MicroConverter™ Техническое замечание - µC001
Использование I²C интерфейса MicroConverter

ВСТУПЛЕНИЕ

Это техническое замечание описывает аппаратную и программную реализацию I2C совместимого интерфейса встроенного в кристаллы MicroConverter.

Основные возможности шины I2C включают:

- необходимо наличие только двух линий: последовательной линии данных (SDATA) и последовательной шины синхронизации (SCLOCK). Обе линии являются двунаправленными, подразумевая, что и ведущий, и ведомый могут быть как приемником, так и передатчиком.

- I2C мастер может соединяться с несколькими ведомыми устройствами. Поскольку каждой ведомое устройство имеет уникальный 7-битный адрес, одна взаимосвязь ведущий/ведомый может существовать все время, даже в конфигурации с несколькими ведомыми.

- При программной реализации ведущего выходные данные могут передаваться на скорости до 140 Кбит/с при тактовой частоте 12 МГц. Аппаратная реализация ведомого способна принимать данные со скоростью до 3.4 Мбит/с.

- Реализованная на кристалле фильтрация устраняет выбросы менее 50 нс на линиях SDATA и SCLOCK для сохранения целостности данных.

Типичная структурная схема I2C интерфейса приведена на Рис.1.

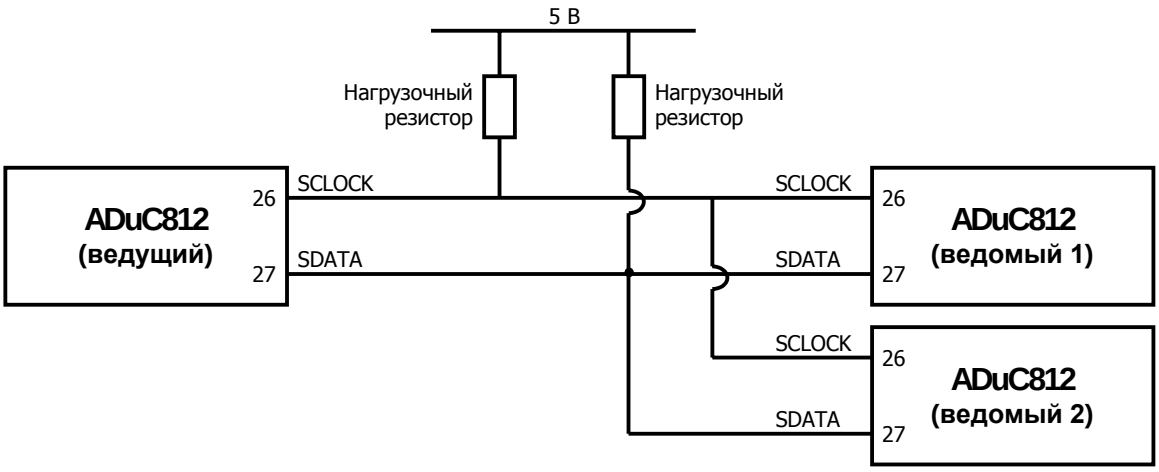


Рис.1. Структурная схема I²C интерфейса с несколькими ведомыми

Оригинал : версия 2.0 [10/2000]
Перевод : версия 2.0 [08/2001]

Техническое замечание µ0001

1.1 ОБЗОР ИНТЕРФЕЙСА I2C

Сигнал SCLOCK управляет передачей данных между ведущим и ведомым. Сигнал SCLOCK всегда передается от ведущего к ведомому. (Однако ведомый может держать эту линию на низком уровне, если он не готов к началу следующей передачи, такой режим называется «clock stretching» («растяжение синхронизации»), для уточнения деталей см. раздел 1.7.). Для каждого передаваемого бита данных генерируется один тактовый импульс.

Сигнал SDATA используется для передачи и приема данных. Вход SDATA должен быть устойчивым во время высокого уровня сигнала SCLOCK. Переход состояния линии SDATA во время высокого уровня линии SCLOCK трактуется как условия START или STOP (начала или завершения передачи, Рис.2а и Рис.2б).

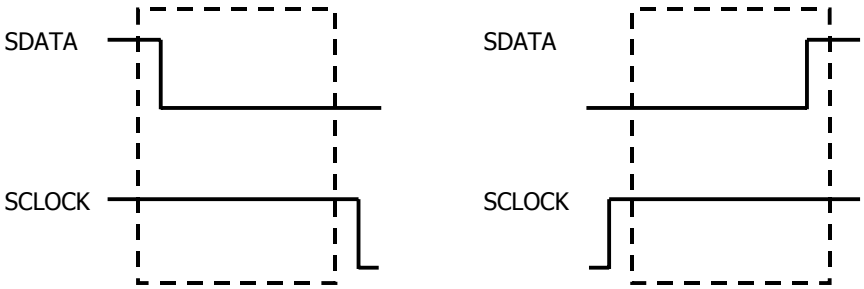
ПРИМЕЧАНИЕ:
Для ADuC812 на линиях SCLOCK и SDATA необходимо наличие нагрузочных резисторов. Для микросхем ADuC816 и ADuC824 такое повышение напряжения на выходе реализовано аппаратно.

1.2 ПОЛНАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ ПЕРЕДАЧИ ДАННЫХ

Условие START (начала передачи)

Типичная последовательность передачи данных по интерфейсу I2C начинается с условия START. Оно характеризуется переходом линии SDATA с высокого уровня на низкий, в то время когда линия SCLOCK удерживается на высоком уровне (Рис.2а). Ведущий является ответственным за генерирование условия START.

