Микропроцессорные 32-битные ядра MIPS для высокопроизводительных встраиваемых систем

Илья АФАНАСЬЕВ
ilya@gamma.spb.ru

Для производителей микропроцессоров и микроконтроллеров рынок становится теснее, но в то же время для разработчиков электроники появляется больше продуктов для осознанного выбора. Процессорные ядра M14K/M14Kc с системой команд microMIPS и особенно новейшая разработка — ядро microAptive — дополняют существующие решения комании MIPS. Все это укрепляет позиции против ее серьезного конкурента — ARM.

Введение

На рынке 32-разрядных процессоров для встраиваемых систем действуют несколько игроков, и компании ARM и MIPS — одни из них. С начала основания компания ARM была нацелена на рынок встраиваемых систем, в то время как MIPS была сфокусирована на создании процессорных архитектур для высокопроизводительных рабочих станций и серверов. (Одно время компания MIPS была частью Silicon Graphics.) Позднее компания ARM стала стремиться к более высокой производительности, потому что именно там самый большой рынок для этой компании — рынок мобильных телефонов. Между тем MIPS обратила внимание на устройства с меньшей вычислительной мощностью, потому что там находится ее крупнейший рынок — потребительская электроника.

Архитектуры, разработанные MIPS, занимают серьезную долю рынка сетевых и телекоммуникационных устройств. Компания MIPS имеет более 125 лицензиатов своих ядер, которые совместно производят более 600 млн процессоров каждый год. Откройте корпус спутниковой приставки, телевизора, DVD-проигрывателя: в большинстве случаев вы обнаружите процессор на основе MIPS-ядра. Процессоры MIPS также широко применяются в сотовых базовых станциях, интернет-роутерах, высокопроизводительных свичах, игровых приставках и других устройствах.

Для рынка встраиваемых систем MIPS предлагает несколько ядер (рис. 1), полностью совместимых снизу вверх. Для 32-битных микроконтроллеров, созданных для решения задач в реальном времени, предназначены ядра M4K/M4KE. Дальнейшее развитие этого семейства — ядро M14K — имеет microMIPS-архитектуру сжатия набора команд, а ядро microAptiv дополнено функционалом цифровой обработки сигналов (ЦОС, DSP) с поддержкой множественного потока данных (Single Instruction Multiple Data, SIMD).

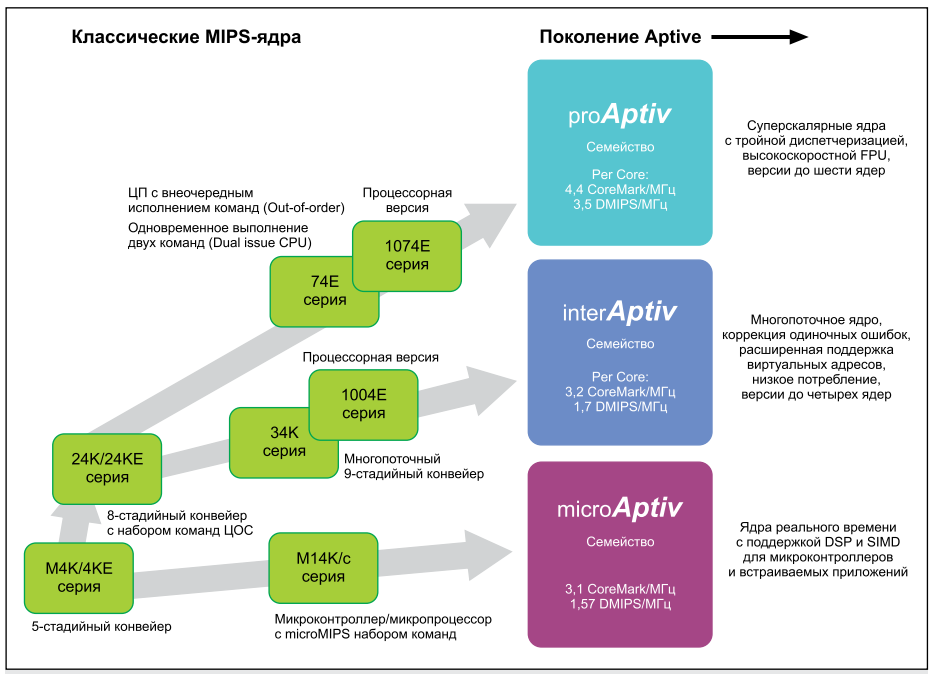

Рис. 1. Развитие 32-разрядных ядер MIPS

Сравнение ядра M4K с Cortex-M3

В 2002 году компания MIPS Technologies представила M4K — высокопроизводительное синтезируемое ядро, оптимизированное для применения в микроконтроллерах. Ядро M4K лицензировали около 30 компаний и используется в мобильных телефонах, модемах, GPS-приемниках, цифровых камерах и т. д., а также является основой микроконтроллеров общего применения PIC32MX компании Microchip Technology, Inc.

