КОММУНИКАЦИОННЫЙ ПРОЦЕССОР ДЛЯ УНИВЕРСАЛЬНЫХ КОМПЬЮТЕРНЫХ ИНТЕРФЕЙСОВ

В.Петров

С появлением современных высокоскоростных последовательных интерфейсов казалось, что наступает новая эра – эра выносных изделий. Но стоит только начать работать с этими интерфейсами, и эйфория проходит. Для поддержки высокоскоростных интерфейсов необходима значительная вычислительная мощность как внешнего устройства, так и компьютера. Эта мощность отбирается от основной задачи тем сильнее, чем выше скорость передачи данных. Нельзя сказать, что ситуация неразрешима, но выход из нее требует значительных усилий от программистов и нестандартных решений от разработчиков аппаратуры. Проблему призвана решить новая концепция построения автономных и выносных устройств сбора данных и цифровой обработки сигналов фирмы "Инструментальные системы".

Не всегда можно объединить компьютер и дополнительное устройство в одном блоке. Например, в профессиональной звукозаписи мешает шум от вентиляторов, моторов дисковода и клавиатуры компьютера. В системах сбора данных с АЦП возможны наведенные помехи со стороны электронных компонентов компьютера. Подобные проблемы способны решить выносные устройства, но при этом интерфейсы "компьютер–усторойство" ограничивают скорость обмена данными. До недавнего времени при проектировании удаленных изделий, требующих высокоскоростного обмена с компьютером, предпочтение отдавали шинам ISA, PCI и VME как высокоскоростным и сравнительно простым в использовании. Но сегодня акценты смещаются в сторону скоростных последовательных интерфейсов, которыми оснащаются современные компьютеры. Прежде всего – это интерфейсы стандартов USB и IEEE 1394, хотя есть и другие. Последовательная универсальная шина USB работает сейчас на скорости 12 Мбит/с, а шина IEEE 1394 – до 400 Мбит/с. С внедрением новых версий этих интерфейсов скорость возрастет до 480 и 3200 Мбит/с, соответственно, что сравнимо с производительностью таких параллельных шин, как PCI, VME и т.п. Однако преимущества последовательных интерфейсов сталкиваются со сложностью их реализации.

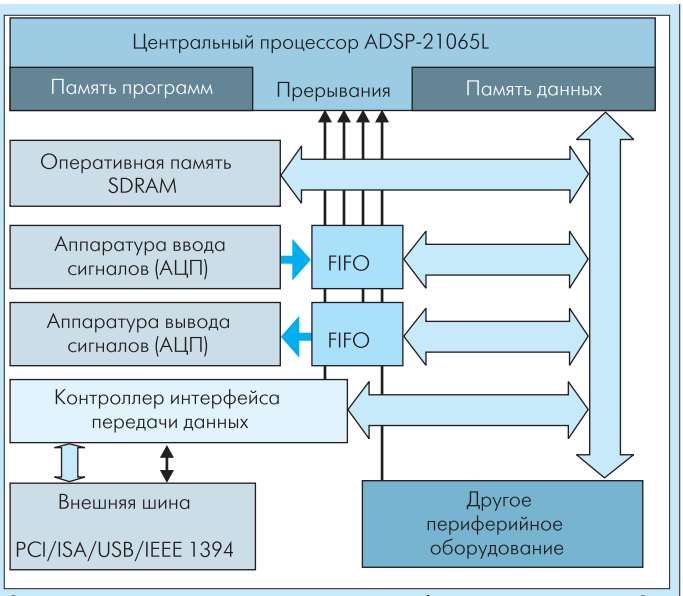
В фирме "Инструментальные системы" в течение года проводился эксперимент по использованию шин USB и IEEE 1394. В результате были созданы такие изделия, как анализатор телефонных сигналов в каналах связи (АТСКС) с шиной IEEE 1394 и процессором TMS320C6701, а также ряд выносных устройств сбора данных с субмодулями АЦП и интерфейсами USB и IEEE 1394 (по выбору). Полученный опыт привел к появлению новой концепции построения таких систем. В дальнейшем она станет основой всех выносных изделий фирмы, избавив разработчиков аппаратуры от бремени программирования интерфейсов и позволив свободно изменять вид интерфейса. Работа программиста упростится, как если бы он имел дело с шинами ISA или PCI. И хотя новая концепция не является принципиально неожиданной, она кардинально меняет специфику и возможности изделий самой фирмы "Инструментальные системы".

