Современные 32-разрядные ARM-микроконтроллеры серии STM32: последовательный интерфейс I²C

Олег Вальпа

В статье приведено описание последовательного интерфейса I²C 32-разрядных ARM-микроконтроллеров серии STM32 от компании STMicroelectronics. Рассмотрены архитектура, состав и назначение регистров конфигурирования интерфейса, а также приведены примеры программ его использования.

ВВЕДЕНИЕ

Интерфейс I²C, или IIC получил свою аббревиатуру от английских слов Inter-Integrated Circuit и представляет собой последовательную шину, состоящую из двух двунаправленных линий связи с названием SDA и SCL, как сокращение от слов Serial Data Address и Serial Clock. Он обеспечивает обмен данными между микроконтроллером и различными периферийными устройствами, такими как АЦП, ЦАП, микросхемы памяти, другие микроконтроллеры и микросхемы. Схема подключения устройств по интерфейсу I²C показана на рисунке 1.

Стандарт на интерфейс I²C был разработан фирмой Philips в начале 1980-х годов. Согласно этому стандарту, интерфейс имел 7-разрядный адрес. Он позволял обращаться к 127 устройствам на скорости до 100 кбит/с. В дальнейшем интерфейс получил свое развитие и стал 10-разрядным, позволяющим обращаться к 1023 устройствам на скорости до 400 кбит/с. Максимальное допустимое количество микросхем, подсоединенных к одной шине, ограничивается максимальной емкостью шины в 400 пФ. Версия стандарта 2.0, выпущенная в 1998 году, представила высокоскоростной режим работы со скоростью до 3.4 Мбит/с с пониженным энергопотреблением. Версия 2.1 2001 года включает в себя лишь незначительные доработки.

ОПИСАНИЕ ИНТЕРФЕЙСА I²C

Микроконтроллер STM32 [1] включает в свой состав интерфейс I²C, который отличается своей развитостью. Он допускает несколько ведущих устройств на шине и поддерживает высокоскоростной режим. Кроме того, в микроконтроллере STM32 интерфейс I²C можно использовать для широкого спектра приложений, включая генерацию и верификацию контрольной суммы. С ним также можно работать по протоколам SMBus (System Management Bus) и PMBus (Power Management Bus). Большинство моделей STM32 включают в свой состав два интерфейса I²C с именами I2C1 и I2C2.

Интерфейс может работать в одном из следующих четырех режимов:

- Slave transmitter (ведомый передатчик);
- Slave receiver (ведомый приемник);
- Master transmitter (ведущий передатчик);
- Master receiver (ведущий приемник).

По умолчанию интерфейс работает в режиме «Ведомый» и автоматически переключается на «Ведущий» после генерирования старт-условия. Переключение с «Ведущего» на «Ведомый» происходит при потере арбитража или после генерирования стоп-условия, что позволяет работать нескольким «Ведущим» микроконтроллерам в одной системе поочередно.

В режиме «Ведущий» I²C инициирует обмен данными и генерирует тактовый сигнал. Передаче последовательных данных всегда предшествует старт-условие, а завершается обмен всегда стоп-условием. Оба этих условия генерируются в режиме «Ведущий» программно.

В режиме «Ведомый» I²C способен распознать свой собственный адрес (7 или 10 бит) и адрес общего вызова. Определение наличия адреса общего вызова можно включить или отключить программно.

Адрес и данные передаются 8-битными посылками, старшим битом вперед. Первый байт, следующий за старт-условием, содержит адрес (один байт в 7-битном режиме и два байта в 10-битном режиме). Адрес всегда передается в режиме «Ведущий».