Ядро M4K разработано с набором функций, комбинация которых предоставляет лучшее в классе быстродействие и значительно превосходит предлагаемые компанией ARM ядра серии Cortex-M3.

Ядро M4K обеспечивает быстродействие 1,5 DMIPS/МГц, в то время как Cortex-M3 имеет быстродействие примерно на 20% меньше (1,25 DMIPS/МГц, как показано на веб-сайте ARM). Быстродействие ядра ARM Cortex-M0 еще меньше и обеспечивает 0,9 DMIPS/МГц. Архитектура Cortex-M0 имеет также ряд других ограничений. Для того чтобы выполнять код наравне с MIPS M4K, Cortex-M3 должен работать на частоте

на 20% выше и, соответственно, потреблять больше энергии.

Достигнутые показатели архитектуры MIPS обеспечиваются, в том числе, следующими особенностями.

Ядро M4K имеет набор из 32-битных регистров общего назначения (РОН), число таких наборов может быть 1, 2, 4, 8 и 16. Эти наборы регистров сохраняют параметры и команды в чипе, снижая тем самым накладные расходы на пересылки в памяти и освобождая командные циклы. Это положительно сказывается на увеличении производительности. Использование набора теневых РОН также увеличивает быстродействие системы за счет уменьшения накладных расходов в обработчике прерывания. При возникновении прерывания или немаскированного прерывания (исключения) ядро определит, какой теневой набор регистров будет использоваться, установит его как активный набор РОН и позволит выполняться программе с вектора прерывания. Этот процесс полностью устраняет необходимость сохранения и восстановления контекста при обработке прерываний. Это также сохраняет содержимое РОН от изменения в прерываниях, уменьшает время обработчика прерывания и сохранения РОН в ОЗУ. Следует отметить, что в реализации ядра в контроллерах PIC32MX используется два набора 32-разрядных РОН, а контроллеры с ядром Cortex-M3 имеют только один набор из 16 регистров.

Помимо 32-битных инструкций MIPS32, ядро M4K поддерживает набор команд MIPS16e, состоящий из наиболее востребованных инструкций MIPS32 в их 16-битном эквиваленте. Команды MIPS16e сжимают код в меньший размер памяти, сохраняя при этом высокий уровень производительности за счет снижения пропускной способности памяти и времени выполнения команд. Применение расширения системы команд MIPS ASE (Application Specific Extension) уменьшает размер кода до 40%.

Системный сопроцессор (System Co-Processor, CP0) — это уникальный модуль для MIPS-архитектуры, который отвечает за преобразование адресов из виртуальной памяти в физическую, систему контроля исключений, диагностику ядра, режимы работы (Kernel, User и Debug) и состояние контроллера прерываний.

Конвейер ядра в M4K имеет пять стадий, в то время как ядро Cortex-M3 оборудованно 3-стадийным конвейером. Бóльшая длина конвейера, с одной стороны, дает возможность выполнять больше инструкций в секунду, а с другой — увеличивает время реакции на прерывания. Однако, как было сказано ранее, наличие в M4K нескольких наборов РОН позволяет существенно снизить накладные расходы на сохранение/восстановление контекста в обработчике прерываний.

В ядре M4K все операции сдвигов выполняются за один цикл. Специальная логика включена в работу конвейера и предоставляет быстрый доступ к данным для использования в следующих инструкциях до того, как команда пройдет весь конвейер. Производительность улучшается в результате уменьшения числа циклов, необходимых для выполнения конкретной задачи.

Рассмотрим два примера Си-кода:

- Пример 1:

```
c = a/b;

d = c+d;
```

- Пример 2:

```
c = a/b;
mPORTESetBits(BIT_0);
z = x + y;
U1TXREG = z;
d = c+d;
```

Примечания. *Примеры скомпилированы при максимальном уровне оптимизации. Проверка проведена на контроллере PIC32MX. Оба примера выполняются ядром MIPS M4K за одно и то же время, что достигается за счет высокопроизводительной реализации модуля умножения и деления.*

Модуль MDU (Multiply Divide Unit, модуль умножения и деления) выполняет умножение 32×16 бит (или MAC-инструкцию) за один цикл, а операцию умножения 32×32 бит — за два цикла.

Модуль MDU — это выделенный вычислительный блок, имеющий собственный конвейер и работающий независимо от конвейера выполнения команд ядра. Любая команда умножения и деления попадает напрямую в MDU, а следующая инструкция (не MDU) — в конвейер ядра, тем самым обеспечивается параллельное выполнение команд без задержек. (Кроме случая, когда следующая за MDU инструкция должна использовать результат предыдущей MDU-команды.)