ПРОБЛЕМЫ ПОДДЕРЖКИ ПОСЛЕДОВАТЕЛЬНЫХ ИНТЕРФЕЙСОВ

Как правило, разработчик аппаратуры стремится возложить все функции устройства (вычислительные, управляющие и связные) на один процессор. Основные аргументы в пользу такого решения – простота реализации однопроцессорных устройств и высокое быстродействие современных процессоров. Очевидно, что в нашем случае процессор одновременно выполняет по крайней мере две задачи – сбора данных и их передачи. Заранее рассчитать распределение вычислительных ресурсов процессора между ними не всегда возможно. Процессы, сопровождающие обмен данными, становятся понятными только после некоторых исследований и экспериментов. С чем столкнется разработчик при создании однопроцессорного устройства сбора и передачи данных (рис.1)?

Прежде всего, только для редких обращений (RS232) можно пренебречь длительностью цикла чтения (записи) контроллера интерфейса в периферийном устройстве. В случае USB или IEEE 1394 это становится существенным ограничителем и делает работу сколь угодно быстрого процессора неэффективной. Например, для ряда контроллеров RS232 и даже USB операция записи (чтения) может занять до 330 нс при 15-нс цикле процессора (например, ADSP-21065L). Обращение к контроллеру интерфейса замедляет работу ядра процессора и его шины. В результате для передачи данных по шине USB со скоростью 1 Мбайт/с в контроллер USB в течение 1 мс будет записано не менее 1024 байт, а займет это около 340 мкс процессорного времени – 30% от всей производительности шины. Данную проблему можно преодолеть не всегда и исключительно за счет усложнения математического обеспечения либо аппаратуры.

Обращения к контроллерам интерфейсов не ограничены только операциями записи или считывания данных, они требуют специальных действий управления и контроля со стороны процессора. Для одних интерфейсов (например, IEEE 1394) таких действий сравнительно немного, а для других (USB) их больше. Удобнее всего эти действия выполнять в программе обработки прерываний. Пока код создают на языке ассемблера (что трудоемко), дело с прерываниями обстоит сносно, но все меняется с применением языка высокого уровня, например Си. В этом случае входы/выходы в прерывание



Оцифрованные данные проходят через буферную память FIFO и накапливаются в ОЗУ (SDRAM). Встроенная в процессор память служит промежуточным буфером в режиме DMA.

Рис.1. Однопроцессорное устройства сбора данных компании "Инструментальные системы"

занимают до 5 мкс. Поскольку сами прерывания могут наступать через каждые 50–100 мкс, что определяется темпом обмена по интерфейсу, временные затраты получаются существенными.

Все современные протоколы обмена данными используют пакетную пересылку, что предполагает подтверждение приема пакетов. Передатчик, не приняв такое подтверждение, обязан повторить отправку. Значит, где-то нужно хранить копию уже переданных данных. Следовательно, интенсивность пересылок по шине процессора возрастает вдвое (от источника в буферную память, затем из буферной памяти в контроллер интерфейса), а при потере пакета – еще больше. Поскольку на шине присутствует несколько параллельных и независимых потоков данных, соответственно возрастает сложность пересылок и объем необходимой памяти.

При формировании пакетов преобладают непрерывные последовательности операций записи или считывания, которые наиболее эффективно реализовывать в режиме прямого доступа к памяти (DMA). Однако несовпадение темпов возникновения данных на стороне передатчика, поглощения их на стороне приемника и скорости самой передачи требует применения особых методов DMA. У программиста возникают серьезные трудности в выборе алгоритма и в его реализации.

В простейшем случае (один источник данных, один передатчик и один приемник), когда все операции на интерфейсе можно выстроить в одну очередь, особых проблем с пересылкой нет. Только в этом режиме однопроцессорные системы эффективны. Но с усложнением структуры обмена необходима организация очередей и таймерных ожиданий. В этом случае цель, ради которой проектировалось устройство, уходит на второй план, а усилия программистов концентрируются на решении связной задачи. Теперь именно она диктует правила и ограничивает возможности, поскольку требует много ресурсов процессора.