ПРИМЕЧАНИЕ:
Условия START (и STOP) являются единственными, когда линия SDATA меняет свое состояние во время высокого уровня линии SCLOCK. В течение нормальной передачи данных (включая адресацию ведомого) данные на линии SDATA должны быть устойчивы при высоком уровне линии SCLOCK.



а) Условие START б) Условие STOP

Рис.2. Условия START и STOP при передаче по интерфейсу I2C.

Адрес ведомого

После условия начала передачи ведущий посылает байт (сначала СЗР) по линии SDATA (вместе с восьмью тактовыми импульсами). Первые семь бит являются 7-битным адресом ведомого. Ведомый должен ответить ведущему только в том случае, если это 7-битное значение совпадает с его уникальным адресом (см. раздел 1.3).
Восьмой бит (МЗР) является разрядом R/W (чтения/записи). Бит статуса R/W определяет направление передачи. Если этот бит очищен, то ведущий будет передавать данные выбранному ведомому устройству. Если этот бит установлен, тогда ведущий ожидает передачу информации от ведомого.

Если ведомое устройство получает корректный адрес, оно возвращает сигнал подтверждения приема ACK (см. далее), переводит линию SCLOCK на низкий уровень и устанавливает бит прерывания I2CI (генерирует прерывание, если все установлено правильно). Линия SCLOCK находится на низком уровне, поэтому ведущее устройство знает, что ведомое еще не готово для принятия следующего байта. Очистка бита I2CI освобождает линию SCLOCK.

Подтверждение (ACK) / Отказ (NACK)

Если адрес ведомого устройства совпадает с адресом, запрошенным ведущим, оно автоматически посылает сигнал подтверждения (ACK), в ином случае оно посылает сигнал отказа (NACK). Сигнал ACK выглядит как низкий уровень линии SDATA на девятом тактовом импульсе. Сигнал NACK выглядит как высокий уровень линии SDATA на девятом тактовом импульсе (Рис.3).

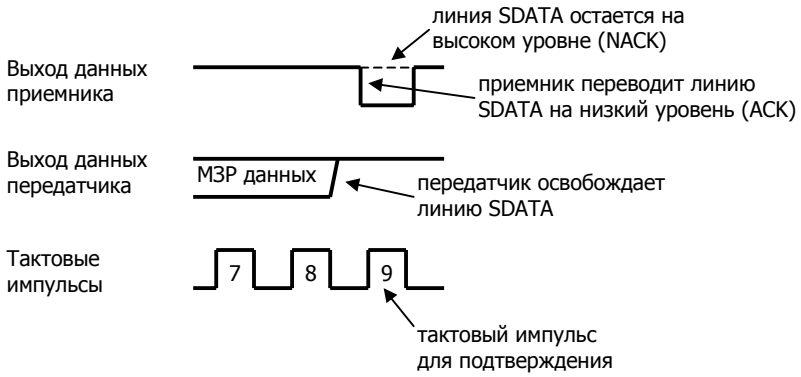


Рис.3. Подтверждение (ACK) и отказ (NACK) на шине I2C.

Во время передачи данных сигналы ACK и NACK всегда генерируются приемником. При этом девятый тактовый импульс, необходимый для этих сигналов, всегда генерируется ведущим. Передатчик должен освободить линию SDATA (высокий уровень) во время девятого тактового импульса. Для посылки сигнала ACK приемник должен перевести линию SDATA на низкий уровень.

Если MicroConverter сконфигурирован для режима ведомого устройства, и сигнал ACK, и сигнал NACK автоматически генерируются на аппаратном уровне по завершению приема каждого байта. Если MicroConverter сконфигурирован для режима ведущего, программа пользователя должна посылать сигнал ACK по завершению приема каждого байта. Если ведущее устройство получает от ведомого сигнал NACK (ведомое устройство не отвечает на адрес, запрошенный ведущим), оно должно сгенерировать условие STOP для прекращения передачи.

После получения последнего байта от ведомого устройства, ведущий должен послать сигнал NACK. После того как ведомый получает сигнал NACK, он освобождает линию SDATA, позволяя ведущему устройству генерировать условие STOP.

Передача данных

В стандартной программе обслуживания прерываний I2C (или в реализации системы с опросом) ведомое устройство решает когда передавать или принимать данные в зависимости от состояния бита R/W, посылаемого ведущим. Ведомое устройство будет принимать или передавать бит на каждом тактовом импульсе от ведущего. Есть до девяти тактовых импульсов (8 для данных и один для ACK), когда ведомый принимает/передает данные от/для ведущего устройства. Бит I2CI устанавливает каждый раз, когда от/для ведомого принят/передан правильный байт данных.

Помните, что если ведомое устройство является передатчиком, ведущий должен сигнализировать окончание приема посылкой NACK после получения последнего байта. После этого ведомое устройство освобождает линию SDATA, позволяя ведущему сгенерировать условие окончания передачи.

Условие STOP (завершение передачи)

Последовательность передачи данных завершается условием STOP. Оно определяется переходом линии SDATA с низкого на высокий уровень, когда линия SCLOCK находится на высоком уровне (Рис.2б).

Условие STOP всегда генерируется ведущим устройством. Ведущий посылает условие STOP в том случае, если решает, что передача данных завершена, или получает от ведомого сигнал NACK. Получение условия STOP сбрасывает ведомое устройство в режим ожидания адреса.