Так как ядро M4K имеет два конвейера, то программист имеет возможность распараллелить процессы, например пересылку данных и MDU-инструкции, и тем самым получить существенный выигрыш в быстродействии.

Вернемся к приведенным выше примерам. Второй пример отличается от первого тем, что результат деления используется не сразу, а через несколько команд. Так как операция деления выполняется за несколько итераций и в зависимости от знака и значения делимого занимает от 11 до 33 тактов, то это время можно занять командами, не использующими MDU.

Модуль MDU особенно востребован в задачах ЦОС, таких как быстрое преобразование Фурье (БПФ), реализации КИХ- и БИХ-фильтров, которые часто применяются в промышленных применениях и задачах связи.

Поддержка в M4K инструкций подсчета количества старших нулевых битов (CLZ) и количества старших единичных битов (CLO) также позволяет ускорить выполнение ЦОС-алгоритмов.

Как пример возможностей ускорения задач ЦОС, PIC32 выполняет 256-точечное 16-разрядное по основанию 4 БПФ за 22 000 циклов (283 мкс при 80 МГц тактовой частоты), что на 14% быстрее контроллера STM32 на базе ядра Cortex-M3 [1].

Лицензируемое синтезируемое ядро MIPS32 M4K нацелено на недорогие, малопотребляющие встраиваемые приложения. Задачи ЦОС также часто востребованы в такого рода приложениях. Ядро M4K может успешно выполнять как контроллер-ориентированные задачи, так и задачи, связанные с цифровой обработкой сигналов.

В сравнении со своим основным конкурентом Cortex-M3 MIPS32 M4K имеет большее быстродействие, меньшую площадь кристалла и меньшее потребление (при сборке ядра с оптимизацией по площади) [2].

При реализации по техпроцессу 180 нм M4K потребляет на 65% меньше и дает в два раза больше эффективности потребления энергии, чем Cortex-M3 при той же тактовой частоте [1].

В таблице 1 сведены параметры по оценке физических параметров архитектур MIPS M4K и ARM Cortex-M3 при реализации на одинаковых техпроцессах. Интересным фактом является то, что оптимизированное по площади ядро M4K при той же тактовой частоте 100 МГц, что и оптимизированный по быстродействию Cortex-M3, имеет на 75% меньше площадь кристалла и на 58% меньшее потребление.

Таблица 1. Сравнение ядер MIPS M4K и ARM Cortex-M3

Ядро	M4K		Cortex-M3	
DMIPS/МГц	1,5		1,25	
CoreMark/МГц	3,11		1,06–1,9	
Оптимизация ядра	По быстродействию	По площади	По быстродействию	По площади
Максимальная тактовая частота, МГц	228	100	135	50
Площадь кристалла, мм^2	0,64	0,185	0,74	0,38
Типовая мощность, мВт/МГц	0,214	0,066	0,165	0,084

Несколько слов о ядре Cortex-M0

Cortex-M0 реализует вариант ARM-архитектуры, называемой ARMv6: она предшествовала той, что реализована в Cortex-M3. Ядро Cortex-M0 имеет 3-стадийный конвейер и даже меньшую производительность (0,9 DMIPS/МГц), чем Cortex-M3. Ядро Cortex-M0 работает с обоими типами инструкций (Thumb и Thumb-2) общим числом 56, и только шесть из них — 32-битные инструкции. Ядро Cortex-M0 не поддерживает атомарных инструкций. В большинстве случаев код, написанный для Cortex-M3, не будет выполняться на Cortex-M0 без модификаций.

Ядро Cortex-M0 вернулось к фон-неймановской архитектуре, той же, что реализована в ARM7. Cortex-M0 не поддерживает локаль-

ную память, вместо этого доступ к коду и данным осуществляется из основной памяти через AHB-шину, что существенно снижает производительность за счет дополнительных циклов ожидания, требуемых до завершения передачи данных.

И в завершение: ядро Cortex-M0 содержит порядка 24 000 вентилей. Однако, несмотря на сравнительно малый размер этого ядра, в Cortex-M0 отсутствуют многие возможности, которые являются стандартными в оптимизированном ядре M4K, которое содержит порядка 33 000 вентилей, причем производительность Cortex-M0 существенно ниже (0,9 DMIPS/МГц против 1,5 DMIPS). Таким образом, незначительный выигрыш в площади кристалла дает существенные потери в производительности и функциональности.

PIC32MX больше, чем M4K

Одним из лицензиатов процессорных архитектур MIPS является компания Microchip Technology, Inc. На основе ядра MIPS M4K она производит семейство контроллеров PIC32MX, краткие характеристики которых приведены в таблице 2.