За 8 тактами передачи байта данных следует 9-й такт, в течение которого

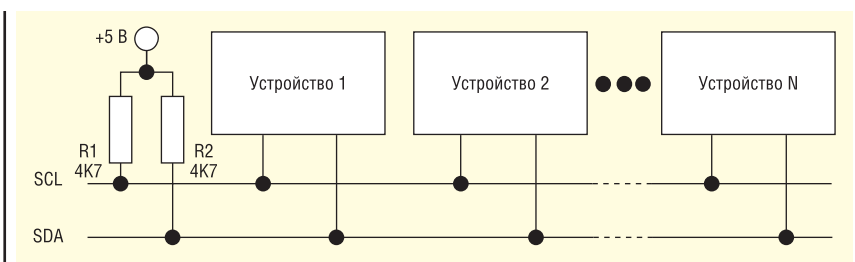

Рис. 1. Схема подключения устройств по интерфейсу I²C

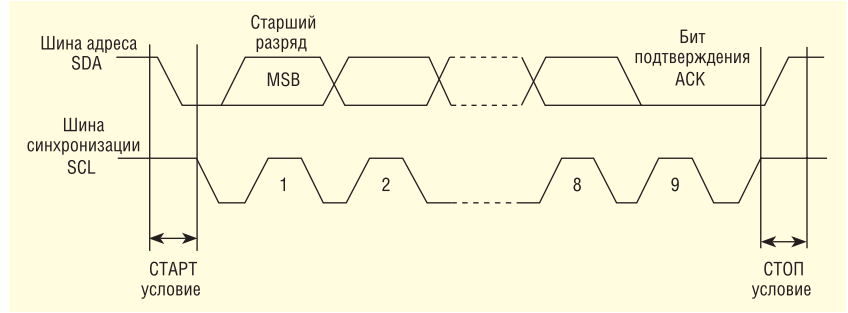

Рис. 2. Временна́я диаграмма одной посылки интерфейса I²C

приемник должен послать бит уведомления ACK, получивший свое название от слова ACKnowledge. На рисунке 2 приведена временна́я диаграмма одной посылки интерфейса I²C.

Наличие уведомления в ответе можно программно включить или отключить. Размерность адреса интерфейса I²C (7 бит или 10 бит и адрес общего вызова) можно выбрать программно.

АРХИТЕКТУРА БЛОКА ИНТЕРФЕЙСА I²C

Функциональная схема блока интерфейса I²C для микроконтроллера STM32 приведена на рисунке 3.

Регистр сдвига на этой схеме представляет собой основной регистр, через который передаются и принимаются данные. Передаваемые данные предварительно записываются в регистр данных, после чего через регистр сдвига последовательно транслируются в линию связи SDA. Принимаемые по этой же линии связи данные накапливаются в регистре сдвига, а затем перемещаются в регистр данных. Таким образом, интерфейс может передавать и принимать данные только поочередно.

Кроме того, регистр сдвига аппаратно подключен к компаратору, который позволяет сравнивать принятый адрес с адресными регистрами и, таким образом, определять, для кого предназначен очередной блок данных.

Узел управления частотой позволяет формировать сигнал синхронизации SCL в роли ведущего и синхронизироваться от этого сигнала в качестве ведомого устройства. Регистр CCR обеспечивает программную настройку данного узла. Блок интерфейса подключен к выходу PCLK1 шины APB1 через два предварительных делителя. Микроконтроллер поддерживает два режима обмена: стандартный (Standard Speed) — до 100 кГц, и быстрый (Fast Speed) — до 400 кГц. В зависимости от режима обмена частота тактирования модуля должна быть не менее 2 МГц в стандартном режиме и не менее 4 МГц в быстром режиме.

Блок вычисления позволяет аппаратно вычислять контрольную сумму блока данных и сохранять ее в регистре PEC.

Управление блоком интерфейса I²C, а также формирование флагов событий и прерываний выполняется узлом логики управления. Он же позволяет обслуживать запросы ПДП и формировать сигнал ACK. Связь этого блока с микроконтроллером осуществляется программно с помощью регистров управления CR1, CR2 и регистров состояния SR1, SR2.

