение из slave
7-ми битный адрес

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Структуры пакетов интерфейса I2C

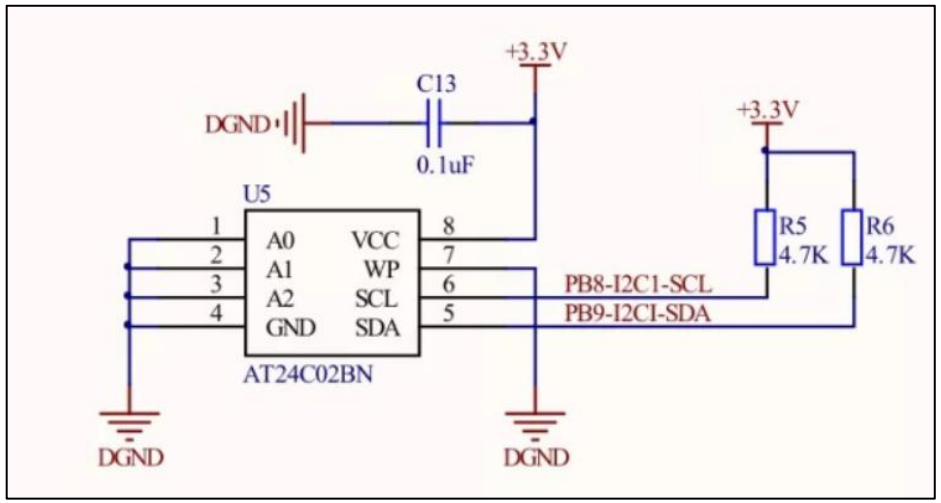




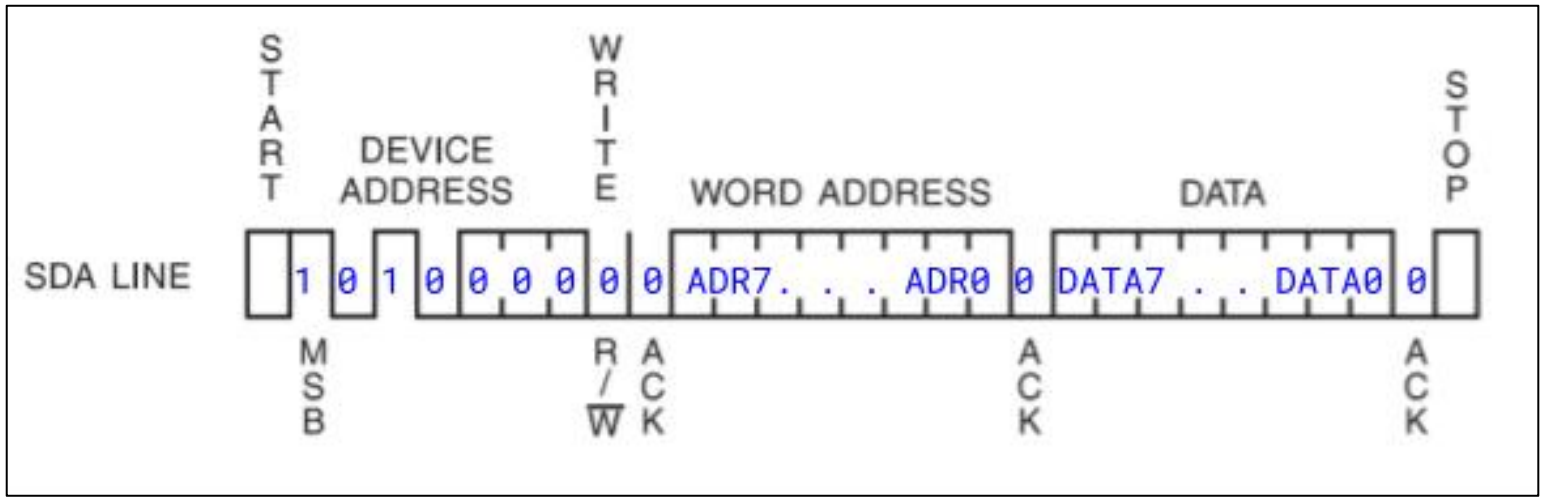
ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Схема подключения микросхемы I2C

