Самый простой интерфейс для PCI

При отладке полнофункционального интерфейса для шины PCI важно знать, как на практике работает его ядро. Самый простой интерфейс, который может представлять практический интерес как самостоятельное устройство или как стартовое устройство для дальнейшей отладки, можно сделать довольно быстро.

Такой интерфейс не включает в себя конфигурационное пространство и может не поддерживать всех типов доступа к данным. Вот несколько схем интерфейсов, состоящих практически из одного ядра, которое выполняет последовательность операций по управлению передачей данных при работе шины (рис. 1, рис. 2 и рис. 3):

Итак, самый простой из ведомых интерфейсов можно сделать на одной микросхеме ПЛМ 22V10 или 29MA16. Этот интерфейс можно использовать только для «внутренних» нужд, так как теоретически эта схема работать не должна. Причин несколько:

1. Существуют рекомендованные микросхемы для работы с шиной PCI. Их входные и выходные характеристики существенно отличаются от характеристик «обычных» микросхем ПЛМ. Предлагаемые микросхемы 22V10 и 29MA16 не входят в этот список, но их характеристики близки к характеристикам рекомендованных микросхем.
2. Схема не стандартна. По условиям стандарта, устройство PCI должно поддерживать работу со всей областью памяти, со всеми типами доступа и иметь конфигурационное пространство.

Эта схема не выполняет ни одного из требований.

И тем не менее на практике эта схема успешно работает и часто бывает очень полезной.

Нужно признать, что делать такие устройства нехорошо и безграмотно. Однако недостаток «простых» методов работы с шиной PCI тормозит ее освоение разработчиками. Поэтому, выбирая из двух зол меньшее, предлагаем рассмотреть одно из самых простых устройств, которые работают с PCI.

Вот его плюсы:

1. Смонтировать схему можно за 15 минут без специальной аппаратуры.
2. Его стоимость равна стоимости одной микросхемы ПЛМ, монтажной платы PCI и стоимости одного рабочего дня инженера.
3. Оно отлаживается одним осциллографом, без логического анализатора.
4. В течение нескольких часов можно убедиться, что этот контроллер PCI работает.

Структурная схема устройства показана на рис. 4.

Из всех сигналов шины PCI в схеме задействованы сигналы:

AD[32..0] — мультиплексированная двунаправленная шина адреса и данных (address/data).

C/BE#[3..0] — шина команд и разрешения байт данных (command/byte enable). При выставлении адреса на этой шине появляется код операции, которая сейчас будет выполняться. В нашем случае работает передача в порт, с кодом 0010 на чтение из порта (команда процессора in) и 0011 на запись в порт (команда процессора out). При выставлении данных низкий уровень сигнала означает, что данные, передаваемые в соответствующем байте, имеют смысл.

CLK — тактовая частота. Основной выигрыш в скорости на PCI достигается с помощью синхронной передачи данных, которая, как известно, быстрее асинхронной. В идеале передача слова на шине занимает 2 такта: такт адреса и такт данных, то есть 66 нс. Асинхронная шина ISA передает слово за 400...800 нс — примерно в 10 раз медленнее. Каждый новый такт на PCI начинается по фронту сигнала CLK. В этот момент все остальные сигналы на шине защелкиваются и анализируются.

FRAME# — сигнал начала операции на шине (cycle frame). По низкому уровню этого сигнала начинаются любые операции по передаче данных. Сигнал удерживается до конца операции.

IRDY# — готовность ведущего (initiator ready). Данные будут удерживаться на шине и не будут посланы до тех пор, пока оба участника пересылки данных — ведущий и ведомый — не установят свои сигналы готовности.

