Окончание. Начало в № 3 `2009

Синхронный последовательный интерфейс SPI в микроконтроллерах «от А до Я» и его реализация в ADuC70xx фирмы Analog Devices

Александр НОВИЦКИЙ

Микроконтроллеры семейства ADuC70xx представляют собой однокристальные системы обработки аналоговых сигналов на базе популярного RISC-ядра ARM7TDMI. Помимо высококачественной подсистемы аналого-цифрового преобразования (5—16 каналов АЦП, 12 бит, до 2 МГц, до 4 каналов ЦАП, 12 бит), микроконтроллеры содержат типовой набор периферийных устройств, в том числе интерфейс SPI.

В техническом описании микросистем семейства [1], несмотря на его значительный объем (92 стр.), отдельные подсистемы отражены весьма кратко, вследствие чего бывает нелегко разобраться в тонкостях их функционирования. Кроме того, в ряде мест описание содержит неточности и досадные ошибки. Автор данной статьи использует в учебном процессе в Санкт-Петербургском политехническом университете отладочные модули, содержащие микроконтроллеры ADuC7020. Для получения ответов на многочисленные вопросы, возникавшие в ходе изучения технического описания и подготовки учебных материалов, пришлось проделать достаточно большое количество экспериментов. Дальнейшее описание подсистемы в значительной степени основано на результатах проведенного тестирования. Для тестирования использовалась среда разработки Keil µVision3 v3.53, которая входит в набор разработчика.

Длина пакета в SPI МК ADuC70xx составляет 8 битов, пакеты большей длины (кратной 8 битам) могут формироваться программно.

Сигналы подсистемы SPI

Подсистема SPI в составе МК ADuC70xx использует четыре сигнала интерфейса, в дальнейшем тексте обозначаемые как:

- MOSI — выход данных в режиме ведущего, вход данных в режиме ведомого;
- MISO — вход данных в режиме ведущего, выход данных в режиме ведомого;
- SCK — выход тактирования (в режиме ведущего) или вход тактирования (в режиме ведомого);
- SS — вход детекции коллизии (в режиме ведущего), вход выбора ведомого (в режиме ведомого).

Последний сигнал в техническом описании [1] имеет обозначение CSL, однако в данной статье будет использоваться общепринятое обозначение SS.

Сигналы интерфейса SPI используют выводы P1.4, …, P1.7 параллельного порта P1 микроконтроллера. Опишем кратко организацию параллельных портов в МК ADuC70xx.

Внешние интерфейсные выводы микроконтроллеров ADuC70xx — многофункциональные, каждый вывод имеет до четырех разных функций. Выводы в МК объединены в пять байтовых портов (P0, P1, … P4). Каждый байтовый порт имеет программно-управляемый мультиплексор, позволяющий программисту выбрать функцию индивидуально для каждого контакта, выбор производится записью четырехбитовых комбинаций в регистр GPxCON (x = 0…4).

Для порта P1 функции выводов и управление мультиплексором сведены в таблицу 1. Выводы и их режимы, относящиеся к SPI, выделены жирным шрифтом. Для настройки выводов P1.4, …, P1.7 в режим SPI следует в соответствии с таблицей 1 записать в биты 31…16 регистра GP1CON значение 0x2222.

С каждым портом ввода/вывода связан многофункциональный 32-битовый регистр GPxDAT (x — номер порта, от 0 до 4), структура которого приведена в таблице 2. В режиме параллельного ввода/вывода биты 31…24 этого регистра определяют направление передачи каждого из выводов, биты 23…16 позволяют управлять выходными сигналами порта в режиме вывода. Чтение битов 7…0 регистра GPxDAT позволяет программе получить состояние сигналов на контактах порта.

Важно и полезно, что эта последняя возможность доступна при различных настройках мультиплексора. В частности, если выводы сконфигурированы для использования с подсистемой SPI, программа имеет возможность считывать уровни сигналов на линиях SPI чтением битов 7…4 регистра GP1DAT, в том числе и во время передачи пакета. Это использовалось при тестировании и позволило получить ответы на ряд вопросов.

Настройки направления передачи (биты 31…28 регистра GP1DAT) в режиме SPI не оказывают влияния на работу выводов P1.4, …, P1.7.

Таблица 1. Функции выводов и управление мультиплексором порта P1

Порт	Вывод	Управляющие биты в GP1CON	Код конфигурации (регистр GP1CON)			
			0000	0001	0010	0011
P1	P1.0	3...0	GPIO/T1	SIN (UART)	SCL0 (I²C)	PLAI[0]
	P1.1	7...4	GPIO	SOUT (UART)	SDA0 (I²C)	PLAI[1]
	P1.2	11...8	GPIO	RTS (UART)	SCL1 (I²C)	PLAI[2]
	P1.3	15...12	GPIO	CTS (UART)	SDA1 (I²C)	PLAI[3]
	P1.4	19...16	GPIO/IRQ2	RI (UART)	**CLK (SPI)**	PLAI[4]
	P1.5	23...20	GPIO/IRQ3	DCD (UART)	**MISO (SPI)**	PLAI[5]
	P1.6	27...24	GPIO	DSR (UART)	**MOSI (SPI)**	PLAI[6]
	P1.7	31...28	GPIO	DTR (UART)	**CSL (SPI)**	PLAO[0]