Синтезируемые ядра MIPS позволяют создавать микроконтроллеры и добавлять ряд пользовательских функций. Разрабатывая PIC32MX — сложную систему на кристалле (System-on-Chip, SoC), компания Microchip внесла ряд функциональных улучшений относительно базового ядра M4K. К таким улучшениям относятся кэш инструкций, шинная матрица, контроллер прерываний и интерфейс отладки.

Буфер и кэш предвыборки инструкций

В PIC32 приложение может выполняться как из внутренней флэш-памяти, так и из внутреннего ОЗУ, которое можно динамически разделить на области программ и данных.

В высокоскоростных процессорах узким местом, снижающим быстродействие, является Flash-память программ, с ее ограничениями на время доступа. Семейство PIC32MX имеет флэш-память с 128-битной шиной, которая позволяет за одно обращение выбирать четыре 32-битные инструкции, что соответствует выборке из флэш-памяти с учетверенной скоростью. Однако даже при наличии 128-битного буфера предвыборки выполнение инструкций с тактовой частотой ядра не представляется возможным, так как реальное приложение содержит ветвления и повторяющиеся инструкции, а также осуществляет чтение данных из флэш-памяти.

Поэтому в архитектуру PIC32MX был введен конфигурируемый кэш предвыборки, состоящий из 16 128-битных строк, из них четыре строки могут использоваться в качестве кэша данных, что полезно при обработке массивов данных.

Кэш предвыборки выполняет две задачи: кэширование инструкций, к которым осуществляется доступ, и предвыборка инструкций из флэш-памяти, до того как они необходимы для исполнения. Каждая строка кэша содержит признак, по которому можно понять, что хранится в строке, и адреса памяти, команды из которых находятся в кэше. Обычно строки кэша содержат копию участка флэш-памяти, данные из которой доступны ядру без задержек.

Использование кэша предвыборки позволяет выполнять линейный код на максимальной частоте тактирования без состояний ожидания. Этому способствуют две линии кэша с адресной маской, которые могут содержать повторяющиеся инструкции, а также механизм предикативной выборки инструкций.

На многих задачах с плавающей точкой PIC32MX работает существенно быстрее, чем, например, Cortex-M3. Диаграмма на рис. 2 демонстрирует нормированное, приведенное к одной частоте время выполнения алгоритмов работы с плавающей точкой на контроллерах Cortex-M3 (Thumb2) относительно PIC32MX. (Источник: www.smxrtos.com. Данные для Cortex-M3 приведены для STM32F103VBT6 и компилятора Keil v4.13; для LM3S8962 и компилятора IAR v5.20; для PIC32 используется GNU-компилятор.)

Диаграмма показывает, что при использовании библиотек плавающей точки в компиляторах Keil и IAR архитектура Cortex-M3 существенно проигрывает MIPS32 M4K, и в частности, контроллерам PIC32MX.

Контроллеры PIC32 являются лидерами в своем классе по производительности. Одно и то же приложение занимает меньше памяти и выполняется быстрее, что позволяет уменьшить тактовую частоту процессора и снизить потребление.

Шинная матрица

Процессоры PIC32MX имеют две раздельные шины: для выборки инструкций и для выборки данных.

Для подключения периферийных устройств PIC32MX содержат две внутренние шины. Одна из них соединяет большинство периферийных модулей. Другая — высокоскоростная — соединяет контроллер прерываний, контроллер прямого доступа к памяти (ПДП), канал внутрисхемной отладки и шину высокоскоростной периферии.

Шинная матрица соединяет ведущие устройства (также называемые инициаторами) с ведомыми устройствами. Процессоры PIC32MX имеют до пяти инициаторов и три ведомых (флэш-память, ОЗУ и низкоскоростная периферия).

Инициаторы, всегда присутствующие в PIC32, — это шина инструкций, шина дан-

Таблица 2. Семейство контроллеров PIC32MX

Микроконтроллер	Тактовая частота, МГц	Размер памяти флэш/RAM, кбайт	Число выводов	UART/SPI/I²C	USB	CAN	Ethernet	PPS	I²S/AC97
PIC32MX1xx	40/50	16–128 / 4–32	28–44	2/2/2	–	–	–	Да	Да
PIC32MX2xx	40/50	16–128 / 4–32	28–44	2/2/2	FS Host/OTG	–	–	Да	Да
PIC32MX3xx	100	32–512/ 16–128	64–100	2/2/2	–	–	–	Да	Да
PIC32MX4xx	100	64–512/ 16–128	64–100	2/2/2	FS Host/OTG	–	–	Да	Да
PIC32MX5xx	80	64–512/ 16–64	64–100	6/3/4	FS Host/OTG	Да	–	–	–
PIC32MX6xx	80	64–512/ 32–128	64–100	6/3/4	FS Host/OTG	–	Да	–	–
PIC32MX7xx	80	128–512/ 32–128	64–100	6/4/5	FS Host/OTG	Да	Да	–	–