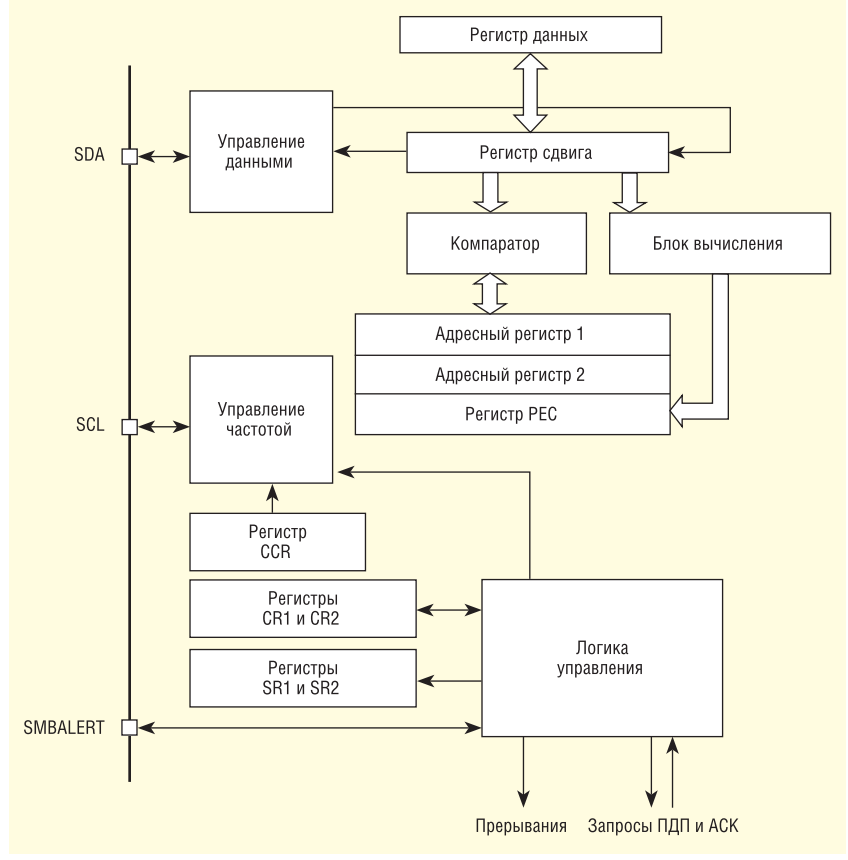

Рис. 3. Функциональная схема блока интерфейса I²C

ПРЕРЫВАНИЯ ОТ I²C

Интерфейс I²C имеет аппаратную организацию, способную формировать запросы на прерывание в зависимости от режима работы и текущих событий. В таблице 1 приведены запросы на прерывание от интерфейса I²C.

ОПИСАНИЕ РЕГИСТРОВ

Для работы с интерфейсом I²C в микроконтроллере STM32 имеются специальные регистры. Карта этих регистров с названием входящих в них разрядов представлена в таблице 2.

Рассмотрим регистры, необходимые для работы интерфейса I²C. К ним относятся:

- I²C_CR1 — управляющий регистр 1;
- I²C_CR2 — управляющий регистр 2;
- I²C_OAR1 — регистр собственного адреса 1;
- I²C_OAR2 — регистр собственного адреса 2;
- I²C_DR — регистр данных;
- I²C_SR1 — статусный регистр 1;
- I²C_SR2 — статусный регистр 2;

- I²C_CCR – регистр управления тактовым сигналом;
- I²C_TRISE — регистр параметра TRISE.

Некоторые разряды этих регистров используются для работы в режиме SMBus.

Регистр I²C_CR1 является первым управляющим регистром интерфейса I²C. Он имеет следующие управляющие разряды:

- разряд 15 SWRST — обеспечивает программный сброс шины I²C;
- разряд 14 — зарезервирован;
- разряд 13 SMBus — формирует сигнал тревоги в режиме SMBus;
- разряд 12 PEC — служит для функции проверки ошибки пакета (Packet Error Checking);
- разряд 11 POS — служит для анализа сигналов ACK или PEC при приеме;
- разряд 10 ACK — возвращает бит уведомления ACK после приема корректного байта адреса или данных;
- разряд 9 STOP — служит для формирования и анализа стоп-условия;
- разряд 8 START — служит для формирования и анализа старт-условия;
- разряд 7 NOSCTETCH — отключает растяжку такта в режиме ведомого;
- разряд 6 ENGC — разрешает общий вызов;
- разряд 5 ENPEC — разрешает сигнал PEC;
- разряд 4 ENARP — разрешает сигнал ARP;
- разряд 3 SMBTYPE — назначает тип интерфейса в качестве ведущего или ведомого для режима SMBus;
- разряд 2 — зарезервирован;
- разряд 1 SMBUS — переключает режимы I²C и SMBus;

Таблица 1. Запросы на прерывание от интерфейса I²C

Событие	Флаг события	Разрешающий бит управления
Послано старт-условие	SB	ITEVFEN
Ведущий послал адрес или ведомый получил свой адрес	ADDR	
Ведущий послал заголовок 10 бит	ADD10	
Ведомый получил стоп-условие	STOPF	
Передача байта закончена	BTF	
Буфер приема не пуст	RxNE	ITEVFEN и ITBUFEN
Буфер передачи пуст	TxE	
Ошибка шины	BERR	ITERREN
Потеря прав на шину ведущего	ARLO	
Ошибка уведомления	AF	
Переполнение/недостача данных	OVR	
Ошибка пакета	PECERR	
Ошибка превышения времени	TIMEOUT	

