STM8 и STM32 — объединенное пространство 8- и 32-разрядных микроконтроллеров

Компания STMicroelectronics, имея в своем портфолио несколько семейств микроконтроллеров (МК), продолжает развивать существующие и выпускать на рынок новые линейки продуктов. ST воплощает новые, подчас революционные концепции, облегчающие работу инженеров и разработчиков. В этой статье рассказано об одной из таких идей — функциональной и архитектурной сочетаемости двух семейств микроконтроллеров: 8-разрядных STM8 и 32-разрядных STM32 на примерах их стандартных линеек STM8S и STM32F. Эти два семейства на данный момент — одни из самых перспективных и доступных продуктов, выпускаемых компанией ST. Более того, предприняты существенные усилия по совершенствованию номенклатуры обоих семейств, что позволяет экономить, используя общую периферию и программные инструменты. Подобный подход облегчает переход с одних микроконтроллеров на другие, что снижает издержки при разработке проектов, состоящих из множества модификаций базового изделия.

Александр БОРОДУЛИН
alexander.borodulin@msk.petrointrade.ru

Введение

Временные и материальные затраты на разработку с использованием нового семейства МК — один из главных критериев при выборе производителя микроконтроллеров. Чем более широкий диапазон МК может быть использован в проекте, тем большее количество оптимальных преимуществ может быть получено, и, как следствие, наблюдается более эффективная отдача от подобных инвестиций. Номенклатура семейств STM8 и STM32 предусматривает микросхемы с количеством выводов от 20 до 144 и размером памяти от 2 до 512 кбайт. При этом в случае, если 8-разрядное приложение исчерпывает доступную производительность, существует возможность модернизации путем перехода к семейству STM32. И наоборот, если есть потребность сократить издержки при использовании 32-разрядной платформы, относительно просто перейти к 8-разрядному семейству STM8.

Читателю, не знающему об особенностях семейств МК STM8 и STM32, рекомендуется предварительно ознакомиться с ранее опубликованными материалами об этих семействах и информацией на сайте производителя (www.st.com).

Рассмотрим некоторые сходные черты и общие для обоих семейств функциональные особенности. Знание подобных деталей облегчит переход от одного семейства к другому в случае необходимости.

Ядро

Ядро STM8 — собственная архитектура компании ST, являющаяся развитием ядра предыдущего поколения — ST7. При этом STM8 можно назвать прорывом с точки зрения 8-разрядной производительности и компактности кода. Семейство STM32 построено на популярном 32-разрядном ядре ARM Cortex-M3, что позволяет использовать имеющиеся в изобилии для ARM-архитектур средства разработки и программные решения. Несмотря на то, что эти два ядра могут, на первый взгляд, казаться радикально отличающимися друг от друга, в действительности они имеют много общих архитектурных черт. Сравнить характеристики ядер можно по таблице 1.

Оба семейства построены на гарвардской архитектуре и имеют 3-ступенчатый конвейер, который минимизирует время выполнения команд. Максимальная тактовая частота — 24 МГц для STM8 и 72 МГц для STM32 (будет увеличена в новых подсемействах).

Семейства разработаны для построения систем с максимальной энергоэффективностью и имеют несколько режимов управления энергопотреблением. STM8 и STM32 используют внутренние интерфейсы памяти шириной больше, чем средняя длина инструкции (32- и 64-разрядные шины соответственно). Это минимизирует число доступов к шине памяти, а следовательно, и потребление электроэнергии, связанное с операциями по шине и чтением энергонезависимой памяти. Технология непрерывной обработки

Таблица 1. Сравнение ядер микроконтроллеров STM8 и STM32

	STM8	STM32
Ширина слова для данных, разряд	8	32
Производительность (0ws)	0,29 DMIPS/МГц	1,22 DMIPS/МГц
Архитектура	Гарвард	Гарвард
Конвейер	3-ступенчатый	3-ступенчатый
Набор инструкций	CISC	RISC
Организация памяти программ, разряд	32	32
Буфер предвыборки, разряд	2×32	2×64
Средний размер инструкции, байт	2	2
Тип прерываний	Векторизованные	Векторизованные
Задержка реагирования на прерывание	9 циклов, поддержка непрерывной обработки прерываний с исключением внутренних операций над стеком	12 циклов, поддержка непрерывной обработки прерываний с исключением внутренних операций над стеком
Режимы управления энергопотреблением	Ожидание события или прерывания; ждущий; активный ждущий режим	Сон (ожидание события или прерывания); сон по выходу; глубокий сон
Отладочный интерфейс	1-проводной (SWIM)	2-проводной или стандартный JTAG

прерываний с исключением внутренних операций над стеком (tail chaining) сокращает время реакции на прерывания и исключает лишние операции со стеком.

