**Лабораторная работа 8**

**Сканнер I2C**

Мощевикин А.П., ПетрГУ

***Общие сведения об интерфейсе I2C***

Inter-Integrated Circuit (IIC или I2C) – последовательный интерфейс для связи цифровых устройств, подключенных по двум линиям – тактирования SCL и передачи данных SDA. Ведущее устройство (мастер) задает такт, выбирает адрес ведомого устройства и задает режим (чтение/запись) для инициированной транзакции. Максимальное количество устройств на одной двухпроводной шине – 127. В I2C также допускается ситуация одновременного подключения нескольких мастер-устройств к шине.

Чаще всего используются скорости передачи 100 кбит/с или 400 кбит/с (межбитовый интервал 10 мкс или 2.5 мкс, соответственно).

Ключевым схемотехническим решением передачи данных по шине является обязательное использование подтягивающих резисторов (pull-up) на обеих линиях, как показано на рисунке 1.



Рис. 1 – Схема подключения устройств по шине I2C [2].

Передающее устройство (мастер или слейв) может либо выставить явный ноль на линию («прижать к земле»), либо «отпустить» линию. При отпускании напряжение в линии само подтянется к питанию за счет резистора. Скорость нарастания напряжения зависит от емкости применяемых проводов и электронной обвязки, а также от номинала подтягивающего резистора (RC-цепь). Чем больше сопротивление резистора, тем дольше заряжаются паразитные емкости, тем сильнее заваливается фронт импульса.

Если при подключении модуля к шине I2C не наблюдается связи, особенно на высоких скоростях, номинал резистора иногда приходится подбирать экспериментально, либо переключаться на более низкие скорости передачи данных.

Передача данных начинается из состояния START (при SCL=1, линия SDA опускается вниз), как показано на рисунке 2. Далее ведущее устройство выставляет первый байт на линии SDA: адрес (7 бит) и бит операции (чтение – 1, запись – 0). Ведомое устройство считывает значение бита на линии SDA в момент, когда SCL становится 1. Таким образом, данные на линии SDA валидны только во время верхней полки сигнала SCL.

Ведомое устройство квитирует получение байта, выставляя бит ACK: мастер отпускает линию, и ведомое устройство самостоятельно держит линию в нуле. Таким образом, значение ACK=0 соответствует состоянию корректно принятого байта.



Рис. 2 – Битовая структура сообщения [2].

Сигналы по линии SCL всегда формируется ведущим устройством, т.е. тем, кто инициировал транзакцию, выставив состояние START на магистрали.

Содержимое первого байта по линии SDA всегда выставляется ведущим устройством, а вот последующие байты могут формироваться либо ведущим (операция записи), либо ведомым (операция чтения).

Как уже было сказано, каждый из байтов должен быть квитирован получателем. В конце каждого байта, когда передатчик отпускает линию SDA, в бите подтверждения может быть либо 0 (сигнал ACK), либо 1 (сигнал NACK – Negative ACK). При записи байтов их квитирует ведомое устройство, при чтении – ведущее.

Если значение бита ACK равно логической единице (состояние NACK), то произошла одна из следующих ситуаций.

1. NACK в первом байте выставлен, поскольку на шине нет устройства с указанным адресом (никто не прижал линию к нулю, сработал подтягивающий резистор).

2. Для команды записи и последующих байтов с данными выставленный NACK означает, что либо слейв не распознал команду, либо не может принять эти данные.

3. Во время операции многобайтового чтения (когда мастер квитирует полученные данные), выставленный мастером NACK после очередного байта сообщает слейву прекратить передачу, на магистраль будет выставлено состояние STOP.

Передача данных заканчивается состоянием STOP (в момент, когда SCL=1, на линии SDA выставляется логическая 1, см. рисунок 2).

Реальные сигналы по линиям SDA и SCL представлены на рисунке 3. Зафиксирован момент опроса микроконтроллером Комдив-МК внешней микросхемы EEPROM-памяти M24C32 по шине I2C0 (микросхема D8 на плате БАГЕТ-ПЛК1-01). Установленные на плате резисторы подтяжки обеспечивают передний фронт импульсов ~1 мкс.



Рис. 3 – Осциллограмма сигналов на линиях SDA (синий луч) и SCL (желтый луч) при выдаче команды на запись для устройства c I2C-адресом 0x54. Скорость передачи данных 400 кбит/с.

Количество и содержимое передаваемых байтов определяется производителем подключенного устройства.

В качестве примера на рисунке 4 представлен перехват I2C-трафика с помощью логического анализатора протоколов.



Рис. 4 – Пример I2C-трафика.