Таблица 2. Структура и назначение полей в регистрах портов GPxDAT

GPxDAT	31...24	23...16	15...8	7...0
Функция	Направление (чтение и запись)	Установка выходов (чтение и запись)	–	Сигналы на выводах (только чтение)
Выводы	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0	7 6 5 4 3 2 1 0

На каждом выводе портов в МК ADuC70xx имеется pull-up резистор (подтягивающий вход к логической единице). Величина этого резистора имеет порядок 100 кОм. Для части портов (в частности, для всех выводов порта P1) есть возможность программного отключения pull-up резисторов. В то же время выводы портов нельзя переключить в режим «открытого стока», которого требует организация параллельной магистрали на базе SPI с несколькими ведущими устройствами [3].

Регистровая модель подсистемы SPI

Программный интерфейс подсистемы SPI в ADuC70xx содержит пять программно доступных регистров:

- SPIDIV — регистр делителя, позволяет задать скорость следования битовых интервалов в режиме ведущего.
- SPITX — буфер передатчика. В него программа должна помещать очередной байт, подлежащий передаче.
- SPIRX — буфер приемника. В этот регистр после окончания обмена очередного пакета из сдвигового регистра копируется принятый байт.
- SPICON — регистр управления подсистемой SPI. Все переключения режимов производятся путем изменения отдельных битов в этом регистре.
- SPISTA — регистр состояния подсистемы SPI. Биты в этом регистре отражают состояние частей подсистемы SPI, а также наличие запросов прерывания от этой подсистемы.

Выбор скорости передачи (частоты следования битовых интервалов)

Сначала о подсистеме тактирования МК. Тактирование может производиться от одного из нескольких источников. Это: а) встроенный низкостабильный генератор 2^{15} Гц = = 32 768 Гц (±3%); б) внутренний генератор, стабилизированный внешним кварцевым резонатором с частотой 32 768 Гц; в) внешний генератор с частотой в диапазоне от 0,05 до 44 МГц. В вариантах *а* и *б* частота 32 768 Гц используется как входная для умножителя ФАПЧ с коэффициентом 1275, формирующим на выходе частоту с номинальным значением 41 779 200 Гц. Эта последняя частота обозначается в документации как UCLK и используется как задающая для тактирования процессора и некоторых внутрикристальных периферийных узлов. В варианте *в* значение UCLK задается внешним генератором.

С целью энергосбережения процессор может тактироваться частотой HCLK = UCLK/2^{CD}, где CD может задаваться программно в пределах от 0 до 7, что соответствует значениям HCLK от 41 779 200 до 326 400 Гц.

Частота следования битовых интервалов SPI задается содержимым регистра в соответствии с формулой:

$$F_{SCK} = \mathrm{UCLK}/(2\times(1+\mathrm{SPIDIV})),$$

где SPIDIV — содержимое регистра делителя SPI, то есть она не зависит от настройки частоты процессора (в отличие от интерфейсов UART и I^2C, которые входят в МК семейства ADuC70xx, для тактирования которых в качестве задающей используется HCLK). Однако максимальная битовая частота SPI ограничена сверху: в таблице 3 дано соотношение между настройкой тактовой частоты процессора и максимальной битовой частотой SPI для номинальной частоты задающего генератора 41 779 200 Гц.

При более низких частотах тактирования процессора (по утверждению производителя) интерфейс SPI неработоспособен.

В режиме ведомого тактирование подсистемы SPI осуществляется внешним сигналом SCK и поэтому происходит независимо от внутренней частоты тактирования МК. Однако максимально допустимое значение частоты SPI зависит от настройки частоты процессора и не может превышать HCLK/4. При максимальной допускаемой техническим описанием частоте тактирования 44 МГц (от внешнего генератора) частота приема данных может составлять до 11 Мбит/с (в техническом описании [1] указана величина 10,4 Мбит/с, она соответствует частоте тактирования 41 779 200 Гц, формируемой внутренним умножителем ФАПЧ).

Двойная буферизация приемопередатчика SPI в ADuC70xx и режимы запуска передачи

Сдвиговый регистр SPI в МК семейства ADuC70xx имеет двойную буферизацию как по передаче, так и по приему.

Буферизация передачи

Передаваемый байт записывается не прямо в сдвиговый регистр, а в буфер передачи SPITX. О состоянии буфера передачи «свободен» сигнализирует нулевое значение бита SPISTA.0 регистра состояния (табл. 5). Если выполнены условия запуска передачи пакета (см. далее), передаваемый байт немедленно копируется в сдвиговый регистр SPI, начинается передача, а буфер SPITX освобождается. В последнем случае программа может немедленно поместить в буфер SPITX следующий передаваемый байт.