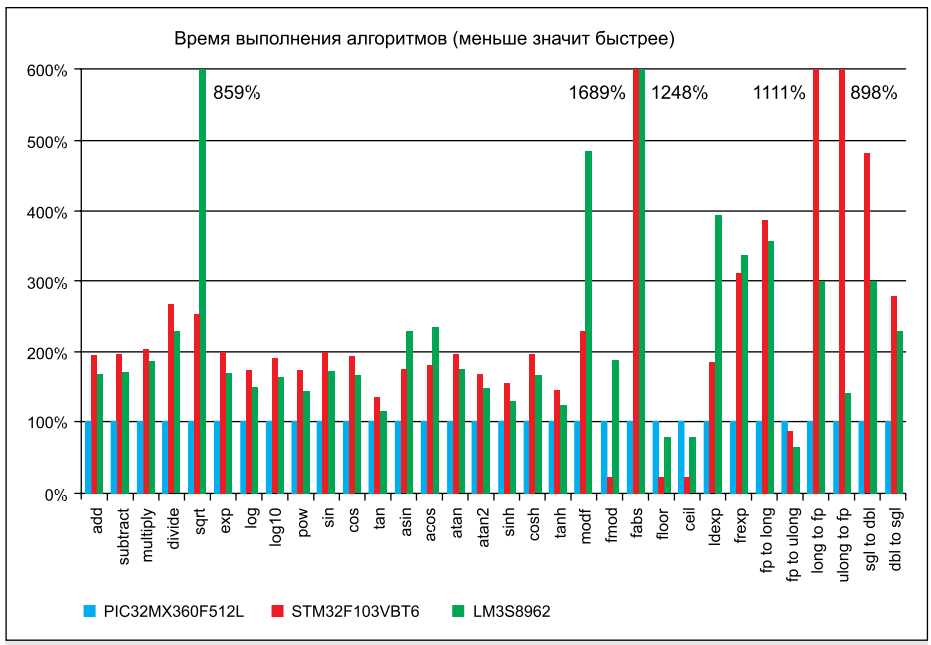

Рис. 2. Сравнение времени выполнения алгоритмов с плавающей точкой для PIC32 и контроллеров на базе Cortex-M3

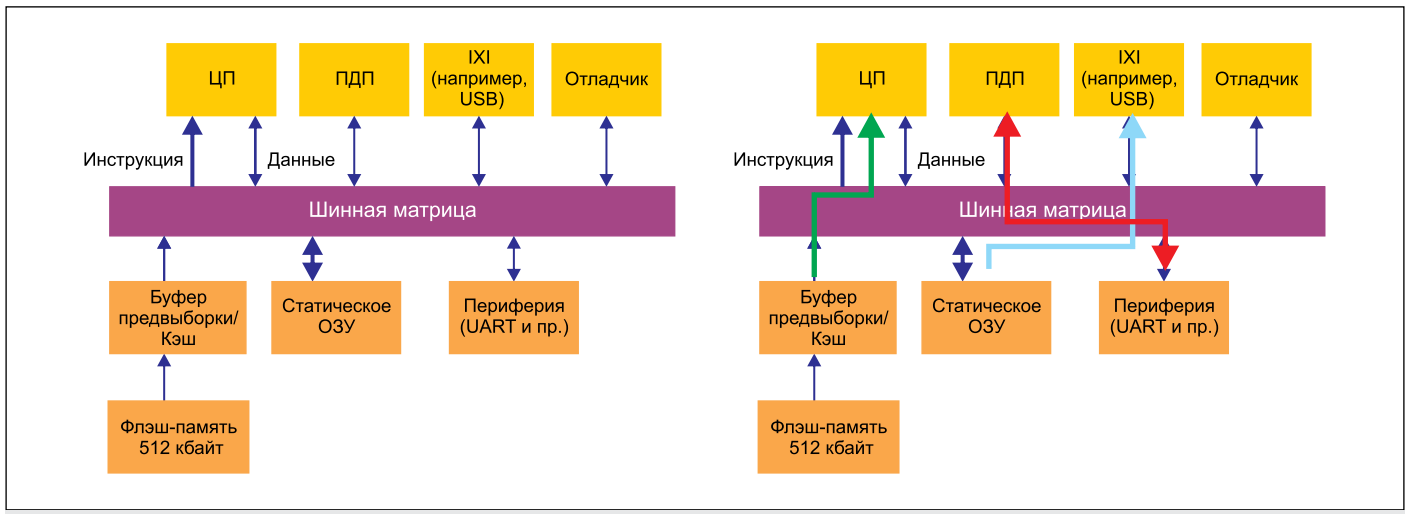

Рис. 3. Обеспечение параллельной работы шинной матрицы

ных, шина внутрисхемного отладчика и контроллер ПДП. Некоторые PIC32MX также содержат интерфейс расширения инициаторов, которые подключают высокоскоростную периферию — USB, Ethernet и т. п.