Таблица 2. Карта регистров интерфейса I²C

Сдвиг	Регистр	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	I²C_CR1	Резерв																SWRST	Резерв	ALERT	PEC	POS	ACK	STOP	START	NOSTRETCH	ENGC	ENPEC	ENARP	SMBTYPE	Резерв	SMBUS	PE
	Исх. значение																	0		0	0	0	0	0	0	0	0	0	0	0		0	0
0x04	I²C_CR2	Резерв																			LAST	DMAEN	ITBUFEN	ITEVTEN	ITERREN	Резерв		FREQ[5:0]					
	Исх. значение																				0	0	0	0	0			0	0	0	0	0	0
0x08	I²C_OAR1	Резерв																ADDMODE	Резерв					ADD[9:8]		ADD[7:1]							ADD0
	Исх. значение																	0						0	0	0	0	0	0	0	0	0	0
0x0C	I²C_OAR2	Резерв																								ADD2[7:1]							ENDUAL
	Исх. значение																									0	0	0	0	0	0	0	0
0x10	I²C_DR	Резерв																								DR[7:0]							
	Исх. значение																									0	0	0	0	0	0	0	0
0x14	I²C_SR1	Резерв																SMBALERT	TIMEOUT	Резерв	PECERR	OVR	AF	ARLO	BERR	TxE	RxNT	Резерв	STOPF	ADD10	BTF	ADDR	SB
	Исх. значение																	0	0		0	0	0	0	0	0	0		0	0	0	0	0
0x18	I²C_SR2	Резерв																PEC[7:0]								DUALF	SMBHOST	SMBDEFAULT	GENCALL	Резерв	TRA	BUSY	MSL
	Исх. значение																	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
0x1C	I²C_CCR	Резерв																		Резерв	CCR[11:0]												
	Исх. значение																	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
0x20	I²C_TRISE	Резерв																								TRISE[5:0]							
	Исх. значение																									0	0	0	0	1	0		

- разряд 0 PE — разрешает работу интерфейса.

Регистр I²C_CR2 является вторым управляющим регистром интерфейса I²C и имеет следующие управляющие разряды:

- разряды 15...13 — зарезервированы;
- разряд 12 LAST — используется в режиме ведущего приемника, чтобы позволить генерацию сигнала NACK по последнему принятому байту;
- разряд 11 DMAEN — разрешает запрос DMA;
- разряд 10 ITBUFEN — разрешает прерывания от буфера;
- разряд 9 ITEVTEN — разрешает прерывания от события;
- разряд 8 ITERREN — разрешает прерывания от ошибки;
- разряды 7 и 6 — зарезервированы;
- разряды 5...0 FREQ[5:0] — задают частоту работы шины.

Регистр I²C_OAR1 — первый регистр собственного адреса, включает в себя следующие разряды:

- разряд 15 ADDMODE — задает 7- или 10-разрядный режим адресации в качестве ведомого;
- разряды 14...10 — зарезервированы;
- разряды 9 и 8 ADD[9:8] — назначают 9 и 8 биты адреса при 10-битной адресации интерфейса;
- разряды 7...1 ADD[7:1] — назначают 7...1 биты адреса;
- разряд 0 ADD0 — назначает бит 0 адреса при 10-битной адресации интерфейса.

Регистр I²C_OAR2 — второй регистр собственного адреса, включает в себя следующие разряды:

- разряды 15...8 — зарезервированы;
- разряды 7...1 ADD[7:1] — назначают 7...1 биты адреса в режиме двойной адресации;
- разряд 0 ENDUAL — разрешает режим двойной адресации.

Регистр данных I²C_DR имеет 8 разрядов DR[7:0] для приема и передачи данных на шину I²C. В этот регистр данные записываются для передачи и читаются из него при приеме. Разряды 15...9 — зарезервированы.