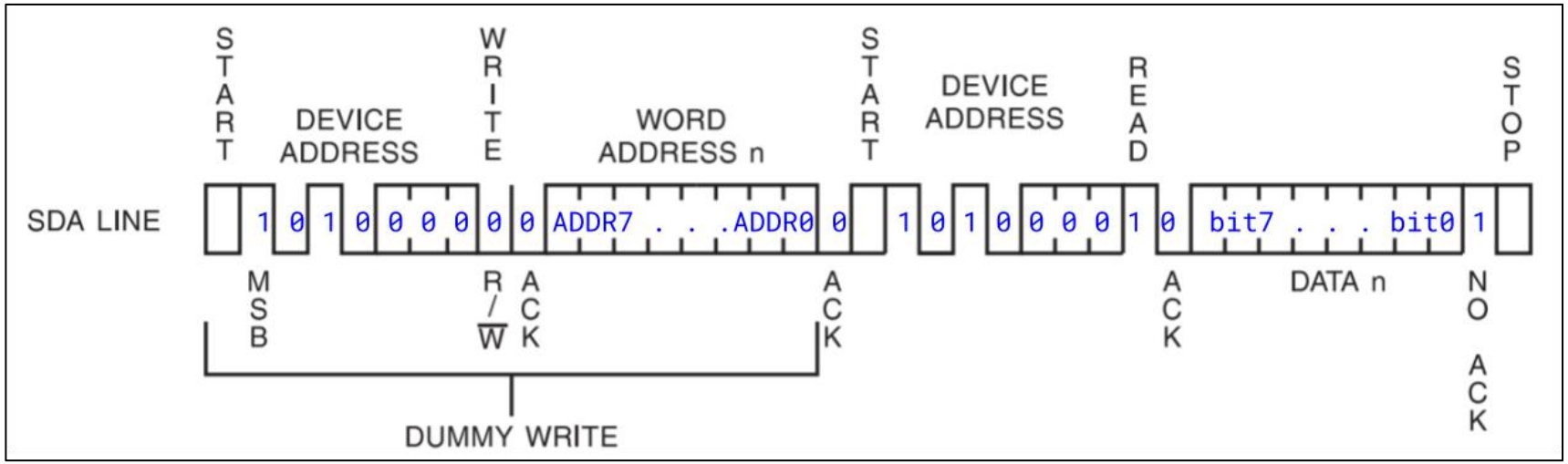
Схема подключения EEPROM



Запись одного байта в EEPROM в ячейку ADDR



Чтение одного байта из EEPROM из ячейки ADDR



ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Настройки модуля I2C1 в режиме Slave

- Включить тактирование модуля I2C от шины периферии (I2C1 подключен на шину APB1)

```
RCC -> APB1ENR |= RCC_APB1ENR_I2C1EN;      // включение тактирования модуля I2C1
```

- Указать входную тактовую частоту шины периферийных устройств в регистре I2C_CR2. В нашем случае это частота шины APB1, т.к. именно к этой периферийной шине подключен модуль I2C1, который мы будем использовать в нашем проекте.

```
I2C1 -> CR2 |= (42 << I2C_CR2_FREQ_Pos); // CR2_FREQ = 42 т.к. мы настроим Freq_APB1 = 42MHz
```

- Дополнительно нужно записать адрес нашего I2C модуля на шине I2C в регистр I2C_OAR1 в поле ADD. Для 7-ми битного режима адресации используются с 1 по 7-ой биты, а для 10-ти битной адресации используется все поле с 0 по 9 биты.

```
I2C1 -> OAR1 |= (0x30 << I2C_OAR1_ADD1_Pos);       // address = 011_0000
// от бинарного числа 0011_0000 взяты 7 младших бит
```

- Указать режим адресации в том же регистре I2C_OAR1 в бите ADDMODE.

```
I2C1 -> OAR1 &= ~(I2C_OAR1_ADDMODE); // сбросить бит ADDMODE в 0 для 7-ми битной адресации
```

- Запрограммировать регистр I2C_CR1, чтобы включить генерацию битов ACK, если это требуется, и включить модуль I2C1 на работу.

```
I2C1 -> CR1 |= I2C_CR1_PE;                 // I2C1 enabled.
I2C1 -> CR1 |= I2C_CR1_ACK;                // разрешение генерации ACK после приема байтов.
/* бит I2C_CR1_ACK можно выставлять в 1 только после включения бита I2C_CR1_PE.
иначе бит I2C_CR1_ACK всегда будет сбрасываться в 0 аппаратно.*/
```

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Работа модуля I2C1 в режиме Slave

- I2C1 модуль ждет свой адрес (7-ми или 10-ти битный) на шине I2C, ожидая в регистре I2C_SR1 бит ADDR = 1. Этот бит говорит о том, что нами был получен наш адрес, обращение по шине I2C сейчас будет выполняться к нашему модулю I2C1.

```
while((I2C1 -> SR1 & I2C_SR1_ADDR) == 0){}; // ждем флаг I2C_SR1_ADDR = 1.
```

- Требуется очистить бит ADDR чтением регистров SR1 и SR2 последовательно.

```
(void)I2C1 -> SR1;
(void)I2C1 -> SR2; // очистка бита ADDR чтением регистров SR1 SR2
```