С точки зрения компактности кода оба семейства имеют превосходные показатели, это следствие используемых наборов инструкций: обновленного 8-разрядного набора команд семейства STM8 и эффективного 16/32-разрядного Thumb-2 режима для семейства STM32.

Итак, можно утверждать, что оба эти семейства являются современными с точки зрения внутренней микроархитектуры. STM8 отвечает тем же требованиям, что и простые 16-разрядные процессоры. STM32 соответствует высокопроизводительным 32-разрядным приложениям, но может быть использовано и в качестве замены 16-разрядных МК среднего и высокого уровня. Вместе семейства устанавливают новое пространство универсальности 8- и 32-разрядной номенклатуры МК, которое теперь также поддержано производителями инструментальных средств, предлагающими унифицированные платформы разработки для обоих семейств.

Периферия

Периферия МК является другим примером связи между 8-и 32-разрядными семействами: бóльшая часть основной периферии была разработана и структурирована для того, чтобы ее можно было переносить от одного семейства продуктов к другому. Это было сделано адаптацией простой, но эффективной 8-разрядной периферии к 32-разрядному миру. Результатом подобного подхода являются низкая стоимость при высокой энергоэффективности, простые в освоении ресурсы, дополненные на уровне кристалла многоразрядными шинами, а также контроллером прямого доступа к памяти, для случаев, когда необходима высокая производительность. Принцип работы периферии, однажды освоенный инженером-разработчиком, применим и к семейству STM8, и к семейству STM32.

На рис. 1 приведено упрощенное представление цифрового периферийного устройства.

Периферийный узел может быть разделен на два главных блока. Первый блок — это ядро, которое содержит конечные автоматы, счетчики и любой вид комбинаторной или последовательной логики. Оно предназначено для выполнения задач, не требующих участия процессора, таких как простые задачи передачи данных, управления аналоговыми входами или выполнения функций, привязанных к синхросигналам. Ядро периферийного узла связывается с внешним миром через порты ввода/вывода МК. Внешние соединения могут состоять из нескольких сигналов или сложных шин.