Буферизация приема

По окончании передачи байт, принятый в сдвиговый регистр SPI, немедленно копируется в буфер приемника SPIRX. Наличие принятого байта в SPIRX сигнализирует единичное значение бита SPISTA.3. Если программа не успеет считать принятый байт до того, как таким же образом будет принят еще один байт, в регистре состояния устанавливается бит SPISTA.5 переполнения приема.

Режимы запуска передачи

Для подсистемы SPI, сконфигурированной как ведущее устройство, можно выбрать один из двух режимов: 1) запуск записью передаваемого байта в буфер передачи SPITX, 2) запуск чтением содержимого буфера приема SPIRX. Эти два режима далее будут обозначаться как «ведущий пуск записью» («ведущий-ПЗ») и «ведущий пуск чтением» («ведущий-ПЧ»). В подсистеме SPI, сконфигурированной как ведомое устройство, передача управляется внешними сигналами от ведущего, а запись в SPITX или чтение SPIRX не влияют на процесс передачи.

Таким образом, передатчик SPI может находиться в следующих состояниях.

1) Буфер передачи SPITX пуст, сдвиговый регистр остановлен.
2) Буфер передачи SPITX полон, сдвиговый регистр остановлен. Это состояние возможно либо в режиме «ведущий-ПЗ», либо в режиме «ведомый».
3) Буфер передачи пуст, идет сдвиг в регистре (передача пакета).
4) Буфер передачи полон, идет сдвиг в регистре.

Приемник SPI может находиться в следующих состояниях:

1) Буфер приема пуст, сдвиговый регистр остановлен.
2) Буфер приема пуст, идет сдвиг в регистре.
3) Буфер приема полон, сдвиговый регистр остановлен (обмен закончился, принятый байт еще не считан).
4) Буфер приема полон, идет сдвиг в регистре.

Таблица 3. Максимальная битовая частота SPI в зависимости от настройки частоты процессора

HCLK, Гц	41 779 200	20 889 600	10 444 800	5 222 400	2 611 200	1 305 600
При CD	0	1	2	3	4	5
FSCK, Гц	3 481 600	1 740 800	870 400	435 200	217 600	108 800
При SPIDIV	5	11	23	47	95	191

5) Буфер приема полон и не был считан, закончился прием следующего байта — переполнение буфера приема.

Далее будут описаны состав и назначение битов в SPISTA — регистре состояния подсистемы SPI. К сожалению, не все из перечисленных состояний можно различить, анализируя биты в SPISTA. В частности, нельзя определить состояние регистра сдвига: продолжается передача либо уже закончилась.

Управление режимами SPI в ADuC70xx

Управление осуществляется через регистр SPICON — управляющий регистр порта SPI (два байта). Функции битов этого регистра указаны в таблице 4.

Рассмотрим функции и особенности отдельных битов управления более подробно:

- Бит 0. Включает или отключает подсистему SPI. При выключении подсистемы очищаются и становятся недоступными для обращения регистры SPISTA, SPITX, SPIRX. Регистр SPIDIV доступен для обращения всегда, его содержимое не изменяется при переключении бита 0.
- Бит 1. Выбор режима «ведущий» либо «ведомый». В режиме ведущего выводы SCK и MOSI переходят в режим «выход», вывод MISO — в режим «вход». Функция вывода SS зависит от состояния бита SPICON.9 (см. описание далее). Режим запуска обмена зависит от значения бита SPICON.6. Бит SPICON.10 — не активен. В режиме ведомого выводы SCK, MOSI и SS переходят в режим «вход». Состояние вывода MISO: при SPICON.9 = 1 — выход, при SPICON.9 = = 1 — отключен.
- Бит 2. Выбор фазы тактирования (порядка действий «захват – сдвиг»). Влияние этого и следующего битов соответствует общепринятому и подробно описано в [2, 3].
- Бит 3. Полярность тактовых импульсов. Значение бита 3 задает уровень сигнала SCK в паузе между пакетами.
- Бит 5. Установка этого бита в 1 позволяет изменить порядок следования битов — начиная с младшего.
- Бит 6. При выборе варианта SPICON.6 = = 1 — «пуск записью» — запрос прерывания формируется в случае, когда в буфере передачи есть место для следующего байта. Изменение состояния буфера приема не вызывает запроса прерывания. Аналогично, при выборе SPICON.6 = 0 — «пуск чтением» — на формирование запроса прерывания не оказывает влияния изменение состояния буфера передачи. В режиме ведомого позволяет выбирать только источник запроса прерывания: состояние буфера приема либо состояние буфера передачи.
- Бит 7. В режиме «пуск чтением», когда ведущему МК требуется только принимать данные от ведомого устройства, позволяет избежать лишних действий при записи в буфер передачи. Эта возможность оказывается полезной при обмене с некоторыми типами ведомых, в частности с микросхемами Flash-памяти.
- Бит 8. Задает поведение буфера приема при его переполнении в режиме «пуск записью», либо в состоянии ведомого.
- Бит 9. В режиме ведущего, при SPICON.9 = 0 включается режим автоматического формирования сигнала SS. Вывод SS при этом является выходом. При пуске передачи пакета (как «чтением», так и «записью») схемотехника SPI переводит выход SS в активное (низкое) состояние, после чего начинается формирование пачки сдвиговых импульсов на выходе SCK. Если к моменту окончания передачи байта имеется условие запуска передачи следующего байта, передача тут же начинается — немедленно либо с промежутком в один период SCK. Наличие этого промежутка, а также поведение сигнала SS между пакетами зависит от установки бита SPICON.12. В режиме ведущего при SPICON.9 = 1 вывод SS переходит в режим входа «детекции коллизии» [3]. Этот режим позволяет обнаружить конфликт на симметричной магистрали с несколькими ведущими. Если данный МК серии ADuC70xx, подключенный к такой магистрали, находится в состоянии ведущего, на нем включен режим детекции коллизии, а какой-либо другой абонент также перешел в состояние ведущего и начал передачу пакета (переключил свой выход, соединенный с линией SS, в низкий уровень), то в данном МК серии ADuC70xx этот факт распознается автоматически. При этом: а) происходит переключение в режим ведомого, б) подсистема SPI отключается, и тем самым удается избежать логического и электрического конфликта на параллельной магистрали.
- Бит 10. Возможность отключить выход ведомого позволяет реализовать режим широковещательной передачи от ведущего ко многим ведомым в симметричной параллельной магистрали или в структуре «звезда». Вопросы реализации симметричной параллельной магистрали с использованием подсистемы SPI в ADuC70xx еще будут обсуждаться далее.
- Бит 11. Об использовании этой возможности автор не может сказать ничего определенного.
- Бит 12. Задает непрерывный или старт-стопный режим автоматического формирования сигнала SS совместно с битом SPICON.9.