И, наконец, следует сказать, что использование USB и IEEE 1394 предполагает, что устройство должно выполнять действия, важные для этих интерфейсов, но никак не для прикладной задачи. Кроме того, от устройства может потребоваться и поддержка работоспособности сети. Это означает, что в прикладную программу должен быть встроен модуль, который нельзя останавливать и даже замедлять. Для него потребуется дополнительное место в оперативной и постоянной памяти.

Из сказанного видно, какой клубок проблем должен распутать разработчик, пожелавший применить в своих изделиях интерфейсы USB, IEEE 1394 или любой аналогичный. Упростить ситуацию можно, сделав независимыми прикладную задачу и задачу передачи данных.

ОСНОВНАЯ ФУНКЦИЯ – ОСНОВНОМУ ПРОЦЕССОРУ, СВЯЗНАЯ ФУНКЦИЯ – СВЯЗНОМУ!

Если какую-либо задачу можно разбить на несколько независимых подзадач, то их разумно решать по отдельности, на самостоятельных процессорах. Пусть в рамках одной задачи будут выполняться все действия, связанные с работой устройства как элемента сети (коммуникационная задача), а в рамках другой – действия по решению прикладной задачи устройства (вычислительная задача). Эти задачи могут быть решены как на одном, так и на нескольких процессорах. С точки зрения гибкости и простоты наиболее оптимален двухпроцессорный вариант. Проведя линию раздела между двумя задачами, мы получили коммуникационный процессор (КП) с одной стороны и вычислительный процессор (ВП) – с другой (рис.2).

Таким образом, вычислительный процессор занимается решением проблем, связанных с функциональным назначением устройства, не отвлекаясь и не расходуя ресурсы на передачу данных в компьютер (в сеть). Последняя задача – функция коммуникационного процессора. Очевидно, что структуры данных, с которыми работают ВП и КП, принципиально различны: для КП они определяются внешним интерфейсом, для ВП – спецификой прикладной задачи. При передаче через последовательную шину данные сгруппированы в пакеты различных видов, дополнены служебной информацией, снабжены заголовками – вся эта дополнительная информация ВП не нужна, с ней работает КП. По одну сторону от КП присутствуют данные с внутренним форматом интерфейса КП–ВП, по другую – данные с форматом внешнего интерфейса (USB, IEEE 1394 и т.д.). Поэтому главную цель КП определим как механистическое преобразование передаваемых данных путем смены их форматов. Для нас неважно, какие именно форматы данных на внешнем интерфейсе, какова структура КП – лишь бы выполнялось требуемое преобразование. Вся специфика КП – в интерфейсе, разделяющем его и ВП. Рассмотрим особенности этого интерфейса, опираясь на многоуровневую модель взаимодействия открытых систем ISO – ВОС (OSI).

СТРУКТУРЫ ДАННЫХ

Как правило, структура данных известна только на начальном и конечном пунктах и не важна для процесса передачи. Например, дан-