Второй блок — настройка и управление периферией, которые осуществляются

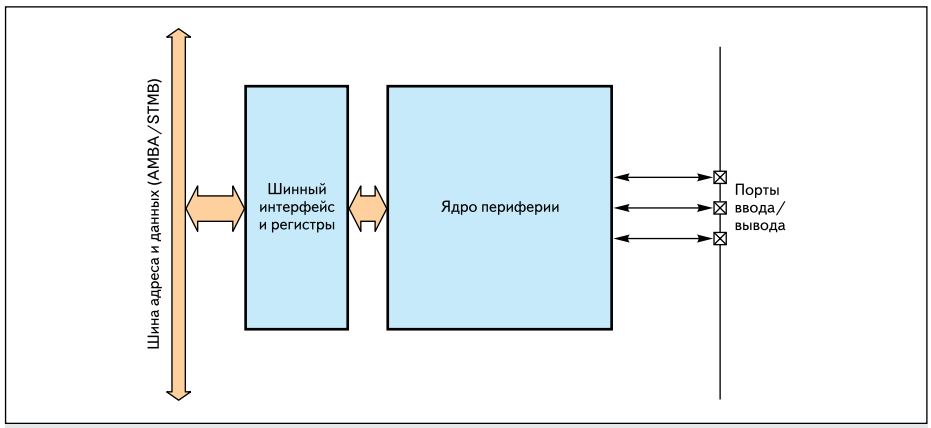

Рис. 1. Внутренняя структура цифровой периферии

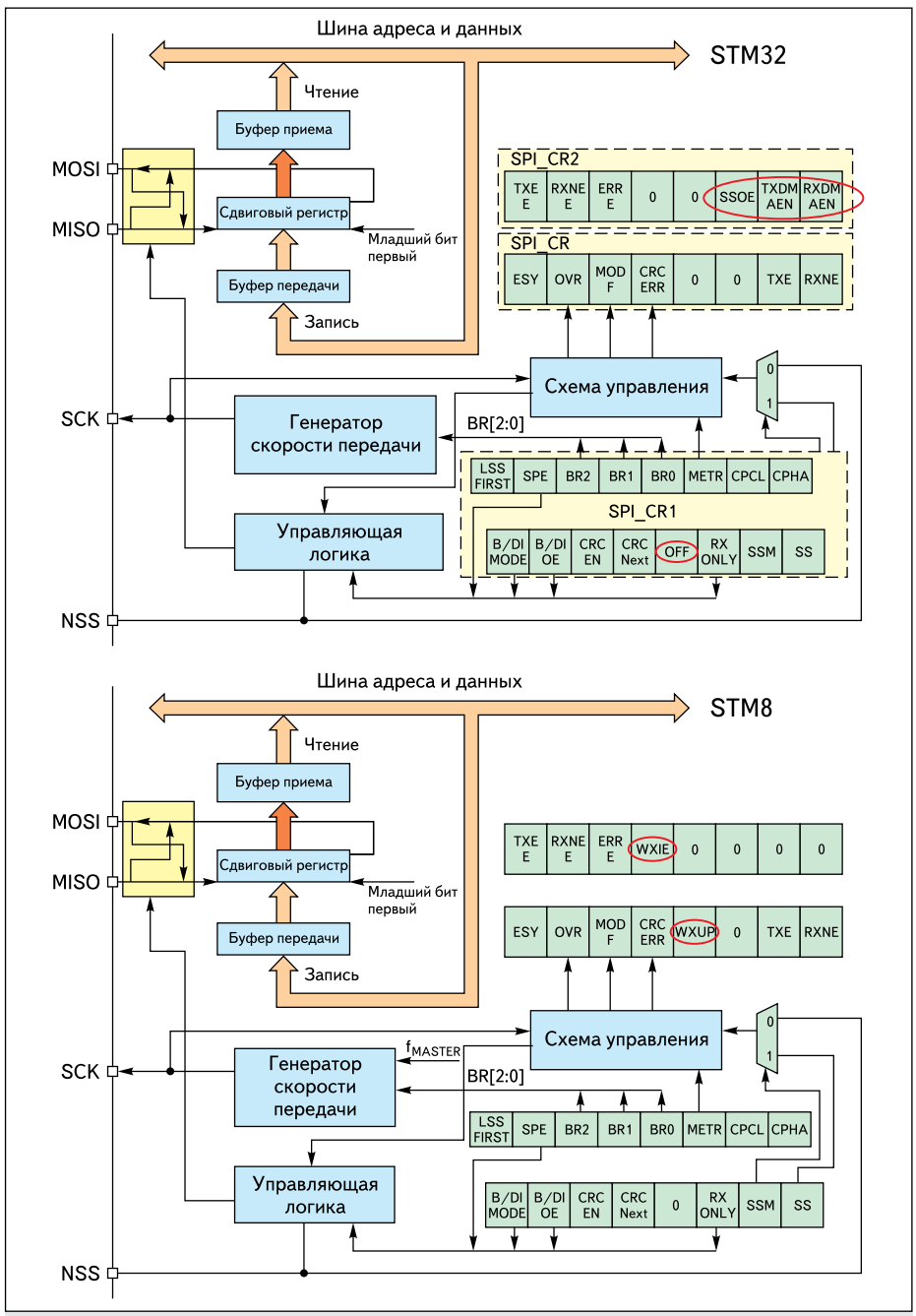

Рис. 2. Блок-схемы периферийных узлов SPI STM32 и STM8

Таблица 2. Карта SPI-регистров STM32

Смещение	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	SPI_CR1	Резерв																BIDIMODE	BIDIOE	CRCEN	CRCNEXT	DFF	RXONLY	SSM	SSI	LSBFIRST	SPE	BR [2:0]			MSTR	CPOL	CPHA
	после сброса																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	SPI_CR2	Резерв																								TXEIE	RXNEIE	ERRIE	PE3EPB		SSOE	TXDMAEN	RXDMAEN
	после сброса																									0	0	0			0	0	0
0x08	SPI_SR	Резерв																								BSY	OVR	MODF	CRCERR	UDR	CHSIDE	TXE	RXNE
	после сброса																									0	0	0	0	0	0	1	0
0x0C	SPI_DR	Резерв																DR[15:0]															
	после сброса																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	SPI_CRCPR	Резерв																CRCPOLY[15:0]															
	после сброса																	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0x14	SPI_RXCRCR	Резерв																RXCRC[15:0]															
	после сброса																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x18	SPI_TXCRCR	Резерв																TXCRC[15:0]															
	после сброса																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x1C	SPI_I2SCFGR	Резерв																				I2SMOD	I2SE	I2SCFG	PCMSYNC		PE3EPB	I2SSTD		CKPOL	DATLEN		CHLEN
	после сброса																					0	0	0	0	0		0	0	0	0	0	0
0x20	SPI_I2SPR	Резерв																						MCKOE	ODD	I2SDIV							
	после сброса																							0	0	0	0	0	0	0	0	1	0