TRDY# — готовность ведомого (target ready).

```
TITLE   PCI SLAVE Starter ChipPATTERN  REC.PDSCHIP REC PAL29MA16
;-------------------------------------------------------------------------------
                                        ; Часть схемы со стороны шины PCI
                        PIN 1       CLK     ; Тактовая частота шины PCI. Все процессы
                                        ; идут синхронно с этой частотой
PIN  13      CLKK     ; то же. См. документацию на м/сх.
PIN  2       /FRAME_   ; Начало кадра
PIN  23      /IRDY_   ; Сигнал готовности инициатора
PIN  8       /DEVSEL_   ; Сигнал выбора устройства
PIN  22      /TRDY_   ; Сигнал готовности устройства
PIN [4..7]     CB[0..3]  ; Выбор команды или байт данных
PIN [14..17]   AD[11..8]  ; Шина адреса/данных
                                        ; Часть схемы со стороны внутреннего интерфейса
PIN [18..21]   D[0..3]   ; Внутренняя шина данных (передаются
                                        ; четыре бита старшего байта)
                                        ; Шина однонаправлена. Два бита 0,1 только читаются.
                                        ; Два бита 2,3 только записываются.
PIN  9       rwStBit   ; Вспомогательный бит чтения/записи
PIN  10      SELECT    ; Вспомогательный сигнал выбора устройства
PIN  3       STORE     ; Вспомогательный сигнал защелкивания данных
; Выводы PIN 8,9,10,11,13 не используются;
Адресная строка. Уравнение адреса выставлено на любой портовый адрес
; в диапазоне 300-3FFSTRING PORT_HIT   '/AD[11] * /AD[10] * AD[9] * /AD[8] * /CB[3] * /CB[2] * CB[1]'
;-------------------------------------------------------------------------------
; Уравнения:
EQUATIONS;
Сигнал выбора устройства выставляется при попадании в портовое адресное
; пространство, когда инициатор не готов, и удерживается до окончания сигналов
; готовности инициатора IRDY_ или до окончания кадра FRAME_.
SELECT       := PORT_HIT * FRAME_ * /IRDY_  + ((IRDY_ + FRAME_) * SELECT)
; Тристабильный выход выбора устройства управляется сигналом SELECT.
DEVSEL_      = VCCDEVSEL_.TRST    = SELECT
; Одновременно с сигналом выбора устройства выставляется сигнал
; готовности устройства
TRDY_        = VCCTRDY_.
TRST    = SELECT;
В начале цикла устанавливается вспомогательный бит чтения/записи.
; Этот бит управляет записью данных из шины PCI или чтением в PC
IrwStBit     := CB[0]
rwStBit.CLKF    = FRAME_
; Данные на внутреннюю шину D3 и D2 записываются из PCI и защелкиваются триггером
; до следующей перезаписи.
D[2]         := AD[10]D[3]         := AD[11]                    STORE        = /IRDY_ * rwStBit * SELECT ;
D[2..3].CLKF   = STORE
; Состояние сигналов на внутренней шине данных читается в шину PCI
AD[8]        = D[0]
AD[9]        = D[1]
AD[8..9].TRST = /rwStBit * SELECT;
;-------------------------------------------------------------------------------
; Моделирование:SIMULATION
; Запись данных. Здесь сигналы на шине выставляются и снимаются командой SETF
SETF /CLK /CLKK /FRAME_ /IRDY_ /D[0] /D[1]
SETF CB[0] CB[1] /CB[2] /CB[3] /AD[11] /AD[10] AD[9] /AD[8]
SETF D[0] /D[1]
SETF FRAME_ /IRDY_
SETF CLK CLKK
SETF /CLK /CLKK /FRAME_ IRDY_ AD[11]
SETF CLK CLKK
SETF /CLK /CLKK /IRDY_
SETF CLK CLKK
SETF /CLK /CLKK
SETF CLK CLKK
SETF /CLK /CLKK
SETF CLK CLKKSETF /CLK /CLKK
SETF CLK CLKK
SETF /CLK /CLKK
SETF CLK CLKK;
Чтение данных
SETF /CLK /CLKK /FRAME_ /IRDY_ /D[0] /D[1]
SETF /CB[0] CB[1] /CB[2] /CB[3] /AD[11] /AD[10] AD[9] /AD[8]
SETF D[0] /D[1]
SETF FRAME_ /IRDY_
SETF CLK CLKK
SETF /CLK /CLKK /FRAME_ IRDY_
SETF CLK CLKK
SETF /CLK /CLKK /IRDY_
SETF CLK CLKK
SETF /CLK /CLKK
SETF CLK CLKK
;-конец файла-;
```

DEVSEL# — выбор устройства (device select). Ведомое устройство, увидев свой адрес на шине, должно откликнуться, выставив сигнал DEVSEL#.