- Проверить, модуль I2C1 находится в режиме передатчика или приемника

```
if ((I2C1 -> SR2 & I2C_SR2_TRA) != 0){// TRA = 1, I2C1 в режиме передатчика
    .....
}
else{                  // TRA = 0, I2C1 в режиме приемника
    .....
}
```

  - Если слейв в режиме передатчика то далее требуется записать байт для передачи в регистр I2C_DR.

```
I2C1 -> DR = Tx_Byte; // запись байта для отправки в регистр данных I2C1_DR
```

  - Ожидается, пока байт данных будет передан по I2C и в ответ на него будет получен бит ACK от устройства. Когда в ответ на байт данных мастер отвечает битом ACK, то в регистре I2C_SR1 выставляется бит TXE = 1. Который показывает, что передатчик пуст и готов принять новый байт данных для отправки.

```
while((I2C1 -> SR1 & I2C_SR1_TXE) == 0){};     // ждем флаг I2C_SR1_TXE = 1. завершение
передачи байта
```

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Работа модуля I2C1 в режиме Slave (продолжение)

- Если слейв в режиме приемника. После получения адреса и очистки ADDR ведомое устройство получает байты по линии SDA в регистр I2C_DR через внутренний сдвиговый регистр.

```
while((I2C1 -> SR1 & I2C_SR1_RXNE) == 0){};  // ожидание получения байта данных
Rx_Byte = I2C1 -> DR;     // чтение регистра данных
```

- После каждого полученног байта интерфейс генерирует последовательно:
  Импульс подтверждения ACK, если установлен бит ACK в регистре I2C_CR1.
  Бит RxNE устанавливается аппаратно, и прерывание генерируется, если биты ITEVFEN и ITBUFEN установлены

- Если установлен RxNE и данные из регистра I2C_DR не считываются до окончания приема следующего байта данных, то в регистре I2C_SR1 бит BTF устанавливается в 1 и интерфейс ждет, пока BTF не очистится при чтении из регистра I2C_DR. При этом модуль I2C увеличивает время паузы на линии SCL – держит там низкий уровень.

Тут нужно отметить, что в 10-битном режиме после получения адреса устройства модуль I2C (в слейв режиме) всегда находится в режиме приемника. Он перейдет в режим передатчика после получения повторного START-условия, за которым следует заголовок с совпадающими битами адреса и выставленный в единицу младший бит (11110xx1 где xx – старшие биты адреса нашего I2C модуля). Бит TRA в регистре I2C_SR2 показывает, находится модуль I2C в режиме приемника или передатчика.

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Настройка модуля I2C1 в режиме Master

- Включить тактирование модуля I2C от шины периферии (I2C1 подключен на шину APB1)

```
RCC -> APB1ENR |= RCC_APB1ENR_I2C1EN;        // включение тактирования модуля I2C1
```

- Указать входную тактовую частоту шины периферийных устройств в регистре I2C_CR2. В нашем случае это частота шины APB1, т.к. именно к этой периферийной шине подключен модуль I2C1, который мы будем использовать в нашем проекте.

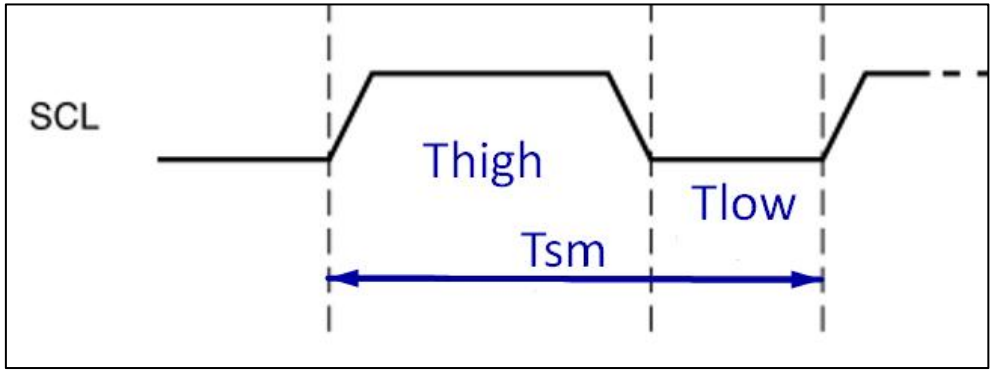
```
I2C1 -> CR2 |= (42 << I2C_CR2_FREQ_Pos);  // CR2_FREQ = 42 т.к. Freq_APB1 = 42MHz
```

- Настроить регистр управления тактовыми сигналами I2C_CCR.

```
I2C1 -> CCR  |= (210 << I2C_CCR_CCR_Pos);       // 100 КГц
I2C1 -> CCR  &= ~(I2C_CCR_FS);// сброс бита FS, работа на частоте 100 кГц (Standard Mode)
```