приложением через регистры, соединенные с внутренней шиной, разделяемой с другими ресурсами МК. В 8-разрядных микроконтроллерах процессор непосредственно пишет и читает эти регистры, тогда как в 32-разрядных МК операции чтения и записи регистров обычно осуществляются через шинный мост. Главное различие между двумя семействами, однако, лежит в специфике шины, которой должен удовлетворять периферийный узел. Это объясняет, почему STM8S и STM32 могут разделять периферию: они построены на сходных ядрах периферийных узлов, но выполнены для различных шинных интерфейсов. В STM32 ARM процессор и периферийные узлы отвечают шинной спецификации AMBA с 32-разрядной шиной данных, тогда как в STM8 используется более простой стандарт 8-разрядной шины.

С функциональной точки зрения отличия в следующем:

- Размер регистров: 8 против 16 или 32 разрядов.
- Максимальная частота синхронизации, которая зависит от рабочей частоты центрального процессора.
- Контроллер прямого доступа к памяти, который разгружает центральный процессор от простых операций с данными и увеличивает максимальную пропускную способность.
- Некоторые присущие продуктам особенности, например управление портами ввода/вывода.

Рассмотрим блок-схемы периферийных узлов SPI STM8 и STM32, приведенные на рис. 2. Структурно они выглядят идентично за исключением нескольких битов, выделенных красным.

Таблица 3. Карта SPI-регистров STM8

Смещение	Регистр	7	6	5	4	3	2	1	0
0x00	SPI_CR1	LSBFirst	SPE	BR2	BR1	BR1	MSTR	CPOL	CPHA
после сброса	0	0	0	0	0	0	0	0	
0x01	SPI_CR2	BDM	BDOE	CRCEN	CRCNEXT	Резерв	RXONLY	SSM	SSI
после сброса	0	0	0	0	0	0	0	0	
0x02	SPI_ICR	TXIE	RXIE	ERRIE	WKIE	Резерв	Резерв	Резерв	Резерв
после сброса	0	0	0	0	0	0	0	0	
0x03	SPI_SR	BSY	OVR	MODF	CRCERR	WKUP	Резерв	TXE	RXNE
после сброса	0	0	0	0	0	0	1	0	
0x04	SPI_DR	MSB	–	–	–	–	–	–	LSB
после сброса	0	0	0	0	0	0	0	0	
0x05	SPI_CRCPR	MSB	–	–	–	–	–	–	LSB
после сброса	0	0	0	0	0	1	1	1	
0x06	SPI_RXCRCR	MSB	–	–	–	–	–	–	LSB
после сброса	0	0	0	0	0	0	0	0	
0x07	SPI_TXCRCR	MSB	–	–	–	–	–	–	LSB
после сброса	0	0	0	0	0	0	0	0	

Теперь проанализируем карты регистров SPI в таблицах 2, 3. Очевидно, они происходят от почти одного и того же ядра периферийного узла: кроме нескольких битов и общей разрядности, регистры и биты в них имеют сходные названия и занимают те же положения.