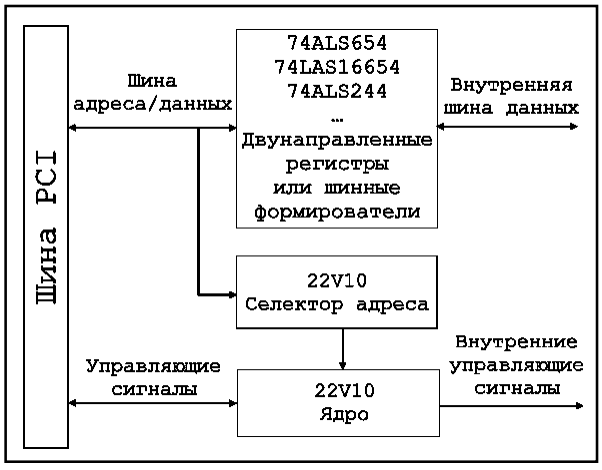

Рис. 1. Интерфейс для работы с 32-разрядной шиной. Самый полный среди минимальных вариантов — интерфейс с отдельным селектором адреса и шинными формирователями. На базе такой схемы можно сделать плату с 32-разрядной внутренней шиной данных и с 32-разрядным дешифратором адреса

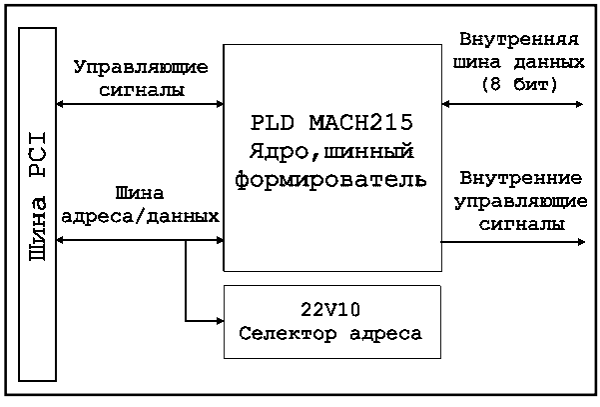

Рис. 2. Интерфейс для работы с 8-разрядной шиной. Для отладки микропроцессоров и программируемых устройств, которые имеют шины типа MCU, I2C для микропроцессорного управления, удобен контроллер без отдельных шинных формирователей на базе микросхемы ПЛМ MACH215, которая производится фирмой AMD

Цикл чтения или записи на шине происходит, как показано на рис. 5.

Цикл начинается с выставления сигнала FRAME#. Одновременно с FRAME# выставляются сигналы на шинах адреса AD и команды C/BE#, которые удерживаются до тех пор, пока от устройства, адрес которого выставлен, не придет отклик — сигнал DEVSEL#. Если такого устройства нет, система выставляет этот сигнал сама и заканчивает цикл. После того, как выбранное устройство откликнулось, передаются данные. Они выставляются на шину AD со стороны PCI или со стороны устройства, и удерживаются до тех пор, пока не будет выставлена готовность ведущего и готовность ведомого. Данные, как и все другие сигналы, защелкиваются по фронту сигнала CLK.

Временная диаграмма работы нашего интерфейса показана на рисунке 6, а его принципиальная схема — на рисунке 7.

Внутренняя структура ПЛМ описана на языке PALASM. Компилятор PALASM выпускается фирмой AMD. Ранние его версии распространялись фирмой бесплатно. Также было выпущено несколько бесплатных компиляторов с похожим синтаксисом языка другими фирмами, например, Intel. С помощью компилятора исходный текст необходимо превратить в файл JEDEC. Этот файл является стандартным для любого программатора, который поддерживает микросхемы 22V10 и 29MA16.