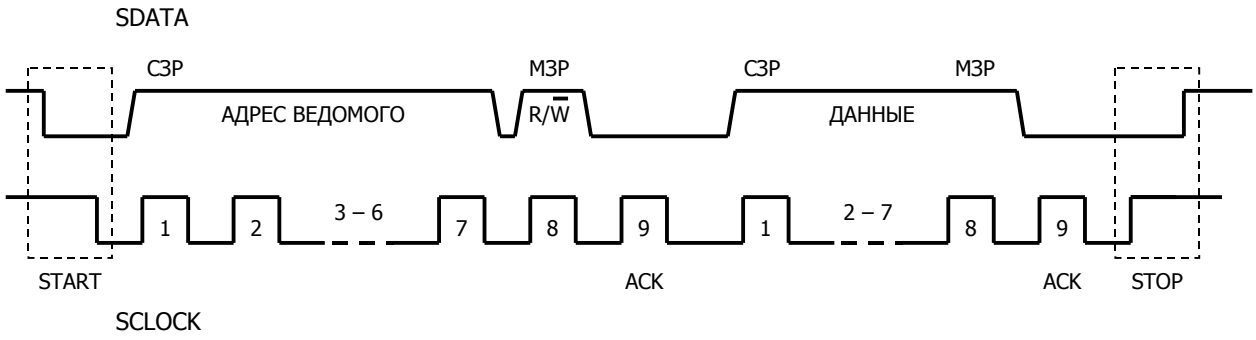


Рис.4. Типичная последовательность передачи по интерфейсу I2C.

1.3. 7-БИТНАЯ АДРЕСАЦИЯ ВЕДОМОГО УСТРОЙСТВА

Ведущий посылает 7-битный адрес ведомого устройства в виде семи старших разрядов первого передаваемого байта после условия START (восьмой бит является битом R/W). Ведомый сравнивает только первые семь разрядов со своим собственным адресом. Для получения законченного байта, ведомый добавляет ноль в позицию старшего разряда. Результат довершения сравнивается со значением в регистре I2CADD.

Поскольку ведомое устройство не осуществляет все манипуляции с адресом, как описано выше, автоматически, ведущий должен посылать подходящий адрес, например: для выбора ведомого устройства с адресом 44h (значение регистра I2CADD на ведомом MicroConverter), ведущий должен послать значение 88h (режим ведущий передает, ведомый принимает) или 89h (режим ведущий принимает, ведомый передает) в первом байте после условия START. Этот пример показан на Рис.5.

Рис.5. Процедура получения адреса ведомым устройством

1.4 РЕАЛИЗАЦИЯ I2C НА MICROCONVERTER

I2CADD
Содержит 7-разрядный адрес устройства (по умолчанию 55h).

ПРИМЕЧАНИЕ:
Этот SFR используется только в режиме ведомого. Обратитесь к разделу 1.3, который описывает адресацию ведомого устройства.

I2CDAT
В режиме ведомого-приемника принятые с линии SDATA данные записываются в этот регистр. После успешной операции приема данные могут быть прочитаны следующим образом:

```
MOV A, I2CDAT
```

В режиме ведомого-передатчика запись в этот регистр подготавливает данные для передачи по линии SDATA под управлением ведущего устройства, например:

```
MOV I2CDAT, #60h
```

записывает значение 60h на линию SDATA, которое будет передано по тактовому импульсу от ведущего устройства.

ПРИМЕЧАНИЕ:
Для ADuC812S/ADuC816/ADuC824 чтение или запись регистра I2CDAT автоматически очищает бит прерывания I2CI. Повторное обнуление этого бита вызовет у контроллера I2C состояние «потери». Для ADuC812 этот бит должен очищаться программным способом.

I2CCON
Содержит биты управления и состояния для операций в режимах ведущего/ведомого (см. таблицу 1).

Таблица 1. Описание битов регистра I2CCON.

Мнемоника	Описание
MDO	Software Master Data Out (только ведущий) Этот бит используется для программной реализации I2C интерфейса ведущего-передатчика. Данные, записанные в этот бит, будут выведены на линию SDATA, если бит MDE установлен.
MDE	Software Master Data Out Enabled (только ведущий) Этот бит используется для программной реализации I2C интерфейса ведущего-передатчика. Установка этого бита переводит вывод SDATA в режим вывода (TX). Очистка этого бита переводит вывод SDATA в режим ввода (RX).
MCO	Software Master Clock Out (только ведущий) Этот бит используется для программной реализации I2C интерфейса ведущего-передатчика. Данные, записанные в этот бит, будут выведены на линию SCLOCK.
MDI	Software Master Data In (только ведущий) Этот бит используется для программной реализации I2C интерфейса ведущего-передатчика. Данные на выводе SDATA захватываются по SCLOCK в этот бит, если бит MDE очищен.

I2CM	I2C Mode Установка этого бита переводит устройство в режим программной реализации ведущего, очистка в режим аппаратной реализации ведомого.
I2CRS	I2C Serial Port Reset (только ведомый) Установка этого бита вызывает сброс I2C интерфейса.
I2CTX	I2C Transmission Direction Status (только ведомый) Этот бит показывает направление передачи. Он установлен, если ведущее устройство передает данные ведомому, и очищен, если ведущее устройство записывает данные на ведомое. Этот бит автоматически загружается с битом R/W после адреса ведомого устройства и условия START.
I2CI	I2C Interrupt Flag (только ведомый) Флаг прерывания для последовательного порта I2C. Этот бит устанавливается после передачи или приема байта данных.

1.5 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ РЕЖИМА ВЕДУЩЕГО

Режим ведущего I2C интерфейса MicroConverter реализуется программно, т.е. пользователь должен программировать MicroConverter для побитного обмена по линиям SDATA и SCLOCK. Режим ведущего выбирается установкой бита I2CM в регистре I2CCON.