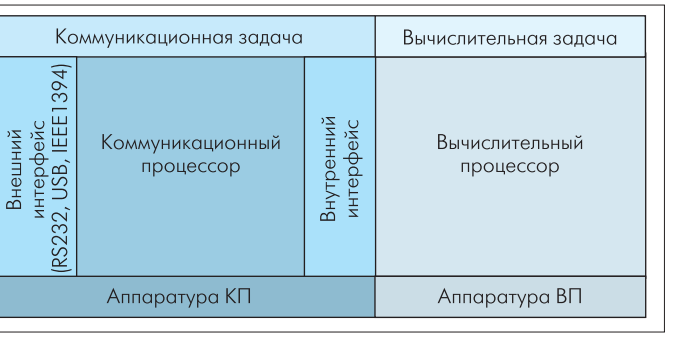
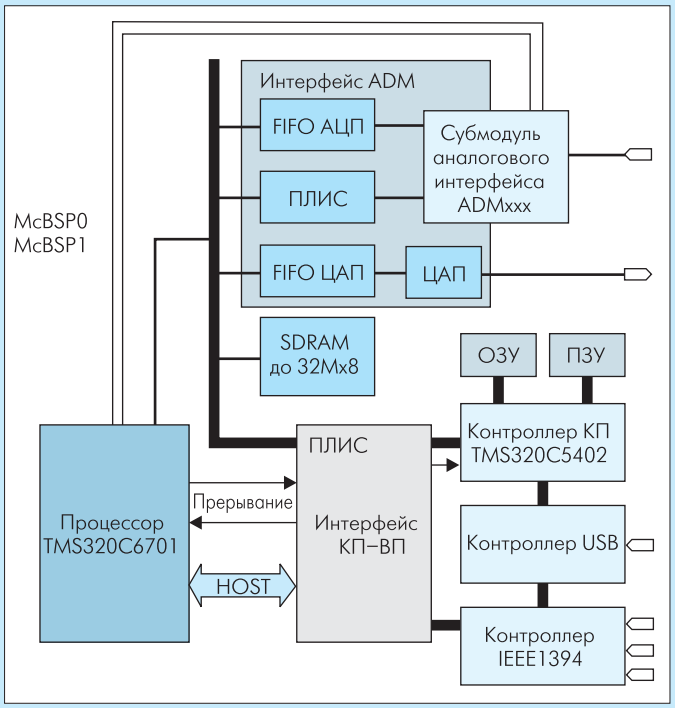


Рис.2. Устройство с разделением задачи между вычислительным и коммуникационным процессорами



ADM – внутренний интерфейс фирмы "Инструментальные системы". McBSP0 и McBSP1 – последовательные порты. Процесс сбора и обработки (фильтрация, преобразования, сжатие и т.п.) контролирует ВП. Данные поступают в процессор и после обработки накапливаются в синхронной динамической памяти (SDRAM). ВП не имеет собственных элементов долговременного хранения, поэтому программу в него загружает коммуникационный процессор, используя для этого собственное ПЗУ и порт HOST. Через этот порт КП может обратиться и ко всем другим элементам общей шины для технической поддержки изделия.

Рис.3. Конструкция устройства с КП компании "Инструментальные системы"

ные оцифровываются АЦП, передаются по интерфейсу USB на компьютер, а там записываются на диск. Получатель данных, естественно, знает об их структуре. Всем же остальным участникам процесса (аппаратуре и программам) знать эту структуру не обязательно и достаточно записать на диск данные в той последовательности, в какой они появлялись на выходе АЦП. При желании можно использовать один канал для передачи данных различного формата (например, 16- и 8-разрядных АЦП). Кроме того, объем передаваемых данных за единицу времени может меняться, а передача – даже приостановиться. Главное, чтобы был соблюден порядок и направление этих данных. Назовем такие данные потоковыми, а процесс их передачи – потоками.

Кроме передачи потоковых данных, в рамках прикладной задачи необходимо управлять другими устройствами сети, передавая различные команды посредством пакетного обмена через внешний интерфейс. Поскольку структура командных данных отличается от потоковых, разумно передавать их по интерфейсу КП–ВП через отдельный – командный – канал.

Таким образом, в режиме передачи данных на интерфейсе КП–ВП должны быть каналы для передачи потоковых и командных данных. Для передачи команд ограничимся одним каналом, а для потоков будем организовывать столько каналов, сколько необходимо в текущей задаче (конечно, в разумных пределах).

ФИЗИЧЕСКИЙ УРОВЕНЬ ВНУТРЕННЕГО ИНТЕРФЕЙСА

Физически интерфейс КП–ВП может быть реализован на отдельной ИС, например – на ПЛИС, как это сделано в устройствах компании "Инструментальные системы" (рис.3). Конкретная организация интерфейса зависит от применяемых процессоров. Архитектура КП не должна влиять на структуру внутреннего интерфейса, чего не скажешь о ВП – ведь это может быть абсолютно любой процессор со своим уникальным набором прерываний и каналами DMA. В состав интерфейсной схемы обязательно входят командный и несколько входных/выходных портов (рис.4). Подчеркнем, что со стороны ВП конструкция портов всех типов не зависит от особенностей КП и от внешнего интерфейса.