- **Вычисление значения CCR:**

I2C работает на частоте 100 кГц - Standard mode
Thigh = CCR * T_plck1
Tlow = CCR * T_pclk1
Tsm = 1/(I2C_freq) = 1/100000 = Thigh + Tlow;
1/100000 = 2 * CCR * T_pclk1
CCR = 1 / (2*100000*T_pclk1)
T_pclk1 = 1 / Freq_APB1;
Freq_APB1 = 42 MHz
T_Pclk1 = 1 / 42000000
CCR = 42000000 / (2*100000) = 210;



ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Настройка модуля I2C1 в режиме Master (продолжение)

- Настроить регистр времени нарастания сигнала I2C_TRISE.

```
I2C1 -> TRISE |=  (43 << I2C_TRISE_TRISE_Pos); // значение поля = I2C1_CR2_FREQ + 1 = 42+1 = 43
```

- Выбрать режим адресации 7 или 10 бит в регистре I2C_OAR2.

```
I2C1 -> OAR1  &=  ~(I2C_OAR1_ADDMODE); // использование 7-ми битного адреса устройства на шине I2C
// для 10-ти битной адресации требуется ADDMODE выставить равным 1
```

- Запрограммировать регистр I2C_CR1, чтобы включить генерацию битов ACK, если это требуется, и включить модуль I2C1 на работу.

Установить бит START в регистре I2C_CR1 для создания условия начала транзакции.

```
I2C1 -> CR1 |= I2C_CR1_PE;            // I2C1 enabled.
I2C1 -> CR1 |= I2C_CR1_ACK;           // разрешение генерации ACK после приема байтов.
/* бит I2C_CR1_ACK можно выставлять в 1 только после включения бита I2C_CR1_PE. иначе бит I2C_CR1_ACK всегда будет сбрасываться в 0 аппаратно.*/
```

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Работа модуля I2C1 в режиме Master

- Чтобы начать передавать данные нужно программно сформировать START-условие. Нужно выставить в регистре I2C_CR1 бит START.

```
I2C1->CR1  |=  I2C_CR1_START;
```

- Проверить, что в регистре I2C_SR1 бит SB = 1. Это говорит о том, что на линии действительно было сформировано START-условие.

```
while((I2C1 -> SR1 & I2C_SR1_SB) == 0){}; // дождаться START-условия на шине I2C
```

- Записать в регистр I2C_DR байт, содержащий адрес устройства в старших 7-ми битах (у нас 7-ми битная адресация) и младший бит – бит RW = 0 (запись во внешнее устройство).

```
I2C1 -> DR = Tx_Byte;     // отправить в I2C_DR адрес устройства и бит WR
```

- Проверить, что в регистре I2C_SR1 бит ADDR = 1. Это значит, что адрес был отправлен и устройство прислало на него ACK-бит.

```
while((I2C1 -> SR1 & I2C_SR1_ADDR) == 0){};        // ждем флаг I2C_SR1_ADDR = 1
```

- Требуется очистить бит ADDR чтением регистров SR1 и SR2 последовательно.

```
(void)I2C1 -> SR1;
(void)I2C1 -> SR2;        // очистка бита ADDR чтением регистров SR1 SR2
```

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Работа модуля I2C1 в режиме Master (продолжение)

- Записывается байт данных в регистр I2C_DR.

```
I2C1 -> DR = Tx_Byte;     // отправка байта данных
```

- Ожидается, пока байт данных будет передан по I2C и в ответ на него будет получен бит ACK от устройства. Когда в ответ на байт данных устройство отвечает битом ACK, то в регистре I2C_SR1 выставляется бит TXE = 1.

```
while((I2C1 -> SR1 & I2C_SR1_TXE) == 0){}; // ждем флаг I2C_SR1_TXE = 1.
```

- После этого можно записывать следующий байт в регистр I2C_DR для отправки в устройство.

- Для завершения передачи данных, после ожидания бита TXE = 1, нужно выставить в регистре I2C_CR1 бит STOP.

```
I2C1->CR1  |=  I2C_CR1_STOP; // генерация STOP-условия
```

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Итоги урока

- Рассмотрена структура шины I2C.
- На линиях I2C обязательно наличие подтягивающих резисторов.
- Линии I2C являются двунаправленными.
- Скорость передачи у интерфейса низкая (1 байт = 90 мкс). В быстродействующих приложениях обязательно работать с модулем I2C через прерывания.

ООО «РСВ Электроникс»
Сайт: pcbteach.ru
E-Mail: pcbteach@yandex.ru

Программирование микроконтроллеров.

Спасибо за внимание!