Для передачи данных по линии SDATA должен быть предварительно установлен бит MDE, чтобы включить функцию вывода SDATA на выход. Бит MDO в регистре I2CCON является битом Вывода Данных (Data Out). Драйвер управления выводом SDATA будет держать линию либо на высоком, либо на низком уровне, в зависимости от того установлен или очищен бит MDO.

Бит MCO в регистре I2CCON является битом Вывода Тактовых Импульсов. Драйвер управления выводом в режиме ведущего всегда включен и держит линию SCLOCK либо на низком, либо на высоком уровне, в зависимости от того установлен или очищен бит MCO.

ПРИМЕЧАНИЕ:
На ADuC812, в отличие от ADuC812S/ADuC816/ADuC824, нет повышения напряжения на выходе на линиях SDATA/SCLOCK, поэтому пользователь должен реализовать внешнее повышение напряжения для удерживания линии на высоком уровне.

Для приема данных должен быть очищен бит MDE, чтобы отключить драйвер управления выводом SDATA. Программа используется для переключения бита MCO (послать тактовый импульс) и чтения состояния линии SDATA через бит MDI. Программа должна контролировать биты MDO, MCO и MDE, соответственно, для генерации условия START, адреса ведомого устройства, битов подтверждения, байтов данных и условия STOP. Данные захватываются в бит MDI на нарастающем фронте SCLOCK только в том случае, если бит MDE очищен. Для генерации сигналов приема и передачи можно использовать функции из примера реализации ведущего устройства (I2Cmstr.asm).

ПРИМЕЧАНИЕ:
- Невозможно прочитать состояние линии SDATA (бит MDE) до тех пор, пока MDE на низком уровне (пока он соединен с другим выводом порта).
- Невозможно прочитать состояние линии SCLOCK (пока она соединена с другим выводом порта).
- Поскольку Ведущий реализован программно, прерывания происходить не будут.

1.6 АППАРТНАЯ РЕАЛИЗАЦИЯ РЕЖИМА ВЕДОМОГО

Режим ведомого I2C интерфейса MicroConverter реализуется аппаратно. По умолчанию устройство находится в режиме ведомого (в отличие от режима SPI). Режим I2C активизируется очисткой бита SPE в регистре SPICON (по умолчанию SPE=0). I2C адрес хранится в регистре I2CADD. Передаваемые или принятые данные хранятся в регистре I2CDAT.

Режим ведущего выбирается очисткой бита I2CM в регистре I2CCON. Состояние по умолчанию подразумевает ожидание условия START. При определении правильного условия START с последующим правильным адресом и битом R/W MicroConverter устанавливает бит I2CI.

Периферия I2C не генерирует прерывание до тех пор, пока пользователь не настроит бит I2C прерывания в регистре IE2/IEIP2 и бит EA в регистре IE:

```
; Разрешить I2C прерывания для ADuC812
MOV IE2,#01h ;   разрешить прерывание I2C
SETB EA
; Разрешить I2C прерывания для ADuC812S/ADuC816/ADuC824
MOV IEIP2,#01h ; разрешить прерывание I2C
SETB EA
```

ПРИМЕЧАНИЕ:
Для ADuC812 является важным, чтобы бит I2CI был очищен за прерывание только один раз. Если пользователь попробует очистить бит I2CI более одного раза, контроллер I2C перейдет в режим «потери».

```
CLR I2CI ; очистить бит прерывания для ADuC812
```

Для ADuC812S/ADuC816/ADuC824 реализована автоматическая очистка бита I2CI, поэтому он обнуляется в регистре I2CDAT при любом обращении чтения или записи.

```
MOV I2CDAT, A ; I2CI очищается при передаче (812S/816/824)
MOV A, I2CDAT ; I2CI очищается при приеме (812S/816/824)
```

В случае прерывания, PC счетчик перейдет на вектор по адресу 003Bh. Для первого байта, при переходе в программу обработки прерывания в регистре I2CDAT будут хранится 7-разрядный адрес и бит R/W.

Бит I2CTX содержит бит R/W, переданный ведущим устройством. Если он установлен, ведомое устройство передает данные путем записи в регистр I2CDAT. Если бит очищен, ведомый получает последовательный байт через регистр I2CDAT.

Ведомое устройство будет удерживать линию SCLOCK на низком уровне до тех пор, пока программно не будет очищен бит I2CI. Это «растягивание тактовой частоты» («clock stretching») описано в разделе 1.7. Оно гарантирует, что ведущее устройство не передаст следующий байт данных до тех пор, пока ведомый не будет готов.

Бит прерывания I2CI устанавливается каждый раз, когда получен или передан законченный байт данных с последующим ACK. Если байт заканчивается NACK, прерывание генерироваться НЕ будет. MicroConverter будет вызывать прерывания для каждого законченного байта до тех пор, пока не будет принято условие STOP или интерфейс будет сброшен.

ПРИМЕЧАНИЕ:
Отсутствует возможность различить прерывание по приему условия START+адреса и прерывания по приему байта данных. Для того чтобы отличить эти прерывания, пользователь должен следить за последовательностью связи. При приеме условия STOP, интерфейс сбрасывается в состояние, в котором ждет адреса («простой»).

1.7 ВОЗМОЖНОСТИ I2C, НЕ РЕАЛИЗОВАННЫЕ НА MICROCONVERTER

Режим ведомого

- MicroConverter (ведомый) не отвечает на Глобальный Адрес Вызова (0000000).

- MicroConverter не может обрабатывать повторяющееся условие START. Повторяющееся условие START позволяет ведущему адресовать другое ведомое устройство (или изменить направление передачи) без необходимости передачи условия STOP. Для MicroConverter условие STOP должно быть передано ведомому устройству в случае завершения сеанса связи или изменения направления передачи. Процесс передачи продолжается в том случае, если повторяющееся условие START является новым.