Шинная матрица является специальным переключателем, который обеспечивает одновременный множественный доступ к различным ведущим устройствам на шинах, которые обращаются к разным адресатам. На рис. 3 приведен пример, где шинная матрица обеспечивает параллельный доступ к различным адресатам: ядро процессора извлекает команду из флэш-памяти, контроллер ПДП соединен с низкоскоростной периферией (UART), а высокоскоростной интерфейс USB подключен к ОЗУ.

Если необходим одновременный доступ более чем одного инициатора к одному и тому же ведомому, шинная матрица организует последовательный доступ с помощью трех режимов арбитража. Режимы арбитража устанавливают уровни приоритетов каждого из инициаторов.

DMA-интерфейс

Контроллер ПДП предназначен для передачи данных между блоками памяти и периферийными модулями без участия ядра контроллера.

Семейство PIC32MX имеет до восьми идентичных каналов ПДП, которые можно использовать как для передачи данных между периферией и памятью, так и для переноса данных между блоками памяти. Дополнительно в PIC32MX присутствует до восьми выделенных каналов ПДП для работы с высокоскоростной периферией (USB, Ethernet, CAN).

Контроллер ПДП позволяет обеспечивать транзакции на уровне слов и байтов. В последнем случае выравнивания по слову данных не требуется. Арбитраж доступа осуществляется на основании фиксированных приоритетов каналов.

Два канала ПДП могут быть объединены в цепочку: после окончания передачи ведущего канала автоматически запускается ведомый канал.

Каналы могут работать в двух адресных режимах: нормальном и расширенном. В нормальном режиме объем передаваемых данных ограничен 256 байтами, но допустима транзакция по невыровненному адресу и передача типа «память — периферия». В расширенном режиме адресации объем передаваемых данных может достигать 64 кбайт.

Контроллер ПДП имеет в своем составе модуль вычисления циклического избыточного кода (CRC), который может быть подключен к любому каналу. Модуль позволяет вычислять CRC любой разрядности с произвольным полиномом.

Набор теневых регистров

В процессорах PIC32MX реализовано два набора РОН. Второй набор предназначен для использования с высокоприоритетными прерываниями. Этот дополнительный набор РОН также называется теневым набором регистров.

Когда происходит высокоприоритетное прерывание, процессор автоматически переключается на теневой набор РОН без вмешательства программы. Это снижает задержку в обработчике прерывания на сохранение контекста и уменьшает время реакции на прерывание. Набор теневых регистров управляется регистрами системного сопроцессора, а также контроллером прерываний.

Контроллер прерываний

Ядро M4K поддерживает три варианта реализации контроллера прерываний: совместимый с ранней реализацией MIPS32 Release 1; векторный и внешний контроллер прерываний. Векторный контроллер прерываний в M4K поддерживает только восемь приоритетных прерываний, поэтому в SoC PIC32MX реализован внешний векторный контроллер прерываний, аналогичный применяемым в 16-разрядных контроллерах Microchip (PIC24 и dsPIC).

Контроллер прерываний в PIC32MX обладает следующими характеристиками:

- Время реакции на прерывание: не более пяти тактов генератора.
- До 96 источников прерываний.
- До 64 векторов прерываний.
- Каждый вектор прерывания может иметь приоритет от 1 до 7.
- Каждый вектор прерывания может иметь дополнительный приоритет от 0 до 3.
- Теневой набор РОН для обслуживания высокоприоритетного прерывания.
- Конфигурируемое положение векторов прерываний.
- Конфигурируемая дистанция между векторами прерываний.
- Возможность программной генерации любого прерывания.
- Таймер отложенных прерываний.

В контроллере прерываний PIC32MX векторов прерываний меньше, чем источников, поэтому часть источников прерываний используют один вектор. Как правило, это прерывания одного периферийного модуля. Приоритеты назначаются пользователем не источнику прерывания, а вектору.

Векторы с приоритетом 7 могут использовать дополнительный набор РОН, что позволяет снизить время входа в функцию обработки прерывания, так как в этом случае контекст сохранять не требуется.

Интересной особенностью контроллера прерываний является наличие таймера отложенных прерываний. Для использования этого таймера необходимо установить его период (32-битный регистр) и приоритет прерывания таймера. При возникновении запроса на прерывание все источники с приоритетом ниже или равным приоритету таймера запустят этот таймер. Как только значение таймера будет равно нулю, установится флаг источника прерывания, запустившего таймер.