Регистр I²C_SR1 — первый статусный регистр, и включает в себя следующие разряды:

- разряд 15 SMBALERT — сигнализирует о тревоге шины SMBus;
- разряд 13 — зарезервирован;
- разряд 14 TIMEOUT — оповещает об ошибке превышения времени для сигнала SCL;
- разряд 12 PECERR — свидетельствует об ошибке PEC при приеме;
- разряд 11 OVR — формируется при ошибке переполнения данных;
- разряд 10 AF — возникает в случае ошибки уведомления;
- разряд 9 ARLO — указывает на ошибку потери прав на шину;
- разряд 8 BERR — устанавливается при ошибке шины;
- разряд 7 TxE — оповещает, что регистр данных пуст;
- разряд 5 — зарезервирован;
- разряд 6 RxNE — информирует, что регистр данных не пуст;
- разряд 4 STOPF — детектирует стопусловие в режиме ведомого;
- разряд 3 ADD10 — устанавливается, когда ведущий послал первый байт адреса при 10-битной адресации;
- разряд 2 BTF — оповещает о завершении передачи байта;
- разряд 1 ADDR — устанавливается, если послан адрес в режиме ведущего или принят адрес в режиме ведомого;
- разряд 0 SB — устанавливается при генерации старт-условия в режиме ведущего.

Регистр I²C_SR2 — второй статусный регистр, включает в себя следующие разряды:

- разряды 15...8 PEC[9:8] — содержат контрольную сумму кадра;
- разряд 7 DUALF — является флагом двойной адресации в режиме ведомого;
- разряд 6 SMBHOST — устанавливается, когда принят заголовок SMBus Host в режиме ведомого;
- разряд 5 SMBDEFAULT — возникает, если принят адрес по умолчанию для SMBus-устройства в режиме ведомого;
- разряд 4 GENCALL — указывает, что принят адрес общего вызова в режиме ведомого;
- разряд 3 — зарезервирован;
- разряд 2 TRA — оповещает о режиме передачи/приема;
- разряд 1 BUSY — информирует, что шина занята;
- разряд 0 MSL — детектирует режим «Ведущий»/«Ведомый».

Регистр I²C_CCR — регистр управления тактовым сигналом, который включает в себя разряды:

- разряд 15 F/S — задает стандартную или быструю скорость для режима ведущего;
- разряд 14 DUTY — назначает скважность 2 или 16/9 в быстром режиме;
- разряды 13 и 12 — зарезервированы;
- разряды 11...0 CCR[11:0] — управляют тактовым сигналом для быстрой и стандартной скорости в режиме ведущего.

Регистр I²C_TRISE — регистр параметра TRISE, который включает в себя:

- разряды 15...6 — зарезервированы;
- разряды 5...0 TRISE[5:0] — определяют максимальное время фронта для быстрой и стандартной скорости в режиме ведущего. Данный параметр задает момент времени, по которому производятся выборка состояния линий.

Более подробное описание назначения всех регистров I²C и их разрядов можно найти на сайте www.st.com [2].

ПРОГРАММИРОВАНИЕ ИНТЕРФЕЙСА I²C

Рассмотрим практическую реализацию по использованию интерфейса I²C. Для этого можно воспользоваться стандартной библиотекой периферии микроконтроллера STM32. Для интерфейса I²C настройки режима, скорости и всего остального находятся в заголовочном файле и объявлены в виде структуры:

```
I2C_InitTypeDef:typedef struct{
uint32_t I2C_ClockSpeed;
uint16_t I2C_Mode;
uint16_t I2C_DutyCycle;
uint16_t I2C_OwnAddress1;
uint16_t I2C_Ack;
uint16_t I2C_
AcknowledgedAddress;
}I2C_InitTypeDef;
```

В этой структуре ее элементы имеют следующее назначение:

- uint32_t I²C_ClockSpeed — частота тактового сигнала, максимум — 400 КГц;
- uint16_t I²C_Mode — режим работы;
- uint16_t I²C_DutyCycle — настройки для работы в быстром режиме;
- uint16_t I²C_OwnAddress — собственный адрес устройства;
- uint16_t I²C_Ack — включено или нет использование бита подтверждения ACK;
- uint16_t I²C_AcknowledgedAddress — выбор формата адреса: 7 бит или 10 бит.

Рассмотрим процедуры инициализации и работы с интерфейсом I²C.