Команды

Некоторые команды служат для диалога между ВП и КП и не выходят из устройства. Другие команды управляют удаленными устройствами, следовательно, транслируются через внешний интерфейс в пакетном режиме. Для USB – это пакеты Control, позволяющие передать команды различным источникам, для IEEE 1394 – асинхронные пакеты, которые выполняют операции адресного считывания и записи. Разумеется, прежде чем попасть на удаленное устройство, команды видоизменяются в КП.

На интерфейсе КП–ВП командный обмен происходит по согласованному между КП и ВП протоколу через двунаправленный командный порт с двухпортовым ОЗУ. Ресурсами этого ОЗУ управляет исключительно КП. Инициирует обмен только ВП посредством простейших операций чтения и записи. Для передачи команды он записывает ее в двухпортовое ОЗУ, а затем сообщает об этом КП аппаратным прерыванием. Прием команды КП подтверждает встречным прерыванием. Если у КП есть командные сообщения для ВП, он не может переслать их самостоятельно. Сначала он сигнализирует об этом прерыванием, а затем ВП в удобный для себя момент считывает сообщение из двухпортового ОЗУ. Обмен через командный порт должен быть максимально надежным. Для этого в командах ис-

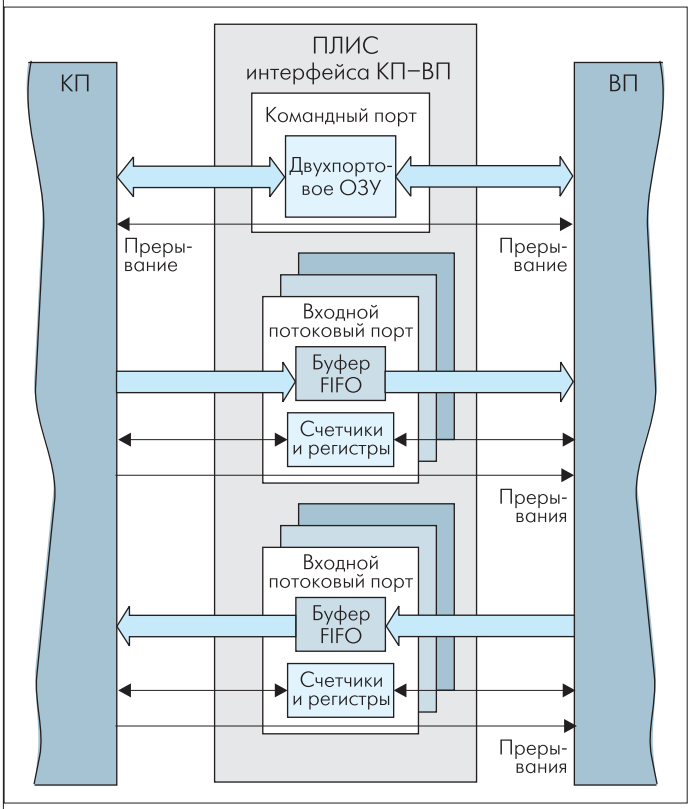


Рис.4. Структура интерфейса КП–ВП

пользуется нумерация, адресация и таймирование обращений (подсчет времени существования с целью обнаружения "зависших" обращений и освобождения занимаемых ими ресурсов). Запросы и ответы разделены по времени, что позволяет ввести обмен с использованием прерываний, именно в те моменты, когда ВП не сильно загружен прикладной задачей. Командный порт, хотя и обязателен для интерфейса КП–ВП, может не использоваться в конкретных задачах передачи данных.

Потоки

Для реализации потоков на интерфейсе КП–ВП организованы однонаправленные порты, в которые ВП последовательно записывает (или считывает) данные посредством различных механизмов, например контроллеров DMA. Естественно, за портом находится буфер (как правило, FIFO), выравнивающий потоки для КП и ВП по скоростям. Размер буфера зависит от пиковой скорости передачи и пауз между передачами в конкретных задачах. По конструкции порты аналогичны друг другу. Их число может наращиваться. Об особых ситуациях, возникших в потоке, ВП извещается специальным прерыванием по потоковому порту. Для каждого потокового порта в ИС интерфейса КП–ВП предусмотрен набор регистров, куда записываются свойства потока, определенные на связном уровне интерфейса. Эти регистры доступны для ВП в циклах записи и считывания.