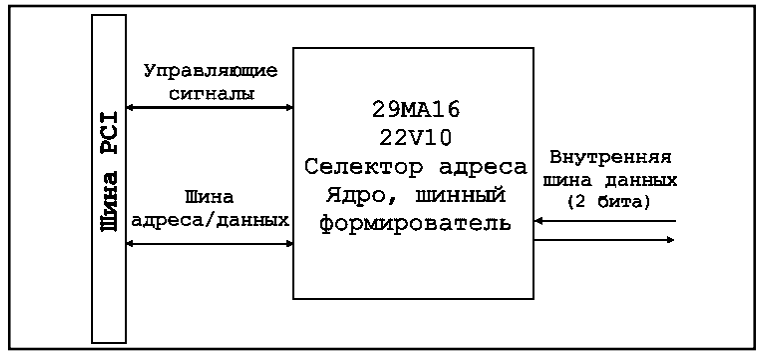

Рис. 3. Интерфейс для работы с 2-разрядной шиной. Эта схема может лишь декодировать портовые адреса PCI, не конфликтуя с основными устройствами на системной плате, устанавливать на двух битах внутренней шины данные, записанные процессором в порт, и считывать обратно в процессор данные, выставленные на двух других битах внутренней шины

Нужно заметить, что описанный выше интерфейс не поддерживает потоковой пересылки данных. Эта пересылка показана на рис. 5: выставляется адрес, а за ним в течение одного цикла следует серия данных. Эту функцию можно дописать, взяв ПЛМ с большим количеством регистров, однако обычно в этом необходимости не возникает. Дело в том, что процессор Pentium при обмене данными с периферийными устройствами, подключенными к PCI, выполняет команды mov, in или out. Все эти три команды процессора выполняют пересылку только одного слова данных. То есть на шине появляется адрес, потом появляется одно слово данных и после этого пересылка прекращается. Это снижает скорость передачи данных. Например, скорость циклического вывода в порт для PCI около 10–12 мегабайт в секунду. При этом, когда процессор циклически выполняет команду out, на шине PCI начинается и завершается пересылка, а потом идут около 10 пустых тактов до выполнения процессором следующей команды out. Для того чтобы организовать потоковую передачу данных, необходимо, во-первых, запрограммировать контроллер прямого доступа к памяти (при этом нужно учесть, что пересылае-

```
C:\>debug                 Вызываем отладчик MS-DOS
-a 100                    Записываем программу, начиная со смещения 100 в некотором сегменте памяти (пусть, 1772)
1772:0110 mov dx,320      Назначаем адрес порта для вывода
1772:0103 mov ax,aaaa     Назначаем данные для вывода (Код AAAA в двоичном виде выглядит как 10101010, и удобен для тестирования устройства и просмотра
                          на осциллографе. Также удобны коды 5555, 0000, FFFF)
1772:0106 mov cx,ffff     Записываем максимальное число в счетчик циклов
1772:0109 out dx,ax       Выводим данные из регистра AX в порт, адрес которого хранится в регистре DX. Команда ввода данных из порта — in ax, dx.
1772:010A loop 109        Данные выводятся в порт до тех пор, пока на будет исчерпан счетчик CX. В нашем случае CX = ffff = 65536 циклов.
1772:010C nop             Пропуск для возможной модификации программы.
1772:010D nop
1772:010E nop
1772:010F nop
1772:0110 mov ah,0b       Один раз в CX циклов проверяем, была ли нажата клавиша.
1772:0112 int 21          Это делается с помощью 21-го прерывания MS-DOS функции 0B. Детальное описание прерывания можно найти в программе «TechHelp!».
1772:0114 cmp al,0        Проверяем, была ли нажата клавиша
1772:0116 jz 100                Если нет, продолжаем циклически выводить в порт данные
1772:0118 int 20          Если клавиша была нажата, выходим из программы с помощью 20-го прерывания MS-DOS.
1772:011A                 Заканчиваем ввод программы.
-n outport.com            Назначаем имя файла, в который программа будет записана.
-rcx                      Записываем в регистр CX длину программы
CX 0000
:1a                       В нашем случае длина программы 1A (= 0100-011A)
-w                        Записываем программу на диск
Writing 0001A bytes       Сообщение об успешной записи
-g                        Запускаем программу
                          Щелкаем по клавише
Program terminated normally  Читаем сообщение о нормальном завершении программы
-q                        Выход из отладчика

C:\>debug outport.com     Эту программу можно повторно использовать с отладчиком.

C:\>outport.com           Или запускать из операционной системы. Программа нормально работает во всех ОС Windows, кроме NT.
```