Таблица 4. Назначение и свойства битов регистра управления SPICON

Бит	Название	Назначение
15–13	Не используются	–
12	Непрерывная передача/ Старт-стопная передача	1 — если буфер передачи полон, промежуток между передаваемыми байтами отсутствует, сигнал SS сохраняет состояние 0. 0 — после передачи байта автоматически формируемый сигнал SS переводится в 1 на время одного тактового периода, даже если буфер передачи полон. Бит активен только в состоянии ведущего (при SPICON.1 = 1)
11	Замкнуть выход на вход (тестовый режим)	Если 1 — тестовый режим (выход передатчика MOSI замыкается на вход приемника MISO). Если 0 — нормальный режим работы
10	Разрешение выхода данных ведомого	Если 1 — в режиме ведомого разрешается выходной сигнал MISO. Если 0 — в режиме ведомого вывод MISO в отключенном состоянии. Бит активен только в состоянии ведомого (при SPICON.1 = 0)
9	Режим работы вывода SS на ведущем	Если 1, то в режиме ведущего линия SS в состоянии «вход детекции коллизии». Если 0, то в режиме ведущего сигнал SS формируется автоматически. Бит активен только в состоянии ведущего (при SPICON.1 = 1)
8	Разрешение замены данных в SPIRX	Если 1, то при переполнении буфера приема SPIRX приходящий байт заменяет ранее принятый. Если 0 — новый принятый байт теряется. Бит не активен в режиме «ведущий пуск чтением» (при SPICON.1 = 1, SPICON.6 = 0)
7	Режим работы SPITX при запуске от SPIRX	Если 1, то (при установленном бите 6) передается нулевой байт. Если 0, то повторяется передача предыдущего байта. Бит активен только в режиме «ведущий пуск чтением» (при SPICON.1 = 1, SPICON.6 = 0)
6	Выбор режима запуска и прерывания	Если 1, то передача запускается в результате записи в буфер передачи SPITX. Запрос прерывания от SPI генерируется, если буфер передачи пуст. Если 0, передача запускается в результате чтения из буфера приема SPIRX. Запрос прерывания генерируется, если буфер приема полон. В режиме ведомого (при SPICON.1 = 0) бит SPICON.6 управляет только формированием сигнала запроса прерывания
5	Порядок передачи битов	Если 1, младшими битами вперед. Если 0, старшими битами вперед (стандартная опция SPI)
4	Не используется	–
3	Полярность тактирования	Если 1, уровень тактирования в паузе между пакетами — высокий. Если 0, уровень тактирования в паузе между пакетами — низкий
2	Фаза тактирования	Если 1, тактовый импульс в начале битового интервала (сначала сдвиг, затем захват [2, 3]). Если 0, тактовый импульс в конце битового интервала (сначала захват, затем сдвиг)
1	Режим ведущего/ведомого	Если 1, SPI в режиме ведущего. Активны биты 6, 9, 12, они определяют событие, запускающее передачу и поведение вывода SS. Если 0, SPI в режиме ведомого
0	Разрешение SPI	Если 1, подсистема SPI включена. Если 0, выключена. При выключении все прочие установки регистра SPICON сохраняют свои значения, а биты регистра SPISTA сбрасываются

Теперь рассмотрим состав регистра состояния SPISTA, он представлен в таблице 5. Следует отметить, что в техническом описании [1] допущен в отношении этого регистра ряд досадных неточностей. Приводимая в статье информация основывается в значительной степени на результатах экспериментального тестирования реальной микросистемы.