Ведущее устройство сначала сформировало первую транзакцию, выставив команду записи (бит W=0) для устройства с адресом 0x60 (0b1100000), а затем тут же ее оборвало, выставив состояние STOP. А затем повторило команду записи и передало два байта 0xB7 и 0x80. На все принятые байты ведомое устройство отвечало битом ACK (выставляло уровень логического нуля).

Необходимо отметить, что I2C-адрес для устройства часто задается только аппаратно при изготовлении. При этом некоторые производители в качестве адреса могут указывать не действительный 7-битный I2C-адрес устройства (сдвигаемый на 1 бит влево из-за добавления справа бита R/W), а получаемое 8-битное значение байта команды (например, не 0x60, как на рисунке 3, а 0xC0).

***Согласование уровней сигналов***

При подключении устройств по шине I2C следует уделять внимание согласованию уровней сигналов. Нельзя подключать на одну шину устройства, имеющие разные уровни напряжения питания. Например, при напряжении питания у микроконтроллера +3.3 вольта, а у датчика +5 вольт. Или наоборот.

Для согласования уровней сигналов используется преобразователь, встраиваемый в шину I2C между группами устройств с напряжением питания +3.3 В и +5 В, как показано на рисунке 5.



Рис. 5 – Согласование уровней напряжения сигналов.

Преобразователь работает следующим образом.

Затвор полевого транзистора BSS138 подключен к меньшему из двух напряжений питаний (к шине +3.3 вольта). Исток транзистора соединен с цифровой линией I2C низковольтной группы устройств, сток транзистора – с цифровой линией высоковольтной группы устройств.

В момент, когда ни одно из устройств не подтягивает цифровую линию к нулю, разница напряжений на затворе и истоке равна нулю, и транзистор закрыт. При закрытом транзисторе уровень напряжения на стоке будет равен напряжению +5 В. Таким образом, линии шин слева и справа от транзистора находятся в высоком состоянии, но имеют различные уровни напряжений.

Если одно из трехвольтовых устройств слева от транзистора подтягивает цифровую линию к нулю, это создает разницу напряжения между истоком и затвором транзистора, и он открывается. Открытый транзистор замыкает пятивольтовую часть линии на трехвольтовую: линии шин слева и справа от транзистора переходят в состояние логического нуля.

Если одно из пятивольтовых устройств справа от транзистора подтягивает цифровую линию к нулю, встроенный «конструктивный» диод (исток-сток) транзистора открывается, и напряжение на трехвольтовой части шины также становится близким к нулю вольт.

***Общие сведения о контроллерах I2C в микроконтроллере Комдив-МК***

Микроконтроллер Комдив-МК содержит три контроллера I2C: I2C0, I2C1 и I2C2. Все они могут работать с использованием прерываний: по линиям 18, 19 и 41 соответственно [3].

Соответствие сигнальных линий SDA и SCL линиям портов ввода-вывода и номерам альтернативных функций в скобках приведено ниже [4].

I2C0\_SDA – GPIO\_B7 (1)

I2C0\_SCL – GPIO\_B6 (1)

I2C1\_SDA – GPIO\_C1 (1)

I2C1\_SCL – GPIO\_C0 (1)

I2C2\_SDA – GPIO\_D7 (3)

I2C2\_SCL – GPIO\_D6 (3)

Базовые адреса контроллеров I2C определены в файле ./psp\_mc/arch/kmk/uke.h.

#define I2C0\_BASE 0xbb400110

#define I2C1\_BASE 0xbb400120

#define I2C2\_BASE 0xbb400130

Взаимодействие с контроллером I2C осуществляется через следующие регистры [3].

Address (8-битный регистр, смещение 0х00) – регистр адреса ведомого устройства, старшие 7 бит – адрес, младший бит – указание операции запись/чтение (0/1).

Data (8-битный регистр, смещение 0х01) – регистр данных.

Control (8-битный регистр, смещение 0х02, биты IEN-ENAB-STA-STP-IFLG-AAK-X-X) – регистр управления, бит 7 (IEN) – разрешение прерывания, бит 6 (ENAB) – разрешение работы, бит 5 (STA) – выставление START в режиме ведущего, бит 4 (STP) – выставление STOP в режиме ведущего, бит 3 (IFLG) – флаг прерывания (события), бит 2 (AAK) – разрешение выставления подтверждения, биты 0..1 – резерв.

Reset (8-битный регистр, смещение 0х07) – регистр сброса (запись любого значения приводит к сбросу).

Frequency (8-битный регистр, смещение 0х03) – регистр частоты на шине (только запись), биты 0..2 – число N, биты 3..6 – число M. Значения M и N для битовой скорости 100 кБит/с – [M=11, N=1], для битовой скорости 400 кБит/с – [M=2, N=1].