- Поскольку MicroConverter не может обрабатывать повторяющиеся условия START, он не поддерживает в режиме ведомого «Стартовый Байт» (Start Byte). Однако, это можно реализовать программно в режиме ведущего.

- MicroConverter не поддерживает 10-битную адресацию.

Режим ведущего

- <u>«Растягивание тактового частоты»</u>
  «Растягивание тактового частоты» используется для интерфейса быстрого ведущего устройства и медленного ведомого. Когда ведомое устройство получает правильный адрес или байт данных, оно автоматически переводит линию SCLOCK на низкий уровень. После того, как в программе будет очищен бит I2CI, ведомый отпускает линию SCLOCK. Для корректной реализации ведущий должен иметь возможность чтения состояния линии SCLOCK и задержать передачу новых данных.

  На MicroConverter ведущее устройство может прочитать состояние линии SCLOCK через I2C интерфейс. Для такого чтения линия SCLOCK должна быть соединена через цифровой вывод (любой вывод порта), через который ведущий будет видеть текущее состояние линии.

- <u>Разрешение конфликтных ситуаций</u>
  В приложениях с несколькими ведущими устройствами необходимо предотвратить ситуацию, когда два ведущих могут записать данные на шину в одно и то же время. Для проверки линии SDATA используется организация доступа к одной шине данных (арбитраж). Она заключается в том, что одно ведущее устройство держит линию SDATA на высоком уровне и другие ждут момента, когда шина освободится.

  На MicroConverter ведущее устройство не может прочитать состояние линии SDATA через I2C интерфейс. Для такого чтения линия SDATA должна быть соединена через цифровой вывод (любой вывод порта).

1.8 РАЗЛИЧИЯ В РЕАЛИЗАЦИЯХ I2C НА MICROCONVERTER

ADuC812

- нет повышения напряжения на выводах SDATA/SCLOCK
- бит I2CI должен очищаться в программе с помощью инструкции `CLR I2CI`
- бит разрешения прерывания I2C (ESI) и бит приоритета прерывания I2C (PSI) находятся в регистре IE2 (адрес SFR A9h) и регистре IP (адрес SFR B8h), соответственно.

ADuC812S/ADuC816/ADuC824

- есть повышения напряжения на выводах SDATA/SCLOCK. Оно работает только в режиме I2C. Для систем с несколькими ведомыми может потребоваться внешнее повышение напряжения.
- бит I2CI автоматически очищается после завершения чтения или записи в регистр I2CDAT. Очистка этого бита переведет контроллер I2C в режим «потери», поэтому нет необходимости проверять состояние этого бита.
- и бит разрешения прерывания I2C (ESI), и бит приоритета прерывания I2C (PSI) находятся в регистре IEIP2 (адрес SFR A9h) в IEIP2.0 IEIP2.4, соответственно.

2.0 ПРИМЕР КОДА ДЕЙСТВИЙ I2C ВЕДУЩЕГО

Этот раздел описывает программу для ведущего устройства (файл `I2Cmstr.asm`). Программа должна работать вместе с программой ведомого `I2Cslave.asm` на двух отдельных оценочных платах. Пример кода показывает как программируется I2C интерфейс MicroConverter для режима ведущего.

Программа ведущего посылает условие START, адрес ведомого устройства и бит R/W (устанавливает режим ведущего-приемника). Далее она принимает один байт от ведомого и передает NACK и условие STOP. NACK показывает ведомому, что ведущий принял последний байт данных. После этого ведущий снова посылает условие START, адрес ведомого и бит R/W (на этот раз он очищен и показывает, что ведущий будет передавать данные). После передачи байта он проверяет получение подтверждения ACK и посылает условие STOP. В завершение ведущий передает принятый байт через UART на ПК, где он может быть просмотрен с использованием программы hyperterminal. Программа ведущего переходит в начало и снова принимает от ведомого байт данных.

Если символ принят через UART, то ведущий посылает его ASCII значение ведомому. Нажатие кнопки INT0 на оценочной плате вызовет увеличение выводимого на ведомое устройство значения.

Блок-схема программы ведущего показана на Рис.6. Подпрограммы RCVDATA и SENDDATA показаны на Рис.7а и Рис.7б, соответственно.

Программные переменные

SLAVEADD: содержит адрес ведомого устройства. В приведенном примере адрес ведомого – 44h. Это что означает, что ведущий передает значение 88h (или 89h), как описано в разделе 1.3.

OUTPUT: содержит значение, передаваемое ведомому, инициализируется 0h.

INPUT: содержит значение, принимаемое от ведомого, которое будет передано через UART.

NOACK: флаг NOACK устанавливается, если был принят NACK, когда ожидался ACK.

ERR: флаг ERR устанавливается, если где-либо в программе был установлен NOACK и позволяет очистить в программе флаг NOACK. Если в конце программы флаг ERR установлен, на UART выводится сообщение об ошибке.

Описание кода

Приведенное описание желательно читать вместе с просмотром блок-схем на Рис.6, Рис.7а и Рис7б.

НАСТРОЙКА КОНФИГУРАЦИИ: конфигурируются UART и Внешнее Прерывание 0 (INT0).
UART: UART настраивается на скорость передачи 9600 бод.
Внешнее Прерывание 0: устанавливаются EX0/TX0 для разрешения прерывания INT0 по фронту сигнала. Устанавливается EA для разрешения прерываний.

ИНИЦИАЛИЗАЦИЯ: инициализируются регистры и флаги I2C.
I2CCON = A8h => режим ведущего
Отключается драйвер выхода на SDATA, линия SCLOCK переводится на высокий уровень.
OUTPUT = 0. Первый байт для передачи инициализируется значением 0h.