Рассмотрим функции битов регистра состояния SPISTA более обстоятельно:

- Бит 0. В техническом описании [1] указано, что этот бит устанавливается при записи в буфер передачи и остается установленным, пока происходит передача данных. Если под передачей понимать сдвиг регистра SPI, то это утверждение является ошибочным. Тестирование показало, что бит 0 отражает

состояние буфера передачи: 0 соответствует состоянию «пуст», а 1 — состоянию «полон». При «пуске записью», если перед этим сдвиговый регистр был остановлен, записанный в буфер передачи байт немедленно копируется в сдвиговый регистр, начинается передача, а бит SPISTA.0 остается в состоянии 0 (пуст). После этого программа может немедленно записать в буфер передачи второй байт, и лишь тогда бит SPISTA.0 переходит в состояние 1 (полон).

- Бит 1. Бит запроса прерывания от передатчика. В режиме «пуск записью» (при SPICON.6 = 1) бит 1 устанавливается, если есть место в буфере передачи (если бит SPISTA.0 = 0). В режиме «пуск чтением» (при SPICON.6 = 0) бит 1 устанавливается, если установлен бит SPISTA.2 (SPITX Data Register Underflow Status Bit).
- Бит 2. Этот бит активен в режиме «ведущий-ПЧ» и «ведомый». Бит 2 устанавливается в 1, если перед пуском обмена (чтением буфера приема SPIRX) в буфер передачи не был помещен новый байт. Это состояние в оригинальном техническом описании названо так: SPITX Data Register Underflow. Автор затрудняется подобрать подходящий русский эквивалент. В русском переводе предварительного технического описания МК семейства ADuC70xx переводчик «сконструировал» термин «недополнение», однако этот термин представляется совершенно неудовлетворительным.
- Бит 3. Устанавливается во всех режимах в последнем, восьмом периоде пачки тактов и сигнализирует о наличии очередного принятого байта в буфере приема SPIRX. Сбрасывается автоматически при чтении буфера приема SPIRX.
- Бит 4. Сигнализирует о наличии непрочитанного байта в буфере приема. Данный бит вызывает запрос прерывания от подсистемы SPI, если сброшен бит SPICON.6, то есть если подсистема SPI находится в режиме «ведущий пуск чтением» либо в режиме «ведомый».
- Бит 5. Устанавливается во всех режимах в последнем, восьмом периоде пачки тактов, если ранее принятый байт не был прочитан из буфера приема. Сигнализирует о том, что один из принятых байтов безвозвратно потерян. Бит SPICON.8 определяет, который байт — новый или старый — будет потерян при переполнении.

Отметим еще одну особенность регистра состояния SPI: его биты никак не отражают состояние сдвигового регистра. Если необходимо по какой-либо причине деактивировать интерфейс SPI сбросом бита SPICON.0, предварительно следует убедиться, что закончилась активность в сдвиговом регистре. Единственная реально существующая возможность узнать об этом — программный опрос сигнала SS, который может быть произведен чтением и анализом бита 7 регистра GP1DAT.

Функционирование в режиме «ведущий пуск записью в TX»

После включения данного режима буфер SPITX свободен, бит SPISTA.0 сброшен. Поскольку прерывание формируется по освобождению буфера SPITX, запрос прерывания появляется сразу.

После записи первого байта в буфер передачи SPITX этот байт немедленно копируется из TX в сдвиговый регистр, начинается сдвиг пакета. Буфер передачи остается свободным, запрос прерывания по-прежнему активен. Это позволяет немедленно записать второй байт в SPITX. Только после этого устанавливается бит SPISTA.0 «буфер передачи полон», а запрос прерывания снимается.

В восьмом периоде тактовой последовательности устанавливается бит SPISTA.3 «байт принят». На чтение принятого байта есть время до окончания приема следующего. Можно (следует) сначала прочитать принятый байт, а затем записать следующий для передачи. Можно и в обратном порядке, но тогда нужно убедиться, что успеем прочитать. Если не успели прочитать до окончания передачи второго пакета, происходит переполнение. Устанавливается флаг ST.5 (переполнение), и какой байт будет потерян, определяет бит CON.8.