Листинг 1

```
GPIO_InitTypeDef gpio; // Создание структуры для портов ввода-вывода
I2C_InitTypeDef i2c; // Создание структуры для интерфейса I2C
void init_I2C1(void)
{
// Включить тактирование
RCC_APB1PeriphClockCmd(RCC_APB1Periph_I2C1, ENABLE);
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE);
// Инициализировать I2C
i2c.I2C_ClockSpeed = 100000;
i2c.I2C_Mode = I2C_Mode_I2C;
i2c.I2C_DutyCycle = I2C_DutyCycle_2;
// Задать адрес=0x12
i2c.I2C_OwnAddress1 = 0x12;
i2c.I2C_Ack = I2C_Ack_Disable;
i2c.I2C_AcknowledgedAddress = I2C_AcknowledgedAddress_7bit;
I2C_Init(I2C1, &i2c);
// Назначить выводы интерфейса
gpio.GPIO_Pin = GPIO_Pin_6 | GPIO_Pin_7;
```

Листинг 2

```
void I2C_StartTransmission(I2C_TypeDef* I2Cx, uint8_t
transmissionDirection, uint8_t slaveAddress)
{
// Ждать освобождения шины
while(I2C_GetFlagStatus(I2Cx, I2C_FLAG_BUSY));
// Сформировать старт-условие
I2C_GenerateSTART(I2Cx, ENABLE);
// Ждать установки бита
while(!I2C_CheckEvent(I2Cx, I2C_EVENT_MASTER_MODE_SELECT));
// Отправить адрес ведомому устройству
I2C_Send7bitAddress(I2Cx, slaveAddress, transmissionDirection);
// Если передача данных
if(transmissionDirection== I2C_Direction_Transmitter)
{while(!I2C_CheckEvent(I2Cx,
I2C_EVENT_MASTER_TRANSMITTER_MODE_SELECTED));}
// Если прием данных
if(transmissionDirection== I2C_Direction_Receiver)
{while(!I2C_CheckEvent(I2Cx,
I2C_EVENT_MASTER_RECEIVER_MODE_SELECTED));}
}
```

Листинг 3

```
// Функция передачи данных
void I2C_WriteData(I2C_TypeDef* I2Cx, uint8_t data)
{
// Вызвать библиотечную функцию передачи данных
I2C_SendData(I2Cx, data);
// Ждать окончания передачи данных
while(!I2C_CheckEvent(I2Cx, I2C_EVENT_MASTER_BYTE_TRANSMITTED));
}
// Функция приема данных
uint8_t I2C_ReadData(I2C_TypeDef* I2Cx)
{
// Ждать поступления данных
while( !I2C_CheckEvent(I2Cx, I2C_EVENT_MASTER_BYTE_RECEIVED) );
data = I2C_ReceiveData(I2Cx); // Считать данные из регистра
return data; // Возвратить данные в вызывающую функцию
}
```

Для настройки интерфейса I^2C в качестве ведущего устройства и передачи данных через него необходимо выполнить следующие действия:

1) разрешить тактирование портов;
2) инициализировать I^2C, задав его скорость, адрес и формат адреса;
3) назначить выводы микроконтроллера;
4) разрешить работу интерфейса;
5) сформировать стартовое условие;
6) послать адрес адресуемого устройства и данные;
7) сформировать стоповое условие.

Для облегчения процесса программирования желательно создать набор основных функций для работы с I^2C.

В листинге 1 приведена функция инициализации интерфейса I^2C в соответствии с описанным выше алгоритмом.

Теперь рассмотрим функцию для общения по I^2C. Для расширения возможностей эта функция имеет три параметра: номер используемого блока I^2C, направление передачи данных и адрес подчиненного устройства. Код данной функции приведен в листинге 2.

Приведенная функция использует простые функции передачи и приема данных, приведенные в листинге 3.

Закончив обмен данными по I^2C, необходимо вызвать функцию формирования стоп-условия I2C_Generate STOP(I2Cx, ENABLE).

На основе приведенных функций можно создавать программы для работы с множеством разнообразных периферийных устройств.

ЗАКЛЮЧЕНИЕ

Неоспоримым преимуществом интерфейса I^2C является простота подключения устройств с помощью всего лишь двух линий связи и общего провода, благодаря чему данный интерфейс надежно закрепился в технике и по-прежнему широко применяется в современной аппаратуре.

Литература:

1. www.st.com.
2. www.st.com/web/en/resource/technical/document/reference_manual/CD00246267.pdf.

CNY

* Статья перепечатана из журнала «Современная электроника», № 1, 2015 г., с разрешения редакции, тел. +7 (495) 232-00-87, www.soel.ru