Status (8-битный регистр, смещение 0х03) – регистр состояния (только чтение), только старших 5 битов – значимые, содержит текущий статус в соответствии с таблицей.

| Код | Статус |
| --- | --- |
| 0x00 | Ошибка на шине |
| 0x08 | Ситуация START была выставлена на шину |
| 0x10 | Ситуация повторного START выставлена |
| 0x18 | Адрес при записи выставлен, подтверждение получено |
| 0x20 | Адрес при записи выставлен, подтверждение не получено |
| 0x28 | Данные выставлены в режиме ведущего, подтверждение получено |
| 0x30 | Данные выставлены в режиме ведущего, подтверждение не получено |
| 0x38 | Потеря арбитража при попытке выставления адреса или данных |
| 0x40 | Адрес при чтении выставлен, подтверждение получено |
| 0x48 | Адрес при чтении выставлен, подтверждение не получено |
| 0x50 | Данные приняты в режиме ведущего, подтверждение получено |
| 0x58 | Данные приняты в режиме ведущего, подтверждение не получено |
| 0x60 | Получен адрес записи, выставлено подтверждение |
| 0x68 | Потерян арбитраж в режиме ведущего, получен адрес записи, выставлено подтверждение |
| 0x70 | Получен широковещательный адрес, выставлено подтверждение |
| 0x78 | Потерян арбитраж в режиме мастера, получен широковещательный адрес, выставлено подтверждение |
| 0x80 | Получен байт данных после адреса, выставлено подтверждение |
| 0x88 | Получен байт данных после адреса, подтверждение не выставлено |
| 0x90 | Получен байт данных в широковещательной посылке, выставлено подтверждение |
| 0x98 | Получен байт данных в широковещательной посылке, подтверждение не выставлено |
| 0xA0 | В режиме ведомого получен STOP или повторный START |
| 0xA8 | Получен адрес чтения в режиме ведомого, выставлено подтверждение |
| 0xB0 | Потерян арбитраж в режиме ведущего, получен адрес чтения в режиме ведомого, выставлено подтверждение |
| 0xB8 | Отправлен байт в режиме ведомого, получено подтверждение |
| 0xC0 | Отправлен байт в режиме ведомого, подтверждение не получено |
| 0xC8 | Последний байт передан в режиме ведомого, получено подтверждение |
| 0xD0 | Выставлен второй байт адреса записи, получено подтверждение |
| 0xD8 | Выставлен второй байт адреса записи, подтверждение не получено |
| 0xE0 | Резерв |
| 0xE8 | Резерв |
| 0xF0 | Резерв |
| 0xF8 | Определенной ситуации на шине нет, ожидание |

В файле ./psp\_mc/arch/kmk/kmk\_i2c.c определены смещения регистров контроллеров относительно базового адреса.

#define I2C\_REG\_DATA 1

#define I2C\_REG\_CTRL 2

#define I2C\_REG\_STATUS 3

#define I2C\_REG\_FREQ 3

#define I2C\_REG\_RESET 7

***Инициализация контроллера I2C***

1. Установить альтернативный режим (указать номер) использования линий портов ввода-вывода для линий передачи данных SDA и тактирования SCL для выбранного контроллера I2C. Например, для I2C0…

GPIO\_SetPinMode(GPIOB\_BASE, 6, GPIO\_MODE\_ALT1);

GPIO\_SetPinMode(GPIOB\_BASE, 7, GPIO\_MODE\_ALT1);

2. Установить частоту тактирования (битовую скорость). Например, для выставления скорости 400 кбит/с записать в регистр I2C\_REG\_FREQ значения M=2, N=1.

\*((volatile unsigned char \*)(I2C0\_BASE + I2C\_REG\_FREQ)) = (2<<3) + 1;

3. Остановить предыдущую передачу (на всякий случай), выставив биты ENAB и STP в регистре I2C\_REG\_CTRL.

\*((volatile unsigned char \*)(I2C0\_BASE + I2C\_REG\_CTRL)) = 0x50;

***Передача данных***

Как уже было сказано, процесс передачи и чтения данных по шине I2C – многостадийный.

После выдачи команды контроллеру и ее отработки, контроллер взводит флаг IFLG в регистре управления I2C\_REG\_CTRL. Одновременно устанавливается новый статус в регистре статуса I2C\_REG\_STATUS. Анализируя статус с помощью таблицы, программа переходит к следующей стадии.