СВЯЗНОЙ УРОВЕНЬ ВНУТРЕННЕГО ИНТЕРФЕЙСА

В общем случае связной уровень определяет структуру пакетов, которыми обмениваются устройства. Здесь происходит формирование пакетов и сопровождение их дополнительной информацией. Пакет со связного уровня передатчика через физический уровень поступает на связной уровень приемника, где от него отделяется служебная информация.

Пакеты потоковых данных организованы наиболее просто. Со стороны ВП их размер зависит от специфики процессора, его каналов DMA и программы. На интерфейсе КП–ВП пакеты выстраиваются в несколько параллельных непрерывных потоков, для каждого – свой порт. В пакетах нет никаких разделителей и заголовков, их роль играет управляющая информация (например, размер пакета) в регистрах портов. Эту информацию – свойства потока – в регистры динамически записывают КП и ВП. Кроме направления передачи и пропускной способности, необходимо учитывать еще три свойства.

Гранулированность данных. Предположим, в потоке поочередно передаются данные от нескольких источников. При передаче через внешний интерфейс часть данных может пропасть. Если потеря допустима, она не должна приводить к невосстанавливаемому смещению данных. Это означает, что синхронно с данными должны передаваться и специальные метки (окрашивание). При пакетной передаче можно отказаться от меток, выравнивая данные по началу пакета. Если ВП использует потоки с гранулированными данными, то КП транслирует их с синхронизацией – окрашиванием или упаковкой в соответствии с размером и положением гранул.

Прогнозируемость потоков. КП должен знать, сколько данных он может забрать из порта для передачи без "разрушения" внешнего интерфейса или сколько их может разместить в порте при приеме. Этот параметр в большей степени зависит от всей совокупности выделенных для потока ресурсов, а также (в меньшей степени) от размеров FIFO.

Управляемость потоков. Во время передачи потока по внешнему интерфейсу возникают коллизии, которые могут исказить данные, замедлить или вовсе прекратить передачу. ВП иногда должен вовремя на это прореагировать. КП сигнализирует ВП о внешних коллизиях при помощи аппаратного прерывания. Получив такое прерывание, ВП должен обратиться за дополнительной информацией к КП через командный порт.

Перечисленные свойства играют важную роль при формировании пакетов для передачи через внешний интерфейс. Так, КП пакетирует данные в соответствии с типом внешнего интерфейса и выделенного в нем трафика, учитывая свойства гранулированности и прогнозируемости потока. Эти свойства передаются только от ВП к КП независимо от направления потока и асинхронно с данными. КП, в свою очередь, сообщает ВП состояние канала связи (свойство управляемости), выставляя аппаратное прерывание. Приняв его, ВП через командный порт запрашивает у КП информацию, считывает ее и, проанализировав, совершает соответствующие действия.

Для командного порта пакет устроен сложнее. Кроме данных в нем присутствует заголовок, содержащий адрес приемника, адрес передатчика, номер транзакции, размер данных, тип операции и другую необходимую информацию. Всей передачей управляет ВП, он определяет и направление обмена. Обмен пакетами – полудуплексный.

Считается, что данные при передаче по внутреннему интерфейсу исказиться не могут, поэтому в пакетах внутреннего интерфейса нет контрольной суммы.

УРОВЕНЬ ТРАНЗАКЦИЙ ВНУТРЕННЕГО ИНТЕРФЕЙСА

Уровень транзакций обслуживает маршрутизацию и коммутацию пакетов. Для потоковых портов на этом уровне никаких действий не происходит – для каждого потока определен свой порт. Для командного же порта на уровне транзакций процедуры со стороны КП и ВП различны.

При передаче от КП к ВП данные одного сообщения могут быть объемными, поэтому КП разбивает их на пакеты, нумерует и ставит в очередь. О наличии пакетов в очереди КП информирует ВП аппаратным прерыванием. Каждый пакет ожидает считывания определенное время, после чего исключается из очереди и теряется. Приняв пакет, ВП должен присоединить его содержимое к принятым ранее данным, а если необходимо, то и обработать, не дожидаясь завершения всей передачи.

В обратном направлении пакеты отправляет ВП. Чтобы не произошла потеря пакетов, когда КП не в состоянии их принять, на уровне транзакций в двухпортовой памяти командного порта организован семафор, разрешающий отправку данных лишь при готовности КП.