Процессоры PIC32MX являются системой на кристалле. Аналогично и процессоры на базе Cortex-M3 нужно рассматривать не как «голое» ядро, а как систему со своими особенностями. Рассмотрим отличия контроллеров прерываний, реализованных в SoC PIC32, и некоторых из систем на кристалле на базе Cortex-M3 (C-M3).

Ядру Cortex-M3 требуется 12 циклов для сохранения контекста. Однако в практических реализациях процессоров ситуация несколько иная. Так, в SoC на базе C-M3 обработчику прерываний нужно от 18 до 28 циклов на вход в прерывание (рис. 4). Время входа в обработчик прерывания состоит из шести циклов реакции на прерывание и синхронизации периферийной шины SoC; автоматического сохранения контекста ядра C-M3 (сохранение регистров r0–r3, r12, LR, PC) за 12 циклов и полного сохранения контекста — еще 10 циклов (сохранение регистров r4–r11, r13, r14).

При обработке высокоприоритетного прерывания, как было сказано ранее, в PIC32MX используется набор теневых РОН, поэтому не требуется сохранение и восстановление контекста. На вход в прерывание нужно только 11 циклов (рис. 5).

При обработке низкоприоритетных прерываний процессору PIC32 требуется 28 циклов (рис. 6).

Время обработчика низкоприоритетных прерываний состоит из:

- 5 циклов реакции на прерывание и синхронизации периферии;
- 5–6 циклов для автоматического сохранения регистров ядра;
- 13 циклов на сохранение контекста.

Только простой обработчик прерываний выигрывает от автосохранения контекста в Cortex-M3. Компиляторы Си автоматически сохраняют контекст при вызове функций и использовании регистров, поэтому общее время обработчика прерывания становится больше.

Рассмотрим случай, когда начата обработка одного прерывания и поступает второе прерывание с таким же приоритетом. Автосохранение контекста в SoC Cortex-M3 в таком случае неэффективно (рис. 7), поэтому применяется специальный механизм цепочек.

Механизм цепочек уменьшает задержку между обработкой двух прерываний, запрещая автосохранение контекста (рис. 8).

Архитектура PIC32MX не требует специальных механизмов для обработки нескольких прерываний. Всего три дополнительных цикла необходимо для обработки двух прерываний в многовекторном режиме (рис. 9).

Интерфейс отладки ICSP

Помимо стандартного для архитектуры MIPS интерфейса JTAG, в PIC32 добавлен интерфейс отладки ICSP, присутствующий

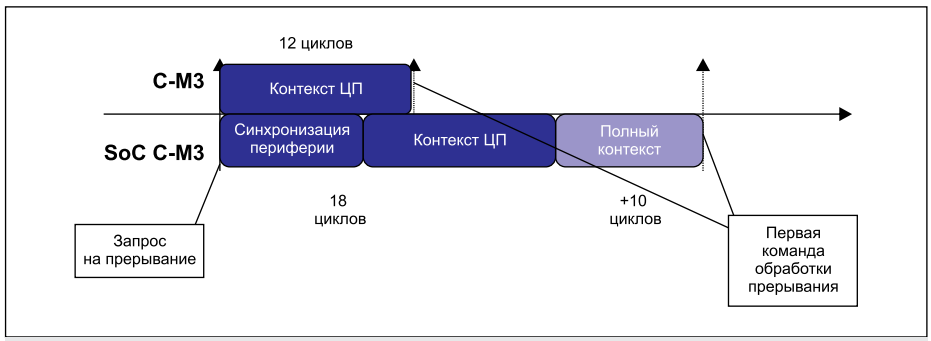

Рис. 4. Различие во времени входа в прерывание для «голого» ядра Cortex-M3 и практических реализаций контроллеров на базе Cortex-M3

