TEXAS INSTRUMENTS

PC97 System Design with the PCI1131

Revision 1.01
Feb. 14, 1997

PC97 and CB Controllers

The Microsoft PC97 initiative covers a wide variety of both hardware and software found in the PC. When designing and implementing a PC97-compliant system, care must be taken to identify the portions of PC97 which apply. For the PCMCIA/CardBus subsystem, the PCI and PC Card chapters of the PC97 specification are relevant. Texas Instruments has worked diligently in the design of its CardBus controllers to ensure that they comply with the PC97 specification and indeed all-applicable industry specs. This document will detail the device and system-level issues in designing a PC97-compliant PCMCIA subsystem.

The following two sections will identify the portions of the PC97 spec which are related to CardBus controller design, specifically the PCI and PC Card chapters, and how the PCI1131 CardBus controller fulfills these requirements. The section after that will detail system implementation details for using the PCI1131.

PCI

PCI 2.1: Reviewing the PCI chapter of the PC97 Hardware Design Guide, the first requirement is a very broad one: compliance to the PCI Local Bus Specification, revision 2.1. Most PCI devices on the market today are PCI 2.0 compliant, but in the past year several devices have been introduced to the market which comply with the latest PCI spec, and the PCI1131 is among them.

The PCI 2.1 specification differs from its predecessor in some subtle but important ways. First, the use of delayed transactions is mandated in the new spec. Both delayed read and delayed write bus transactions are intended to improve bus efficiency by preventing slow I/O peripherals or bridges from inserting large numbers of wait states during accesses. The delayed transaction allows a PCI target device to 'retry' a read or write cycle, which frees the bus for other activity while the target device initiates the appropriate cycle on its secondary bus. The original PCI master device will later attempt the cycle again, until the target device has the desired data and the transaction is complete. This is an important consideration for the PCI1131, which is responsible for bridging transactions between the PCI bus and 16-bit PC Cards. These 16-bit PC Cards are slow, asynchronous peripherals with access times similar to an ISA bus. A PCI 2.0-compliant PCMCIA controller would insert long wait states on the PCI bus while accessing 16-bit LAN or modem PC Cards, which could fatally starve other PCI peripherals like the graphics or IDE controllers.

Another PCI 2.1 addition was to mandate the correct use of the Memory Read (MR), Memory Read Line (MRL), and Memory Read Multiple (MRM) bus commands. The intent of these commands is for PCI master devices to choose the correct command based on how much data the requesting device wishes to read; the MR command for reads less than a cache line, MRL for a cache line, and MRM for multiple cache line reads. The PCI target device can then use the type of command to pre-fetch the correct amount of data. This feature is also important in the PCI1131, when bridging between CardBus and PCI devices. When a CardBus card uses the MR command, the PCI1131 will fetch a single double-word of data from the requested PCI address; when a card uses the MRL or MRM commands, the PCI1131 will pre-fetch up to eight double-words of data, starting from the requested PCI address. For transferring a cache line, the PCI1131's correct use of the PCI MRL read command results in a CardBus card receiving the desired data 5 to 6 times faster than a PCMCIA controller which pre-fetches only a single doubleword of data.

Closing Bridge BAR's: Another PC97 requirement for PCI peripherals is for bridges that implement base/limit style bridge windows. The PC97 spec requires a specific method for closing bridge BAR's, that matches the convention used by the operating system. The

PCI1131 implements this type of base/limit registers for its CardBus memory and I/O windows, and complies with this closing method described in PC97.

Subsystem ID/Subsystem Vendor ID: The PCI 2.1 specification added several new registers to the defined configuration space, called Subsystem ID (SSID) and Subsystem Vendor ID (SSVID) registers. The SSVID register should be populated with the PCI Vendor ID of the subsystem OEM, in this case, the PC manufacturer. The SSID register should be populated with a distinct ID for the subsystem itself. The SSVID & SSID's are similar to the PCI Vendor and Device ID's, but refer to the subsystem rather than just the PCI silicon. The PC97 specification requires that PCI peripherals implement these registers. This requirement will be enforced in the Microsoft logo tests after 7/1/96. The PCI1131 implements these registers as read/write registers, such that the PC's initialization BIOS can write the appropriate values into these registers upon boot-up. Further clarification of the SSID/SSVID register implementation was made available after the PCI1131 was released to market, which describes these registers as read-only, however the PCI1131 will satisfy PC97 requirements by following the directions found in a later section of this document, called 'PCI1131 Implementation Guidelines'.

That concludes the PC97 PCI requirements for the PC Card controller. The next section will discuss the PC Card-related issues for PC Card controller design.

PC Card

PC Card Standard: The PC97 chapter entitled 'PC Card' contains requirements for both PC Card controllers and PC Cards. The items specific to the PC Card controller are primarily related to register models and compliance to standards. The obvious standard mentioned here is the PC Card Standard, maintained by the PCMCIA. PC97 mandates that the PC Card controllers comply with the February 1995 release of this standard. The PCI1131 does so, of course, because this was the first PCMCIA standard to describe CardBus. Texas Instruments is an active member of PCMCIA, and holds a seat on its board of directors to ensure that the latest changes to the PC Card Standard are incorporated into TI's next generation PC Card controllers.