В таблице 4 перечислены сходные для двух семейств периферийные узлы.

Таблица 4. Разделяемая периферия между STM8 и STM32

STM32	STM8
Независимый сторожевой таймер (IWDG)	
Оконный сторожевой таймер (WWDG)	
Последовательный периферийный интерфейс (SPI)	
Интерфейс внутрисхемной связи (I^2C)	
Универсальный синхронный/асинхронный приемник/передатчик (USART)	
Таймеры расширенного управления	Таймер расширенного управления
Таймеры общего назначения	Таймеры общего назначения
Базовые таймеры	Базовый таймер

Несмотря на то, что таймеры, на первый взгляд, отличаются друг от друга и имеют множество различных конфигураций, их архитектура внутри и между семействами продуктов одинакова. Стоит обратить внимание только на вариации архитектуры отдельных таймеров. Начиная с максимально полного набора функций, подблоки таймера могут быть сокращены для уменьшения числа каналов захвата/сравнения или упразднения функций, необходимых только для некоторых специфичных применений, например, таких как управление электродвигателями.

Системные функции

Современные МК — сложные системы на кристалле, которые включают в себя не только большой набор периферии, но и развитую функциональность на уровне системы, ведущую к сокращению перечня используемых элементов, увеличению безо-

пасности и надежности работы продуктов. Это верно и для 8-, и для 32-разрядных платформ МК компании STMicroelectronics.

Сброс

Как показано на рис. 3, у STM8 и STM32 используется сходная схема сброса. Вывод NRST является одновременно и входом, и выходом с открытым стоком и встроенной нагрузкой к напряжению питания. Для увеличения надежности имеется фильтр, останавливающий распространение электромагнитных помех в цифровую схему. Можно обозначить три преимущества двунаправленного сброса:

- Для систем с несколькими МК двунаправленный сброс гарантирует, что все процессоры будут корректно синхронизированы при включении или в случае «горячего» перезапуска.
- Схемы мониторинга напряжения (сброс при подаче питания и сброс при отключении питания), встроенные в МК, могут также использоваться на системном уровне другими микросхемами.
- Помогает во время отладки для отслеживания факта внутреннего сброса.

Синхронизация

Продукты этих двух семейств имеют три главных источника синхроимпульсов со схожими электрическими особенностями (табл. 5).

Таблица 5. Характеристики источников синхронизации STM8/STM32

Источник синхроимпульсов	Частота		Точность	Ток потребления
	STM8	STM32		
Внешний высокочастотный генератор (HSE)	1–24 МГц	4–16 МГц	Зависит от используемого резонатора, до нескольких десятков ppm	1–2 мА
Внутренний высокочастотный генератор (HSI)	16 МГц	8 МГц	около 1%	100–250 мкА
Внутренний низкочастотный генератор (LSI)	110–146 кГц	30–60 кГц	от 20 до 50%	1–5 мкА

Генератор, работающий с внешним кварцевым резонатором, называется HSE (внешний высокоскоростной). Этот генератор используется для приложений со строгими требованиями по точности и стабильности частоты синхроимпульсов, например для систем передачи данных. Предусмотрена возможность тактирования МК внешним синхросигналом.

Приложение может выполняться в МК на высокой тактовой частоте и при использовании встроенного в кристалл генератора HSI (внутренний высокоскоростной). В этом случае внешний резонатор не используется. Этот источник синхроимпульсов имеет энергопотребление в 10 раз меньше, чем HSE, высокую точность и стабильность частоты (±1%) и может использоваться как источник для внутреннего умножителя частоты.