мые данные не считываются контроллером ни с самой PCI, ни с ISA, которая обычно подключена к PCI) и, во-вторых, запрограммировать контроллер PCI процессора. Это довольно сложные операции и их выполняют только драйверы графических адаптеров, написанные известными фирмами. Все остальные, желая получать данные по шине со скоростью до 60 мегабайт в секунду, ставят на устройство, подключенное к PCI, интерфейс типа «мастер». Например, многие контроллеры SCSI, обмениваясь данными, при включении компьютера объявляют себя мастерами. В тот момент, когда нужно получить поток данных, процессор посылает в контроллер специальную команду. По этой команде контроллер захватывает шину PCI, выставляет заранее известный адрес памяти, где находятся требуемые данные, получает их потоком и передает эти данные в SCSI-устройство.

После того как исходный текст откомпилирован и записан в ПЛМ, можно тестировать готовое устройство. Для того чтобы удобнее видеть все сигналы на двухлучевом осциллографе, нужно написать маленькую программу, циклически посылающую в порт некоторые данные. Необязательно при этом пользоваться компилятором C. Можно вызвать в MS-DOS отладчик debug и в нем написать ассемблерный код.

Программа, показанная ниже, работает в 16-битовом режиме. Она чрезвычайно проста и поможет быстро понять, работает ваша плата или нет. Программа нормально работает независимо от типа компьютера и типа локальной шины (долгое время она использовалась для отладки устройств на шинах ISA). Естественно, предполагается работа на ПК с процессорами Intel.