ЧТО МОЖНО И ЧЕГО НЕЛЬЗЯ ТРЕБОВАТЬ ОТ КП?

Коммуникационный процессор как элемент сети способен решать три глобальные задачи. Это – поддержка работоспособности внешнего интерфейса, техническое обслуживание всего изделия и передача данных как по внешнему, так и по внутреннему интерфейсу.

Поддержка работоспособности внешнего интерфейса подразумевает, что КП выполняет все действия, обязательные для узла сети в соответствии с ее спецификацией. Например, последовательности инициализации и конфигурации. Шина IEEE 1394, в отличие от USB, допускает работу без центрального компьютера. Поэтому для IEEE 1394, кроме процедур пересылки данных асинхронными и изохронными пакетами, КП должен поддерживать функции узлов управления и контроля работоспособности шины (Bus Manager, Isochronous Resource Manager и Power Manager).

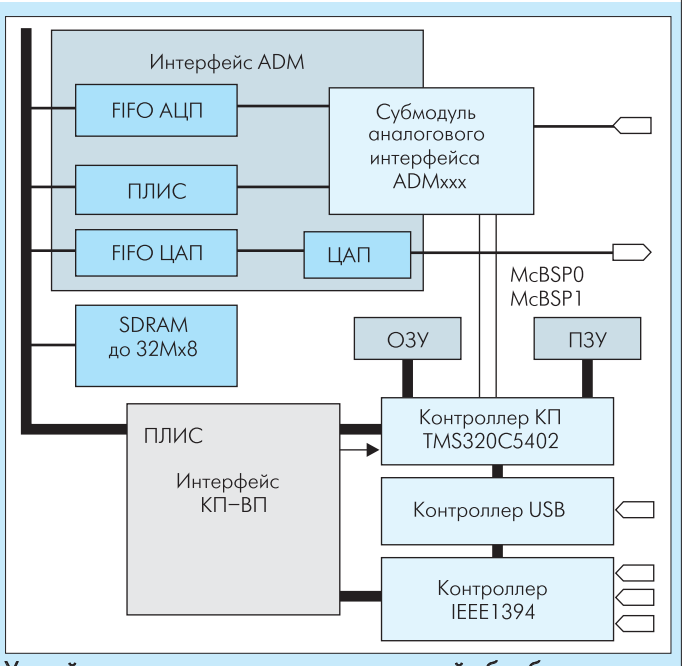
Задача технического обслуживания включает начальную загрузку из ПЗУ программируемых ресурсов устройства; автоматическую загрузку из ПЗУ и запуск программы для ВП; перезагрузку

ресурсов и программы из компьютера по внешнему интерфейсу; перепрограммирование фрагментов ПЗУ; тестирование работоспособности изделия и его частей через интерфейс JTAG или путем прямого обращения к ресурсам.

Передача данных – то, ради чего создан КП. В этом процессе неизбежно участвует ВП. В ряде случаев, особенно на начальной стадии разработки или для вспомогательных нужд, интерфейс КП–ВП можно игнорировать, воспользовавшись альтернативными возможностями, реализованными в КП, – например, прямой записью из КП в адресное пространство ВП, используя порт HOST или захват шины. Эти функции упрощают процедуру обмена, но жертвуют другими свойствами (например, надежностью или функциональностью). В КП фирмы "Инструментальные системы" реализованы такие альтернативные функции обмена, как непосредственный доступ в адресное пространство ВП, обмен через дополнительный буфер (двусторонний FIFO) и обмен через дополнительное двухпортовое ОЗУ.

Кроме того, КП может поддерживать простые, не относящиеся к обмену данными, но часто требуемые функции. Например, службы энергонезависимого времени, формирования электропитания, слежения за потреблением и температурным состоянием устройства, сторожевой таймер.

Нередко возникает соблазн возложить на КП обработку ситуаций, выходящих за рамки нижних трех уровней модели ВОС. Например, восстановление потерянных данных в потоке, используя резервные каналы интерфейса. Однако функциональная нагрузка КП должна быть ограничена, иначе он станет чрезмерно сложным, дорогим и, как следствие, ненадежным. Поэтому концептуально КП не поддерживает процедуры, выходящие за рамки этих трех уровней. Но соответствующие механизмы можно реализовать силами ВП через внутренний интерфейс КП–ВП. Указанное ограничение касается только связных функций как наиболее ресурсоемких, а не дополнительных задач технического обслуживания и нестандартного обмена. Более того, КП может выполнять простые процедуры обработки данных, и, если позволяет задача, обходиться без ВП (рис.5).