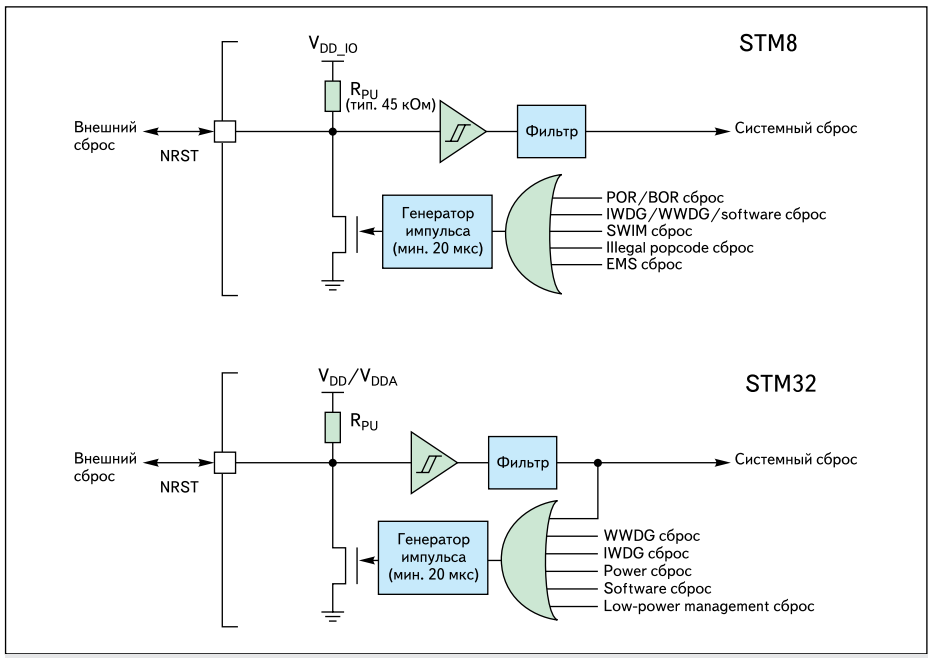

Рис. 3. Схема сброса: а) STM8; б) STM32

Очень низким собственным энергопотреблением (несколько мкА) обладает низкочастотный внутренний генератор (LSI). Его можно использовать для синхронизации периферийных узлов, обеспечивающих автоматическое пробуждение из режимов ожидания или остановки, второго внутреннего сторожевого таймера или для тактирования всего МК (в STM8). Точность LSI невысока (порядка нескольких десятков процентов), однако, как и для HSI-генератора, предусмотрена возможность его перекалибровки в приложении для компенсации технологических отклонений или дрейфа параметров по температуре.

Память

Оба семейства базируются на энергонезависимой памяти, в том числе и байты конфигурации МК.

Перечислим возможности, которые доступны на всех новых МК:

- Сброс в режимах ожидания, остановки и дежурном режиме помогает избежать зависаний, если МК входит в режим пониженного энергопотребления случайно, что полезно для приложений, не предназначенных для работы в такой конфигурации.
- Аппаратный или программный запуск сторожевого таймера — обеспечение аппаратного старта сторожевого таймера, сразу после сброса.
- Защита памяти от считывания — предотвращение пиратских заимствований содержания программ.
- Защита памяти от записи, защита части памяти, содержащей важный код. Обычно это относится к загрузочному коду или коду перепрограммирования в приложении.

Эти возможности значительно увеличивают безопасность и надежность. Приложение может восстановиться, даже если вторжение в работу МК происходит перед выборкой процессором самой первой инструкции.

STM8 и STM32 имеют встроенный загрузчик, позволяющий перепрограммировать внутреннюю энергонезависимую память через последовательный интерфейс (например, UART). Любой персональный компьютер с последовательным интерфейсом может быть использован как инструмент для программирования или обновления содержимого памяти МК. Компания ST предоставляет бесплатную программную утилиту для выполнения всех операций, поддерживаемых этим загрузчиком.

Безопасность

Автомобильная промышленность первой потребовала увеличения надежности электронных средств управления на базе МК. За этим последовали подобные запросы и от других промышленных сегментов рынка. Теперь и бытовая техника должна отвечать определенному стандарту — IEC60335-1. При использовании сертифицированных библиотек внутреннего тестирования, а также благодаря некоторым, присущим обоим семействам и перечисленным ниже аппаратным особенностям, значительно облегчается разработка изделий, отвечающих этому стандарту, по классу Б. Указанные средства способствуют значительному уменьшению времени разработки и быстрой сертификации приложений со строгими функциональными требованиями по безопасности.