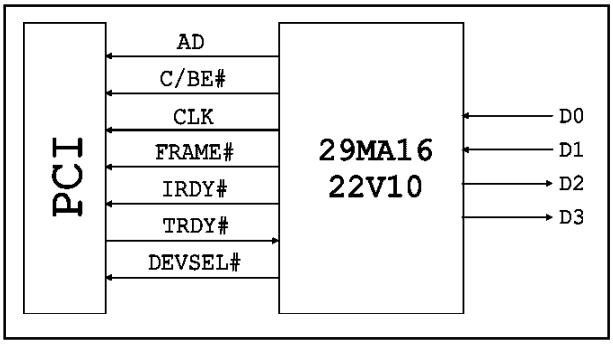

Рис. 4. Структурная схема самого простого интерфейса для PCI

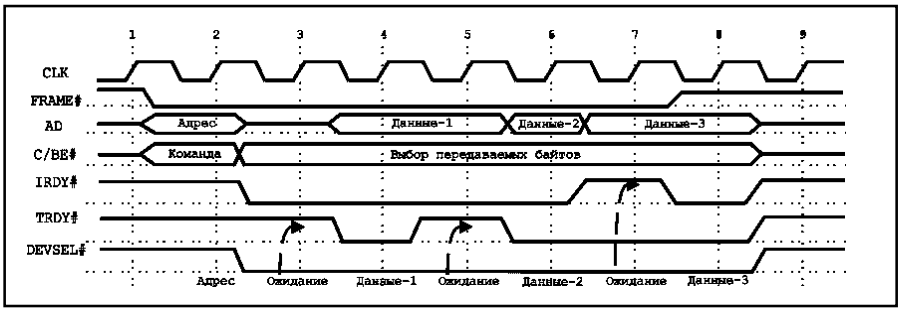

Рис. 5. Обмен данными по шине PCI

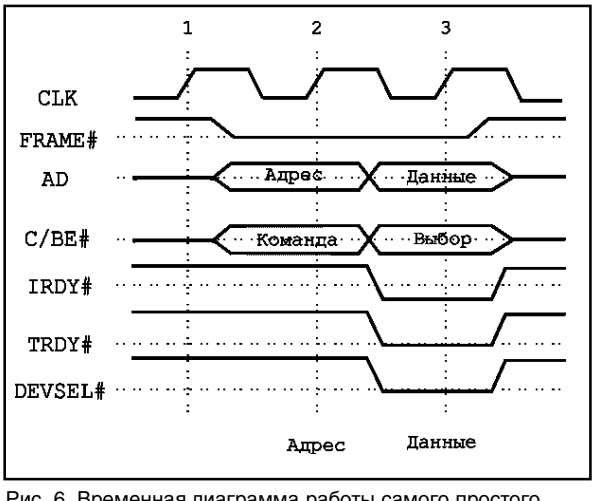

Рис. 6. Временная диаграмма работы самого простого интерфейса для PCI

После запуска программы лучше синхронизировать осциллограф не по сигналу тактовой частоты, а по сигналу FRAME#, с которого начинается цикл записи. В случае, если плата не будет работать, система после нескольких тактов ожидания сама выставит сигнал DEVSEL#, завершит запись и перейдет к новому циклу. Если плата будет нормально работать, вы увидите временные диаграммы, показанные выше.

Те, кого эта программа не устраивает (например, хочется работать с большими объемами памяти в защищенном режиме), могут сделать примерно то же самое, написав программу на C для компилятора WC. Можно использовать любую версию компилятора, поддерживающую 32-битный режим с помощью драйвера pharlap.

Существуют более правильные способы работы с шиной. Все фирмы, производящие ПЛМ, разработали наборы для построения интерфейса PCI. Документацию и исходные тексты найти довольно легко. В одном из номеров журнала EDN за прошлый год есть очень хороший обзор всех инструментов такого типа, необходимых для построения интерфейсов PCI. Среди самых популярных:

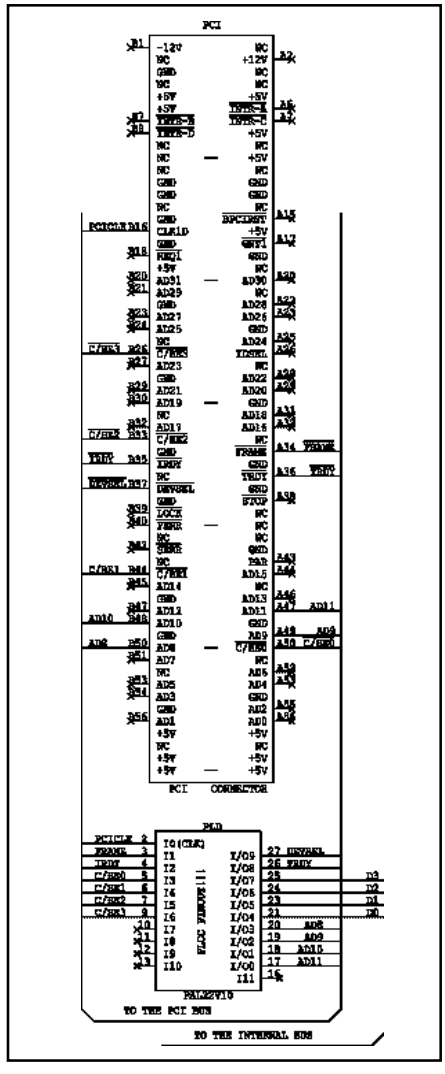

Рис. 7. Принципиальная схема самого простого интерфейса для PCI

AMD — www.amd.com,
Altera — www.altera.com
Xilinx — www.xilinx.com
QuickLogic — www.quicklogic.com

Много статей с советами на эту тему можно найти на сайте фирмы Daia I/O. Там же можно бесплатно выписать два десятка журналов за прошедшие месяцы со статьями на тему ПЛМ и PCI. Журналы придут довольно быстро.

Фирма AMCC и некоторые другие выпускают готовые контроллеры PCI. Эти контроллеры соответствуют всем требованиям стандарта PCI и работают очень хорошо. Однако у них есть принципиальный недостаток. Контроллер имеет два интерфейса: интерфейс для PCI и пользовательский интерфейс, к которому должна подключаться схема периферийного устройства. Эта схема, в свою очередь, тоже имеет некоторый интерфейс, что заставляет разработчика делать дополнительный интерфейс между своей схемой и контроллером.

Хорошо выполненное описание контроллеров PCI было сделано фирмой Intel. Его можно поискать в архивах на сайте или запросить в представительстве. И самый первый шаг, который нужно сделать при серьезном подходе, — купить в PCI Special Interest Group спецификацию стандарта PCI Local Bus.

Вадим Стрижов,
Москва