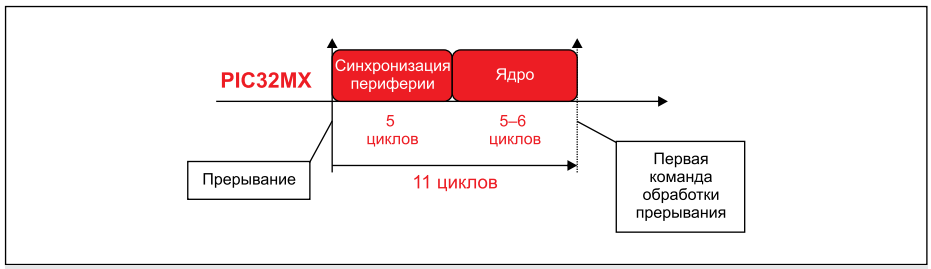

Рис. 5. Обработка высокоприоритетного прерывания в PIC32MX

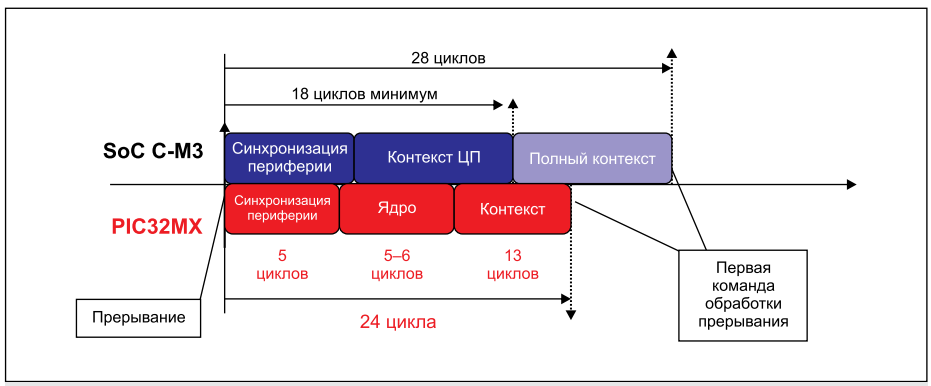

Рис. 6. Сравнение обработчика прерываний для PIC32MX и контроллеров на базе Cortex-M3

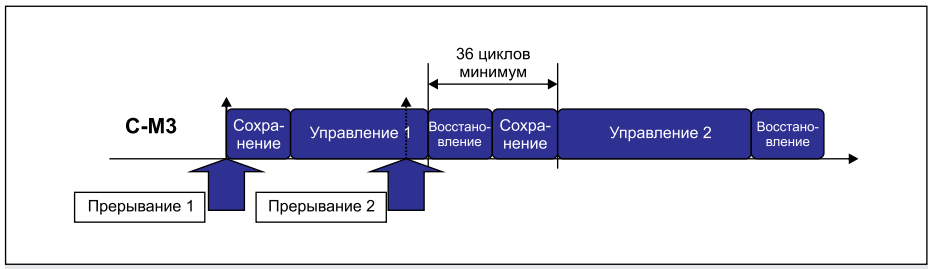

Рис. 7. Вариант автосохранения контекста без использования цепочек

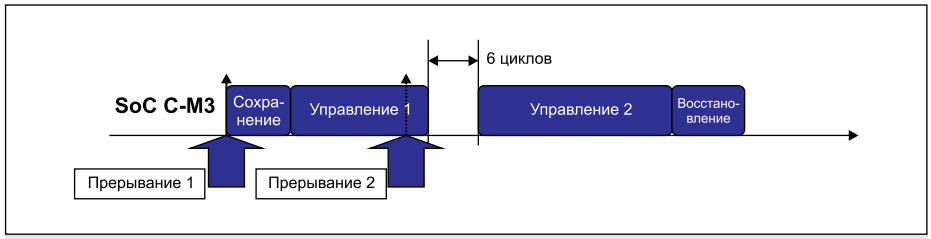

Рис. 8. Механизм цепочек в SoC Cortex-M3

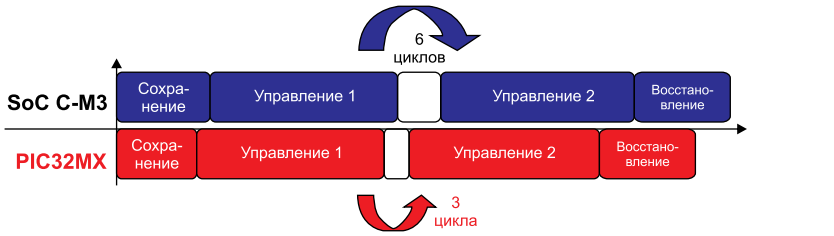

Рис. 9. Сравнение обработки последовательных прерываний в PIC32MX и SoC Cortex-M3

во всех PIC-микроконтроллерах Microchip, что обеспечивает совместимость отладочных средств и программаторов с остальными семействами контроллеров. За счет этого разработчики могут переходить от одного семейства контроллеров этого производителя к другому, не меняя отладочные средства.

Для тех же, кто имеет JTAG-отладчик, остается возможность отладки через этот интерфейс.

Заключение

В следующей части статьи мы рассмотрим дальнейшее развитие платформы MIPS для встраиваемых устройств и решения задач цифровой обработки сигналов: ядро MIPS microAptiv, его сравнение с Cortex-M4, а также основанное на MIPS microAptiv следующее поколение микроконтроллеров Microchip — PIC32MZ с производительностью до 330 DMIPS.

Литература

1. Beyond the Hype: MIPS — the Processor for MCUs. MIPS Technologies, Inc.
2. An Independent Analysis of the MIPS Technologies MIPS32 M4K Synthesizable Processor Core. Berkeley Design Technology, Inc.
3. www.microchip.com
4. www.arm.com