ПРИМЕЧАНИЕ:
Поскольку выбран режим ведущего, нет необходимости включать прерывание I2C или заносить значение в регистр I2CADD.

ПРИЕМ: прием байта данных производится следующим образом (см. Рис.7а)

1) передается условие START
2) передается адрес ведомого устройства (вместе с битом R/W, установленным для приема)
3) проверяется ACK
4) если получен NACK, передается условие STOP и устанавливается флаг ERR
5) если получен ACK, ведомому устройству передаются 8 тактовых импульсов и после каждого их них считывается бит MDI. Полученный таким образом байт записывается в INPUT.
6) передается NACK, что означает завершение сеанса связи.
7) передается условие STOP
8) если получен NACK, устанавливается флаг ERR

ПЕРЕДАЧА: передача байта данных производится следующим образом (см. Рис.7б)

1) передается условие START
2) передается адрес ведомого устройства (вместе с битом R/W, установленным для передачи)
3) проверяется ACK
4) если получен NACK, передается условие STOP и устанавливается флаг ERR
5) если получен ACK, ведомому устройству передаются 8 тактовых импульсов, во время каждого из них в бит MDO регистра I2CCON записывается соответствующий разряд из OUTPUT.
6) проверяется ACK
7) если получен NACK, устанавливается флаг ERR
8) передается условие STOP

ПРОВЕРКА ERR: проверяется флаг ERR, чтобы узнать, была ли ошибка. В случае наличия ошибки, через UART на ПК передается соответствующее сообщение.

ПЕРЕДАЧА НА ПК: на ПК через UART передается принятый байт.

ЗАДЕРЖКА: задержка используется для того, чтобы дать визуальное представление о работе программы, во время которой буден виден мигающий светодиод.

ПЕРЕКЛЮЧЕНИЕ СВЕТОДИОДА: каждое переключение светодиода показывает, что от ведомого принят байт данных, и ведущий передал байт обратно на ведомое устройство.

ПРОВЕРКА RI: если ведущим принят байт данных через UART, то он загружается в OUTPUT для последующей передачи нового значения OUTPUT. Байт может быть передан с ПК на ведущее устройство через последовательный порт, нажатием кнопки клавиатуры. В этом случае ASCII представление этого символа будет записано в OUTPUT.

ОБРАБОТКА ПРЕРЫВАНИЯ INT0: нажатие кнопки INT0 на оценочной плате (вызывающей прерывание INT0) приведет к увеличению значения OUTPUT на единицу.