На рис. 1 показаны диаграммы сигналов на выходах SS, SCK и MOSI. Последний сигнал сформирован искусственно на выходе параллельного порта P4.2 и демонстрирует поведение флага состояния буфера передатчика SPISTA.0. Диаграмма получена при циклическом выполнении фрагмента программы со следующей структурой:

```
while(1) {
   while (SPISTA&1) {GP4SET = 0x40000;} // Опрос буфера
                                        // передатчика
   GP4CLR = 0x40000;
   SPITX=0x36;                          // Передали байт 0x36
   while (SPISTA&1) {GP4SET = 0x40000;} // Опрос буфера
                                        // передатчика
   GP4CLR = 0x40000;
   SPITX=0x74;                          // Передали байт 0x74
   while (SPISTA&1) {GP4SET = 0x40000;} // Опрос буфера
                                        // передатчика
   GP4CLR = 0x40000;
     delay(200);
}                                       // Конец цикла while
```

Эксперимент был проведен при следующих установках:

- SPICON.0 = 1 — подсистема SPI включена.
- SPICON.1 = 1 — режим ведущего.
- SPICON.2 = 0 — фаза тактирования: «сначала захват, затем сдвиг».
- SPICON.3 = 0 — полярность тактирования: уровень в паузе «низкий».
- SPICON.5 = 0 — порядок передачи — старшими битами вперед.
- SPICON.6 = 1 — пуск записью в буфер передачи.
- SPICON.9 = 0 — автоматическое формирование сигнала SS.
- SPICON.12 = 1 — старт-стопная передача.

Остальные, не указанные в перечне биты регистра SPICON были равны нулю, эти установки не существенны для режима «ведущий пуск записью», в котором производилось тестирование.

Битовая частота составляла около 600 кГц, длина битового интервала — около 1,6 мкс. Тактовая частота ядра была равна 21 МГц, среднее время выполнения команды ARM — около 100 нс.

Из приведенного тестового фрагмента программы должно быть ясно, как формировался сигнал, демонстрирующий поведение флага буфера передачи: между посылками байтов анализировалось состояние флага, если он установлен в 1, то вывод P4.2 также устанавливался в 1.

Вначале флаг SPISTA.0 сброшен, буфер SPITX свободен, в него помещается первый байт 0x36. Последующий анализ флага снова дает результат «свободен», в буфер SPITX помещается второй байт 0x74, только после

Таблица 5. Назначение и свойства битов регистра состояния SPISTA (один байт)

Бит	Описание
7–6	Резервировано
5	Флаг переполнения буфера приемника SPIRX. Устанавливается, если следующий байт принят до того, как был прочитан предыдущий. Сбрасывается при чтении SPISTA или при чтении SPIRX
4	Флаг запроса прерывания от приемника SPI. Устанавливается, если установлен бит SPISTA.3[1]. Сбрасывается при чтении регистра SPIRX
3	Флаг наличия нового байта в SPIRX. Устанавливается по окончании приема байта. Сбрасывается при чтении регистра SPIRX
2	Оригинальное название этого бита — SPITX Data Register Underflow Status Bit. Активен только в режиме SPICON.6 = 1 (запуск передачи по чтению буфера SPIRX). Устанавливается в 1 в том случае, если перед запуском передачи в буфер SPITX не был помещен байт. Бит сбрасывается только при чтении регистра статуса SPISTA[2]
1	Бит запроса прерывания от передатчика SPI. Устанавливается, если SPISTA.0 = 0, то есть если буфер передачи SPITX пуст, либо если SPISTA.2 = 1, то есть если перед «запуском чтением» в буфер передачи не был помещен байт (SPITX Data Register Underflow)
0	Флаг состояния буфера передатчика. Устанавливается в 1, если буфер полон. Сбрасывается после начала передачи последнего байта, когда освобождается буфер SPITX[3]

1 В техническом описании [1] указано, что SPISTA.4 устанавливается, если установлен любой из битов SPISTA.3 или SPISTA.5. Однако бит 5 может быть установлен только при установленном бите 3, поэтому указание второго условия представляется излишним.
2 Вопреки утверждению, приведенному в техническом описании [1], бит SPISTA.2 не удается сбросить записью в буфер передачи SPIRX.
3 В техническом описании написано: “This bit is set during transmission of data”. («Этот бит установлен, пока продолжается передача»). Данное утверждение неверно.

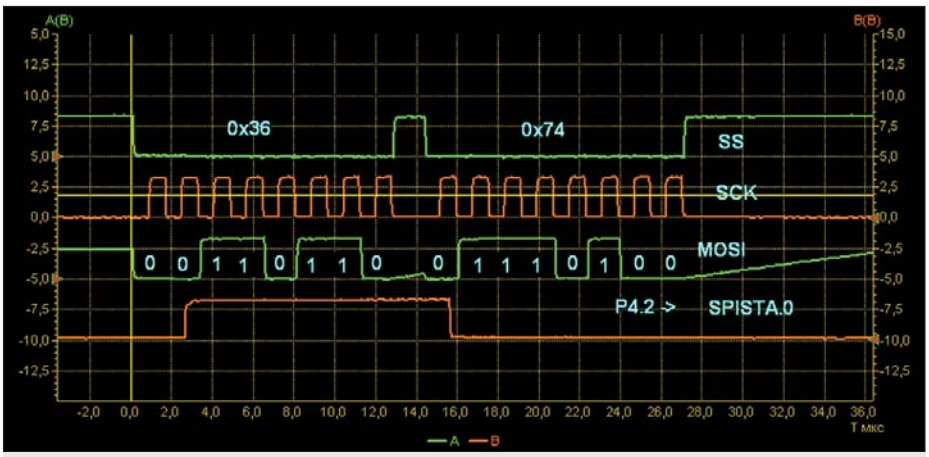

Рис. 1. Временные диаграммы передачи двух байтов 0x36 и 0x74 при SPICON.2 = SPICON.3 = 0 в режиме «старт–стоп» (SPICON.12 = 0)