Этот процесс реализован в функции static unsigned char kmk\_i2c\_fsm(kmk\_i2c\_ctrl \*pI), содержащейся в файле ./psp\_mc/arch/kmk/kmk\_i2c.c.

***Задания для самостоятельной работы***

1. Тщательно изучить текст пособия по лабораторной работе, включая таблицу статусов, и статью [1].
2. Изучить принципиальную схему платы БАГЕТ-ПЛК1-01 (лист 6, микросхемы D1, D8, D13), разобраться с подключением микросхемы D8 EEPROM-памяти M24C32 по шине I2C0. Активировать аппаратный доступ к EEPROM-памяти – установить перемычку SA9 на плате БАГЕТ-ПЛК1-01.
3. Используя исходный код лабораторной работы 1 для проекта на языке Си, создать программу, которая периодически (раз в 5 секунд) сканирует шину I2C0 и выводит адреса подключенных устройств.

*Возможный алгоритм программы*

В цикле для всего пространства 7-битных адресов…

1. Установить состояние START на шине (выставить биты ENAB и STA в регистре I2C\_REG\_CTRL).

2. Опрашивая регистр I2C\_REG\_CTRL, дождаться появления бита IFLG.

3. Считать регистра статуса I2C\_REG\_STATUS. Если значение в норме (0x8 или 0x10, т.е. состояние START наступило), то в регистр данных I2C\_REG\_DATA записать следующий адрес (сместив его влево на 1 бит), и стартовать передачу, выставив бит ENAB в регистре I2C\_REG\_CTRL. Иначе, корректно завершить итерацию цикла, и реиницилизировать контроллер I2C.

4. Опрашивая регистр I2C\_REG\_CTRL, дождаться появления бита IFLG.

5. Считать регистра статуса I2C\_REG\_STATUS. Если значение статуса 0x18 или 0x40, то устройство с таким адресом существует на шине I2C, вывести адрес в консоль. Иначе возникла ситуация с NACK в конце первого байта, устройства с таким адресом на шине не оказалось.

6. Корректно остановить передачу, выставив биты STOP в регистре I2C\_REG\_CTRL.

Если на шине не было найдено ни одного устройства, в конце цикла вывести сообщение об этом.

1. Опционально. При наличии стороннего модуля I2C с питанием 3.3 В, подключить его параллельно EEPROM D8, убедиться, что программа находит модуль.
2. Опционально. При наличии стороннего модуля I2C с питанием 3.3 В, подключить его на другой контроллер, например, I2C1. Модифицировать программу для сканирования устройств шины I2C1.

Разработчики платы БАГЕТ-ПЛК1-01 [5] рекомендуют использовать среду разработки и структуру проекта, описанные в [6] и [7, раздел 3].

Задания к лабораторной работе подразумевают, что работа выполняется в ОС Windows с предустановленной виртуальной машиной WSL2 с Debian [6] и ППП МК [7-8]. Проекты на языке Cи следует создавать в папке ./psp\_mc/apps/ .

Для формирования импульса можно использовать функцию delay\_us(), для замера времени работы –\_\_micros(), для вывода результата – \_\_print\_console(), для форматирования строки вывода – itoa() и intToHex().

***Список литературы***

[1] Интерфейсная шина IIC (I2C) // http://easyelectronics.ru/interface-bus-iic-i2c.html

[2] I2C: How it works and how to use it? // https://ensatellite.com/i2c/

[3] Микросхема интегральная К5500ВК018. Контроллеры I2C и I3CКонтроллер. ЮКСУ.431295.019Д4.12 // К5500ВК018\_d4\_12\_I2C\_v1\_I3C\_v1.doc

[4] Микросхема интегральная К5500ВК018. Контроллер GPIO. ЮКСУ.431295.019Д4.19 // К5500ВК018\_d4\_19\_GPIO\_v.2.4.2.doc

[5] ПЛК «БАГЕТ-ПЛК1-01» Руководство по эксплуатации. ЮКСУ.421457.002-01РЭ. https://www.niisi.ru/БАГЕТ-ПЛК1-01\_РЭ\_v3.3.pdf

[6] Мощевикин А.П., Голяков М.А. Установка среды разработки в Windows 10 для БАГЕТ-ПЛК1-01 // BAGET-PLK1-01\_getting\_started\_v\_2\_X.doc

[7] Пакет поддержки программирования микроконтроллера (ППП МК). Описание применения. ЮКСУ.91264-01 31 01 // ППП\_МК\_31\_Описание\_применения v12.docx

[8] Пакет поддержки программирования микроконтроллера (ППП МК). Руководство программиста. ЮКСУ.91264-01 33 01 // ППП\_МК\_33\_Руководство\_программиста\_v5.docx