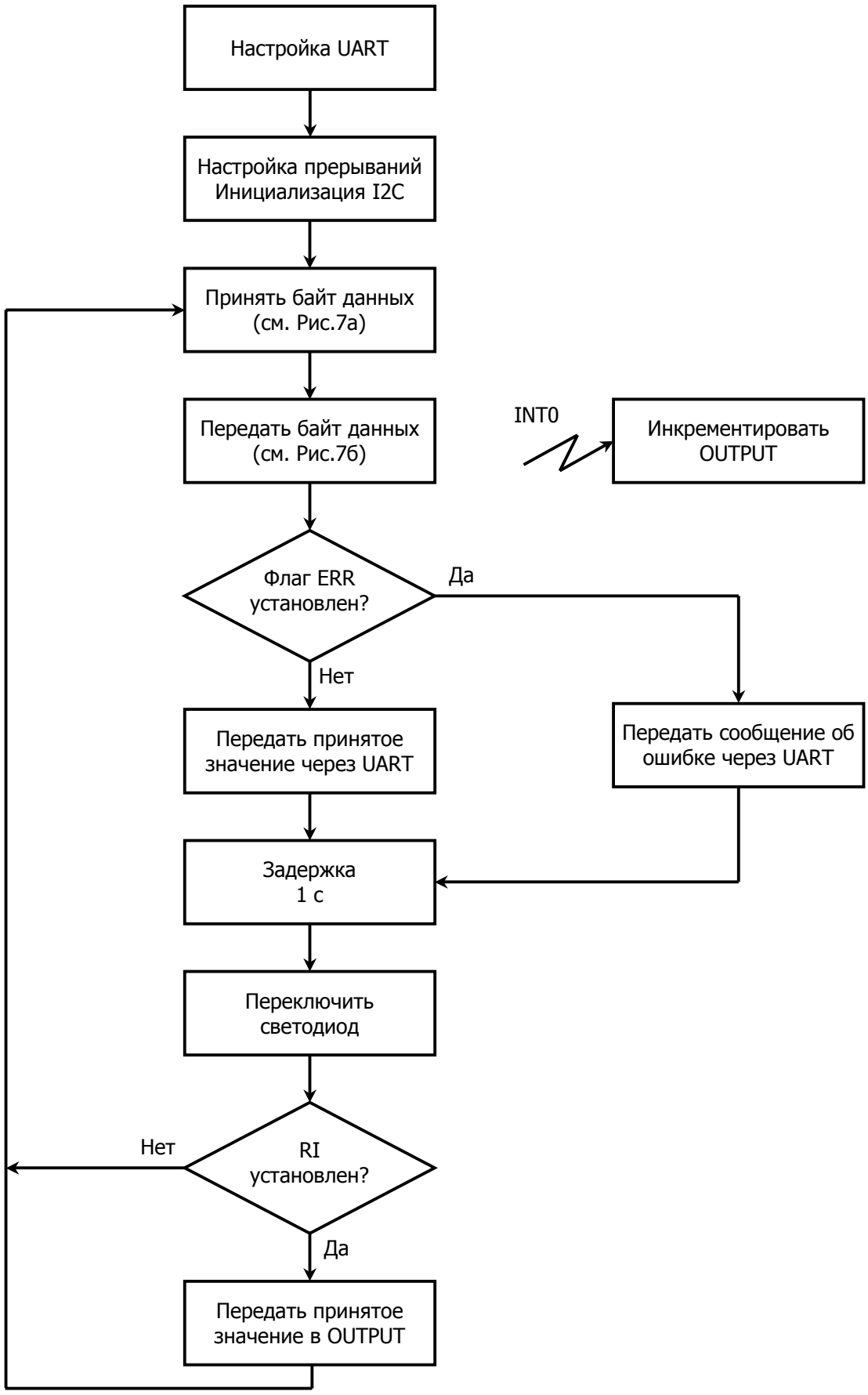
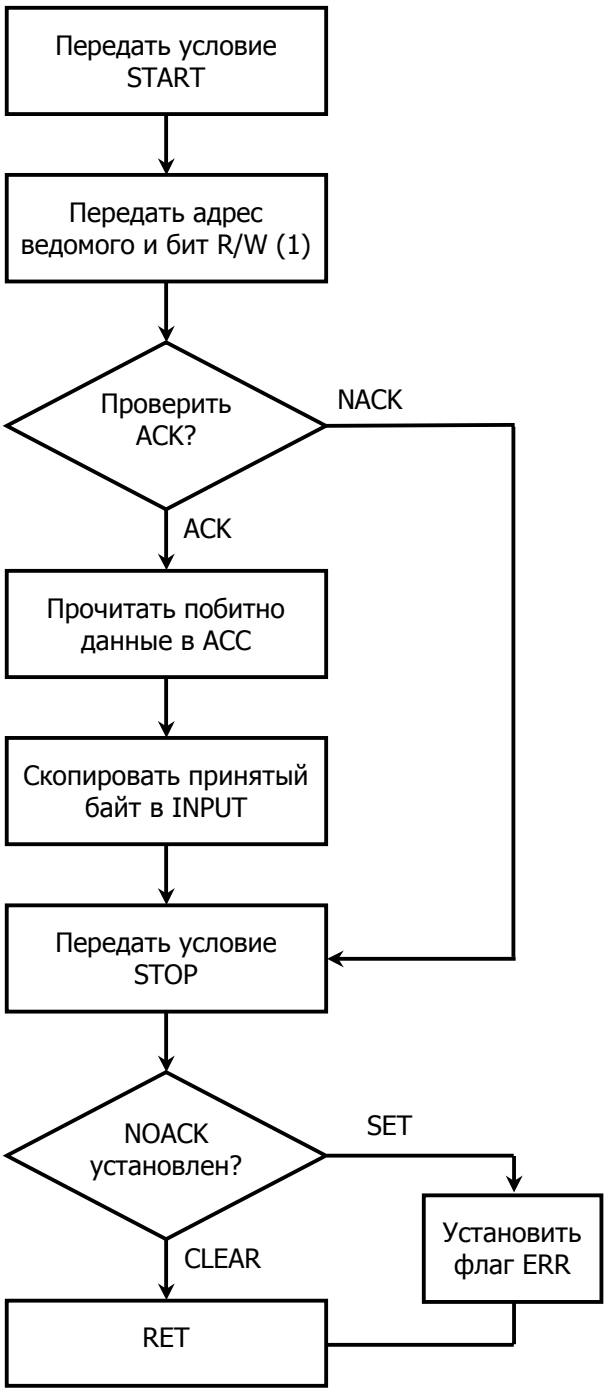


Рис.6. Блок-схема программы ведущего (I2Cmstr.asm).

ПОДПРОГРАММА RCVDATA



ПОДПРОГРАММА SENDDATA

Рис.7а/7б. Блок-схема подпрограмм RCVDATA и SENDDATA.

2.1 ПРИМЕР КОДА ДЕЙСТВИЙ I2C ВЕДОМОГО

Этот раздел описывает программу для ведущего устройства (файл `I2Cslave.asm`). Программа должна работать вместе с программой ведущего `I2Cmstr.asm` на двух отдельных оценочных платах. Пример кода показывает как программируется I2C интерфейс MicroConverter для режима ведомого.

Программа ведомого ждет прерывания I2C. После прерывания она проверяет, какой режим выбран ведущим – прием или передача. Если выбран режим передачи, то ведомый записывает данные в регистр I2CDAT и работает передатчиком. Если выбран режим приема, то ведомый ждет следующего прерывания I2CI и принимает данные. После завершения приема ведомый передает принятый байт на ПК через UART, где он может быть увиден в программе hyperterminal. Далее программа переходит на начало и ждет следующего I2C прерывания.

Если через UART принят символ, то ведомый сохраняет новое значение для выходного байта, передаваемого ведущему. Нажатие кнопки INT0 на оценочной плате вызовет инкрементирование значения в этом байте.

Блок-схема программы ведомого показана на Рис.8.

Программные переменные

OUTPUT: содержит значение, передаваемое ведущему, инициализируется 0h.

INPUT: содержит значение, принимаемое от ведущего, которое будет передано через UART.

GO: флаг GO используется для индикации ожидания прерывания I2C.

FIRST: флаг FIRST используется в режиме приема, он устанавливается для первого прерывания, чтобы программа читала только байт данных (без адреса).

Описание кода

Приведенное описание желательно читать вместе с просмотром блок-схемы на Рис.8.