Устройство не предназначено для сложной обработки данных. Процесс сбора контролирует КП, он же выполняет несложную обработку (например, сжатие). Данные накапливаются в SDRAM. Они доступны КП через интерфейсную ПЛИС. Часть ресурсов подключена к КП напрямую, через последовательные порты McBSP0 и McBSP1. Компания "Инструментальные системы" использует такую архитектуру только в собственных изделиях сбора данных, не предназначенных для программирования сторонними пользователями.

Рис.5. Устройство сбора данных без ВП

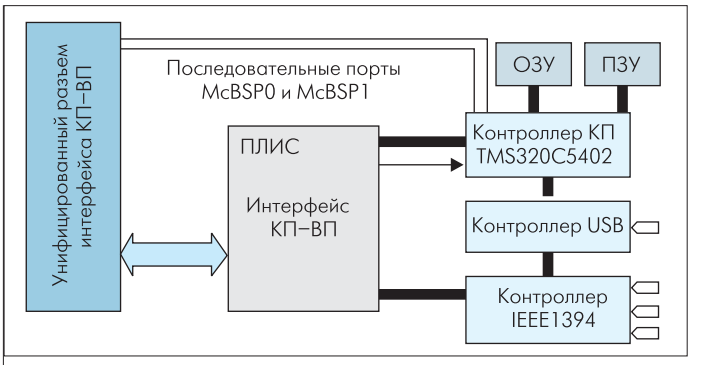


Рис.6. Модуль КП для подключения к аппаратуре сторонних изготовителей

ПОЧЕМУ НУЖНО ИСПОЛЬЗОВАТЬ КП

На первый взгляд, применение КП ведет к усложнению аппаратуры. Но поскольку работу интерфейса КП–ВП обеспечивает ПЛИС, усложнение не существенно. Конструкция КП одинакова во всех применениях – вся специфика ВП или задачи отражается лишь на типе и конфигурации интерфейсной ПЛИС. Компания "Инструментальные системы" предполагает два варианта работы с КП. Во-первых, может быть предложена принципиальная схема КП и необходимые программные ресурсы, включая конфигурацию ПЛИС. Кроме того, КП планируется выпускать в виде отдельного субмодуля со стандартизированным разъемом (рис.6). Разработчику, который решил интегрировать в свое изделие КП, достаточно подключить его к своему проекту (на стадии разработки принципиальной схемы либо как готовое изделие посредством специального разъема), выбрать соответствующую задаче интерфейсную ПЛИС и соблюсти правила работы с интерфейсом КП–ВП.

Проектируя изделие с КП, нет смысла предусматривать отладочные точки, разъемы и даже загрузочные ПЗУ – все это может предоставить КП через встроенные функции технического обслуживания. Если КП работоспособен, он непосредственно управляет отладкой с помощью подключенного через интерфейс компьютера. При подаче питания КП в автоматическом режиме выполняет комплекс проверок и программирование ВП, в том числе и загрузку "боевой" программы. Далее можно с компьютера перепрограммировать любой элемент ВП и данные стартовой загрузки.

Поскольку КП проще остальных частей изделия и оснащен отладочным интерфейсом JTAG, привести его в рабочее состояние несложно. Ремонт изделий также упрощается, выявление прогнозируемых неисправностей легко автоматизировать.

Для программиста важно сконцентрироваться на основной задаче, используя для передачи данных простые и однотипные операции, особенно когда задача достаточно сложна и нет возможности осваивать хитрости интерфейсов USB, IEEE 1394 и им подобных. КП позволяет изменять внешний интерфейс передачи данных, не трогая (даже не транслируя заново) математическое обеспечение основной задачи. Работа программиста остается пока наиболее трудоемкой и дорогой в любом проекте, использующем процессор. Поэтому от изделий с коммуникационным процессором выигрывает даже не столько прикладной программист, сколько организация, использующая его труд. ❍