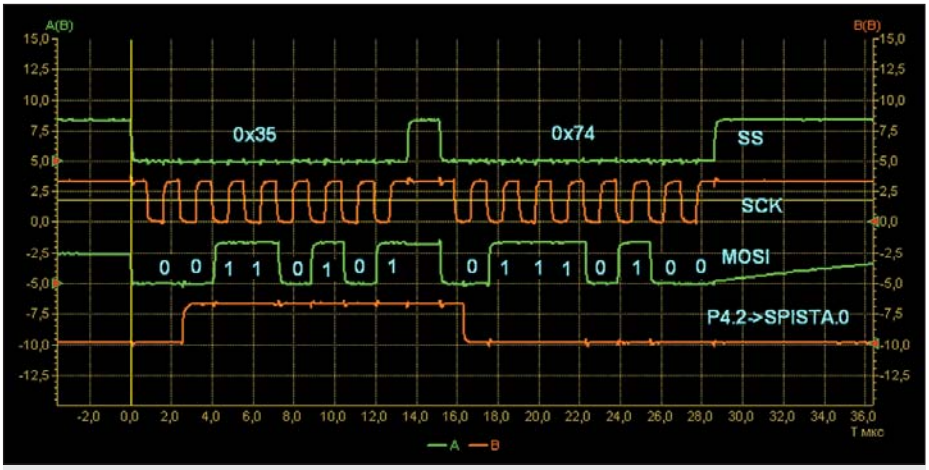

Рис. 2. Диаграммы передачи двух байтов 0x35 и 0x74 при SPICON.2 = SPICON.3 = 1

этого анализ флага дает значение «занято», которое исчезает по окончании передачи первого байта. Задержки изменения состояния выхода, наблюдаемые на диаграмме, вызваны выполнением команд опроса и записи байта в буфер.

Диаграмма сигнала SS формируется автоматически, на ней видно, что между передачами байтов сигнал SS на один период тактирования возвращается в неактивное состояние. В периоды, когда сигнал SS неактивен, выход MOSI переходит в отключенное состояние, на диаграмме виден процесс перезаряда паразитной емкости линии MOSI через «подтягивающий» резистор. Переключения битов данных происходят по четным перепадам тактового сигнала.

На рис. 2 изображены диаграммы, отличающиеся от предыдущих полярностью и фазой тактовых импульсов, а также значениями передаваемых байтов (0x35 и 0x74).

Как видно на диаграммах, в данном случае тактовая последовательность начинается с запаздыванием в полпериода по отношению к сигналу SS, а переключения битов данных привязаны к нечетным перепадам тактового сигнала. Длительность передачи одного байта в этом режиме оказывается на полпериода длиннее, нежели в предыдущем. Диаграммы, так же как и предыдущие, соответствуют режиму старт-стопной передачи (SPICON.12 = 0).

Запуск чтением SPIRX

Для выбора этого режима должен быть сброшен бит SPICON.6. В этом режиме, если требуется передача в обоих направлениях, следует сначала записать передаваемый байт (один!) в буфер передачи SPITX. При этом установится флаг SPISTA.0 «буфер передачи полон».

Для запуска передачи пакета следует выполнить операцию чтения буфера приема SPIRX. В результате начинается сдвиг пакета, действия при автоматическом формировании сигнала SS будут происходить в следующем порядке:

- Устанавливается активный (низкий) уровень сигнала SS.
- Одновременно происходит копирование содержимого буфера SPITX в сдвиговый регистр.
- Сбрасывается флаг SPISTA.0 «буфер передачи полон»; проверив его, можно поместить в буфер передачи очередной байт.
- Через полпериода битовой частоты начинается формирование тактовой последовательности.
- По началу восьмого тактового импульса устанавливается флаг SPISTA.3 «буфер приема полон». Можно считать содержимое буфера приема (что запустит передачу следующего пакета).
- Если запуска передачи следующего пакета не было, заканчивается восьмой тактовый период, а затем устанавливается пассивный (высокий) уровень сигнала SS.

Работа интерфейса в режиме ведомого устройства

В режиме ведомого устройства функция бита SPICON.6 ограничивается, он управляет только режимом формирования запроса прерывания от подсистемы SPI.

При необходимости передачи данных из ведомого устройства в ведущее первый передаваемый байт на ведомом должен быть заблаговременно (до активизации внешнего сигнала SS) помещен в буфер передачи SPITX. При этом немедленно установится бит SPISTA.0 «буфер передачи занят» и снимается запрос прерывания (бит SPISTA.1).