Сторожевые таймеры

STM8 и STM32 имеют два сторожевых таймера:

- Сторожевой таймер оконного типа предназначен для контроля нахождения времени выполнения цикла в заданных пределах. Тактируется системным синхросигналом.
- Независимый сторожевой таймер может быть активирован для увеличения надежности системы. Этот сторожевой таймер продолжит работать даже в случае отказа основного генератора.

Мониторинг синхросигналов

Стандарт также требует обнаружения отказа кварцевого резонатора или его колебаний на нетиповых гармониках. Это достигается использованием системы синхронизации с периодическим измерением частоты внешнего резонатора. Система безопасности тактирования (CSS) контролирует работу внешнего генератора и автоматически переключает системный синхросигнал на внутренний генератор в случае отказов или отклонений.

Энергопотребление

Вдобавок к режимам пониженного энергопотребления, обеспечиваемым самим ядром, оба семейства способны уменьшать полное потребление на уровне кристалла.

Расход энергии в режимах выполнения и ожидания может быть уменьшен одним из следующих способов:

- Замедление системы: потребление может быть настроено согласно требуемой производительности приложения изменением частоты системных синхроимпульсов.
- Отключение тактирования простаивающей периферии: для минимизации динамического потребления.

Оба семейства имеют встроенные стабилизаторы напряжения 1,8 В для питания внутренней логики. Стабилизаторы имеют существенное собственное потребление (несколько десятков мкА) в режиме выполнения, для которого характерно общее потребление тока кристаллом в диапазоне десятков мА. Для уменьшения собственного потребления есть возможность использования стабилизаторов в специальном режиме, позволяющем минимизировать собственное потребление, в случае, если ток, необходимый для питания кристалла, не превышает нескольких мкА. Такая ситуация типична для режимов «остановка» или «дежурный». Специальный режим обеспечивает низкое собственное энергопотребление стабилизатора, однако время, требуемое на пробуждение МК, несколько больше, чем при работе с конфигурацией для режима выполнения.

Программная библиотека

Сходимость периферийных узлов семейств STM8 и STM32 способствует разработке межплатформенных решений и значительно помогает при переходе от одного семейства к другому. Однако когда дело доходит до времени разработки, существенным фактором является программная поддержка. Для STM8 и для STM32 доступны обширные программные библиотеки, обеспечивающие пользователю уровень аппаратной абстракции (HAL) для всех ресурсов МК. Все биты контроля/статуса покрываются соответствующей функцией на С или программным интерфейсом приложения (API).

Программная библиотека имеет три уровня и включает в себя:

1. Полную адресную карту регистров со всеми битами и полями битов, объявленными в С. С этой картой программная библиотека облегчает задачу инженера и, что еще более важно, предоставляет файл соответствий, не содержащий ошибок, что ускоряет раннюю фазу проекта.
2. Подборка подпрограмм и структур данных в форме API, покрывающая все периферийные функции. Эта подборка может непосредственно использоваться как справочная структура, потому как она также включает в себя макроопределения для поддержки особенностей ядра, общие определения констант и типов данных. Кроме того, она не привязана к типу компилятора и потому может использоваться с любым существующим или будущим инструментальным набором. API разработана с использованием автомобильного стандарта MISRA C.
3. Набор примеров, покрывающих все доступные периферийные узлы (85 примеров для семейства STM32 и 57 — для семейства STM8), с типовыми проектами для самых популярных сред разработки. При использовании стандартной оценочной платы требуется всего несколько часов для начала работы с совершенно новым микроконтроллером.

Инженер сам выбирает, как использовать библиотеку. Можно отобрать файлы, полезные для проекта, или использовать примеры для обучения и быстро оценить продукт. Можно использовать API для максимального сокращения времени разработки.

Для STM8 и STM32 предоставляются свои стандартные библиотеки. Однако для перехода от библиотеки одного семейства к другому в именах файлов, перечисленных ниже, достаточно просто заменить “stmxxx _” на “stm32f10x” или “stm8s” в зависимости от выбранного продукта.