НАСТРОЙКА КОНФИГУРАЦИИ: конфигурируются UART и Внешнее Прерывание 0 (INT0).
UART: UART настраивается на скорость передачи 9600 бод.
Внешнее Прерывание 0: устанавливаются EX0/TX0 для разрешения прерывания INT0 по фронту сигнала. Устанавливается EA для разрешения прерываний.

ИНИЦИАЛИЗАЦИЯ: инициализируются регистры и флаги I2C.
I2CCON = 00h => режим ведомого
I2CDAT = 44h => режим ведомого
Инициализация OUTPUT = 30h.

ОЖИДАНИЕ I2CI: ожидается прерывание I2C.

ПРОВЕРКА I2CTX: после того, как произошло прерывание I2C, проверяется бит I2CTX. Он показывает ведомому, какой режим выбран ведущим – прием или передача. Если выбран режим передачи, в регистр I2CDAT заносится значение из OUTPUT и программа ждет следующего прерывания. В режиме передачи программа ждет второго прерывания (после первого в I2CDAT записан адрес). После второго прерывания содержимое регистра I2CDAT записывается в INPUT.

ПЕРЕКЛЮЧЕНИЕ СВЕТОДИОДА: каждое переключение светодиода показывает, что ведомый принял или передал байт данных ведущему.

ПЕРЕДАЧА НА ПК: на ПК через UART передается принятый байт.

ПРОВЕРКА RI: если ведущим принят байт данных через UART, то он загружается в OUTPUT для последующей передачи нового значения OUTPUT ведущему. Программа переходит в режим ожидания.

ОБРАБОТКА ПРЕРЫВАНИЯ INT0: нажатие кнопки INT0 на оценочной плате (вызывающей прерывание INT0) приведет к увеличению значения OUTPUT на единицу.

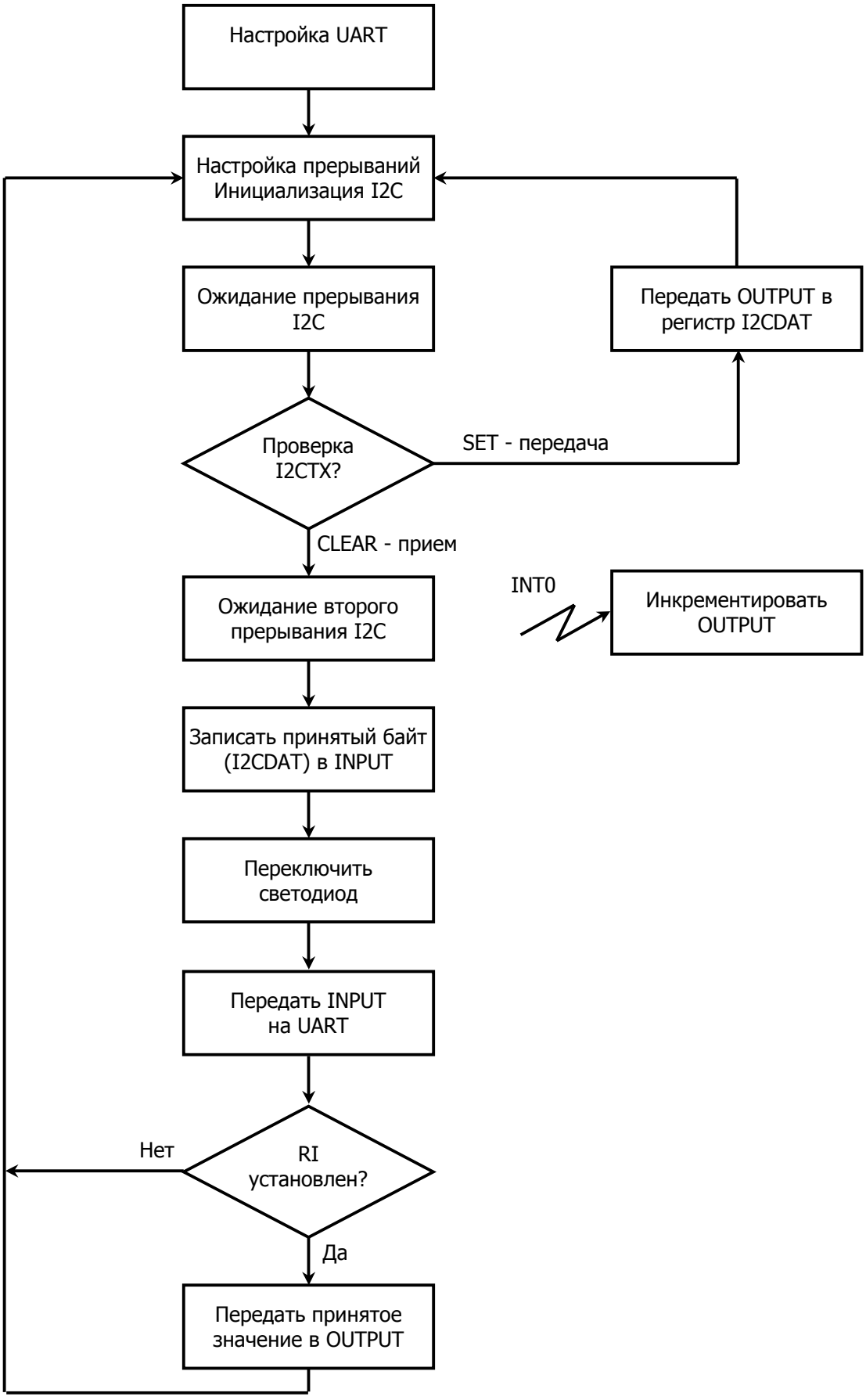


Рис.8. Блок-схема программы ведомого (I2Cslave.asm).