Копирование байта в сдвиговый регистр с освобождением буфера передачи SPITX происходит только при активации входа SS из ведущего устройства. Об этом сигнализирует сброс бита SPISTA.0 и установка бита SPISTA.1. Последнее действие может вызвать запрос прерывания, если он разрешен.

Затем под действием последовательности тактовых импульсов, поступающих из ведущего, происходит обмен битов. На последнем, восьмом такте в сдвиговый регистр ведомого поступает восьмой бит данных, что немедленно вызывает установку бита SPISTA.3.

Сигнал запроса прерывания от подсистемы SPI в режиме ведомого формируется в зависимости от состояния бита SPICON.6 либо по опустошению буфера передачи, либо по заполнению буфера приема.

SPI и подсистема прерывания

Как уже было отмечено, подсистема SPI генерирует запрос прерывания в зависимости от значения бита SPICON.6 либо по опустошению буфера передачи (SPICON.6 = 1), либо по заполнению буфера приема (SPICON.6 = 0). Разрешить одновременную реакцию на оба эти события невозможно.

Запрос, сформированный подсистемой SPI, поступает на один из двух входов внутрикристального контроллера прерываний, один вход активен в режиме SPI-ведущего, второй — в режиме SPI-ведомого. Который из этих двух входов активен, определяется состоянием бита SPICON.1 «ведущий — ведомый». В свою очередь контроллер прерываний дает возможность программисту направить каждый из этих двух входов (или оба вместе) на один из двух типов запроса пре-

рывания, определенных спецификацией ARM: запрос обычного прерывания IRQ или запрос быстрого прерывания FIQ [4].

Автор не может объяснить целесообразности наличия в контроллере прерывания двух различных запросов, соответствующих режимам SPI ведущего и ведомого.

О возможности организации симметричной параллельной магистрали с несколькими ведущими

Такая возможность предусмотрена во многих существующих реализациях интерфейса SPI, принципы ее организации подробно рассмотрены в [3]. В микроконтроллерах семейства ADuC70xx реализована аппаратная детекция коллизии, необходимая для организации симметричной параллельной магистрали, хотя техническое описание [1] не содержит какой-либо информации о детекторе коллизии и его свойствах.

Детектор коллизии в ADuC70xx работает следующим образом. В режиме ведущего в данном МК установка бита SPICON.9 = 1 переключает вывод SS в режим входа детекции коллизии. Если другой абонент на параллельной симметричной магистрали делает попытку перейти в режим ведущего и начать передачу (устанавливает на линии SS низкий уровень), в данном МК срабатывает детектор, то есть автоматически происходит сброс двух младших битов в регистре SPICON: включение режима «ведомый» (SPICON.1 = 0) и отключение подсистемы SPI (SPICON.1 = 0). Это позволяет избежать электрического конфликта на линиях магистрали.

Чувствительность аппаратного детектора коллизии в экспериментах автора с МК ADuC7020 была достаточно высока: срабатывание происходило даже при касании вывода SS стандартным щупом осциллографа (1 МОм, 30 пФ).

Однако реализация магистрали оказывается затруднительной из-за того, что в ADuC70xx отсутствует запрос аппаратного прерывания и флаг ошибки, сигнализирующие о возникновении коллизии. Подобная возможность имеется во многих МК (см., например, «прародителя» SPI — семейство 68HC11 [5]). По этой причине придется анализировать в ADuC70xx отсутствие коллизии программно, например, проверяя после попытки перехода в режим ведущего состояние битов SPICON.(1:2).

Еще одна причина, затрудняющая реализацию магистрали, — отсутствие в ADuC70xx возможности переключения линий интерфейса SPI в режим работы «с открытым стоком» (подобная возможность имеется в семействе 68HC11 и во многих других). Разработчику придется принимать дополнительные меры, чтобы избежать электрических коллизий на линии SCK.

В заключение автор считает необходимым еще раз отметить: стандарта на интерфейс SPI не существует, его реализации могут существенно различаться, и это затрудняет его использование в практических разработках и перенос программ с одной платформы на другую.

Литература

1. ADuC70xx Series: Precision Analog Microcontroller, 12-Bit Analog I/O, ARM7TDMI MCU Data Sheet (Rev B, 04/2007). Техническое описание микроконтроллеров семейства ADuC70xx. www.analog.com.
2. Новицкий А. Синхронный последовательный интерфейс SPI в микроконтроллерах «от А до Я» и его реализация в ADuC70xx фирмы Analog Devices // Компоненты и технологии. 2009. № 3.
3. Новицкий А. Синхронный последовательный интерфейс SPI в микроконтроллерах «от А до Я» и его реализация в ADuC70xx фирмы Analog Device // Компоненты и технологии. 2009. № 5.
4. ARM Architecture Reference Manual. 1996–2005. ARM Limited. Order No. ARM DDI 0100 I. Спецификация архитектуры ARM. www.arm.com.
5. MC68HC911B32 Technical Data, Motorola Semiconductor, Order No. MC68HC912B32TS/D.