- stmxxx_lib.h — это единственный файл заголовка, который должен быть включен в исходный текст на С, обычно в main.c. Он включает другие файлы заголовков библиотеки.
- stmxxx_conf.h — это файл конфигурации драйверов периферийных узлов, где можно определить состав периферии, который будет использоваться в приложении, и некоторые другие параметры.
- stmxxx_it.c — этот файл содержит шаблон заголовка прерываний, который должен быть заполнен, но это является уже первым шагом разработки.

После усвоения вышеупомянутых принципов и организации файлов для простого приложения на практике можно переключаться от одного продукта к другому, не обращаясь к справочному руководству.

Возьмем практический пример: SPI-периферия, сконфигурированная в привилегированном режиме и используемая для того, чтобы читать во внешней EEPROM или писать в ней.

В листингах 1, 2 приведен код инициализации (использующий программную библиотеку) для STM8 и STM32 соответственно.

```
/*————————Initialize SPI in Master mode————————*/
SPI_Init (SPI_FIRSTBIT_MSB,
SPI_BAUDRATEPRESCALER_4,
SPI_MODE_MASTER,
SPI_CLOCKPOLARITY_LOW,
SPI_CLOCKPHASE_2EDGE,
SPI_DATADIRECTION_1LINE_TX,
SPI_NSS_SOFT,
0x07); /*CRC polynomial*/
/*————————Enable SPI————————*/
SPI_Cmd (ENABLE);
```

Листинг 1. Пример кода для STM8

```
/* Private variables — -*/
SPI_InitTypeDef SPI_InitStructure;
/* ———— SPI1 Master ————*/
SPI_InitStructure. SPI_FirstBit = SPI_FirstBit_MSB;
SPI_InitStructure. SPI_BaudRatePrescaler = SPI_BaudRatePrescaler_4;
SPI_InitStructure. SPI_Mode = SPI_Mode_Master;
SPI_InitStructure. SPI_CPOL = SPI_CPOL_High;
SPI_InitStructure. SPI_CPHA = SPI_CPHA_2Edge;
SPI_InitStructure. SPI_Direction = SPI_Direction_1Line_Tx;
SPI_InitStructure. SPI_NSS = SPI_NSS_Soft;
SPI_InitStructure. SPI_CRCPolynomial = 7;
SPI_InitStructure. SPI_DataSize = SPI_DataSize_8b;
SPI_Init (SPI1, &SPI_InitStructure);
/* ———— SPI1 Master ————*/
SPI_Cmd (SPI1, ENABLE);
```

Листинг 2. Пример кода для STM32

Все параметры идентичны, а процедуры подобны: с двумя вызовами функций для инициализации и запуска. Главное различие лежит в методе, каким параметры передаются в функцию. Для STM32 используют структуру, передаваемую адресом, а для STM8 параметры передаются непосредственно, чтобы минимизировать количество оперативной памяти, используемой во время фазы инициализации.

Другое различие при использовании STM32 — возможность определения размера данных (8- или 16-разрядные) для SPI. В этом примере размер данных явно определен, однако библиотека построена таким образом, чтобы это определение могло быть опущено. Если это поле не инициализируется в структуре, то по умолчанию используется 8-разрядный размер данных, для обеспечения совместимости с семейством STM8.

Заключение

Универсальность номенклатуры 8- и 32-разрядных МК компании ST означает много общих черт двух семейств. На уровне пери-

ферии они разделяют стандартные периферийные узлы, такие как последовательные интерфейсы передачи данных и таймеры. На уровне системы имеются идентичные особенности, уменьшающие количество внешних компонентов (схемы синхронизации, схемы сброса, функции безопасности и т. д.).

Эти общие черты дополнены рядом программных библиотек, которые оказывают значительную помощь в начале разработки. Библиотеки могут также служить базисом для унифицированной платформы, поддерживающей и 8-, и 32-разрядные МК. Наконец, общие черты STM8 и STM32 в сочетании с их программными библиотеками дают возможность повторного использования разработки, ускоряют выведение продуктов на рынок, особенно если конечное изделие имеет множество модификаций с различными требованиями по производительности, количеству интерфейсов или сложности функций управления.