PCIC & Yenta: Another PC97 requirement for the PC Card controller is to support the legacy PCIC or 82365 register set, and to follow the Yenta Register specification. While the 82365 register set was the de-facto standard for the 16-bit PC Card controllers, the Yenta specification defines the register set for the latest generation of CardBus controllers. Support for the 82365 register set in the PCI1131 provides a legacy mode of operation. A number of mature Socket Services are available in older operating systems that can configure and operate the 16-bit PC Cards with the PCI1131. Of course, to take advantage of CardBus functionality, a new set of Socket Services must be used which recognizes the Yenta register set. Yenta defines a new PCI header type, Type 2, just for CardBus controllers. The Type 2 header definition is a cross between the standard Type 0 PCI header and the PCI-PCI Bridge Type 1 header. A Yenta CardBus controller is a multi-function device; there are separate PCI functions, both Type 2, for control of each socket individually. The PCI1131 Yenta register set is shown below in Figure 1. The Yenta portion is from offset 00h to 7Fh

Register Name				Offset
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Revision ID	08h
BIST	Header Type	Latency Timer	Cache Line Size	0Ch
CardBus Socket Registers / ExCA Base Address Register				10h
Secondary Status		Reserved		14h
CardBus Latency Timer	Subordinate Bus Number	CardBus Bus Number	PCI Bus Number	18h
Memory Base Register 0				1Ch
Memory Limit Register 0				20h
Memory Base Register 1				24h
Memory Limit Register 1				28h
I/O Base Register 0				2Ch
I/O Limit Register 0				30h
I/O Base Register 1				34h
I/O Limit Register 1				38h
Bridge Control		Interrupt Pin	Interrupt Line	3Ch
Subsystem ID		Subsystem Vendor ID		40h
PC Card 16-Bit I/F Legacy Mode Base Address				44h
Reserved				48h - 7Fh
System Control Register				80h
Reserved	Reserved	Reserved	Reserved	84h - 8Ch
Buffer	Device Control	Card Control	Retry Status	90h
Socket DMA Register 0				94h
Socket DMA Register 1				98h
Reserved				9Ch - FFh

Figure 1. PCI1131 Register set

PCI1131 Implementation Guidelines

There are several implementation details called out in the PC97 specification, which are appropriate to designing with the PCI1131. These are board layout or BIOS considerations that ensure proper operation with Microsoft operating systems and PC97 logo compatibility.

Correct IRQ Routing: The PC Card controller typically allows several methods to communicate edge-mode IRQ's from the controller to the PIC (Programmable Interrupt Controller), one of which is using discrete pins for each IRQ on both devices. The requirement for maintaining correct IRQ routing means that IRQ9 on the PC Card controller should be connected to only IRQ9 on the PIC. 'Crossing' IRQ's, where IRQ9 from the controller is connected to, say, IRQ10 on the PIC, may be done in a system when the corresponding Card & Socket Services is fully aware of this implementation. However, this is a difficult guarantee over multiple operating systems and after full installations of a new OS, so this design practice is discouraged, and PC97 forbids it.

PCI and IRQ Interrupts: While the use of interrupts and the signaling style is unique to the type of computing platform in use, the x86 PC architecture can handle both PCI and IRQ interrupts. Devices may use the type of interrupt most appropriate to their function. The CardBus PC Card controller, however, requires both to be connected to the PIC simultaneously. The choice of which types and numbers of interrupts to be used is decided by Card & Socket Services and the operating system at run-time. The 16-bit PC Cards typically use IRQ interrupts, but may use level-mode PCI interrupts. CardBus cards, on the other hand, use PCI interrupts exclusively. Therefore, it is very important that both styles of interrupts are connected in PCI1131 implementations. Note that for this requirement and for the preceding one, the use of Serialized IRQ's on the PCI1131 results in the most efficient board layout (using only three pins for interrupts), and the fewest resource constraints.

As with any PCI device in the PC, some amount of BIOS initialization of the hardware is required for the OS to boot properly and enumerate the system. The CardBus controller is no exception, and Microsoft has outlined in detail the BIOS setup requirements of the CardBus controller in a white paper entitled 'CardBus Host Controllers and Windows Compatibility'. This white paper can be obtained by searching the Microsoft web site for 'http://www.microsoft.com/hwdev/busbios/cardbus1.htm'. Details such as the choosing the correct Plug-and-Play ID (PNP0E03), creating a dev node in the BIOS for the controller and the need for the PCI IRQ routing table are covered in this Microsoft white paper.

As mentioned previously, the PCI1131 requires a specific convention for programming of its Subsystem ID & Subsystem Vendor ID registers to ensure PC97 compliance. Very simply, at system boot-time the notebook motherboard BIOS should write valid SSID/SSVID values to any PCI1131's on the notebook motherboard; if a docking station is detected at boot-time with a PCI1131 on-board, the SSID/SSVID registers on this PCI1131 should *not* be written. If during later operation, a docking station with a PCI1131 is hot-docked with the notebook, the notebook BIOS again should *not* attempt to populate the SSID/SSVID fields on the dock. When this convention is correctly implemented, the operating system will see a clear and consistent distinction between PCI1131's on the notebook motherboard and those on a dock.

Future TI CardBus Controller Products

In conclusion, this application note has described the design features in the PCI1131 CardBus controller from TI that guarantee its compliance with the PC97 specification. Additionally, implementation guidelines were discussed which ensure that the resulting notebook PC will meet Microsoft's PC97 system guidelines, and will receive the 'Designed for Microsoft Windows95' logo throughout 1997.

Industry standards are changing very rapidly, however, and TI's CardBus controller team is busy providing solutions for the next changes in the PC environment. On 1/1/98, notebook PC's that would receive the Microsoft logos must support ACPI and PCI Power Management. So, TI has introduced its PCI1250 and PCI1220, the first in its high-performance Multimedia series and fully compliant with ACPI and PCI Power Management. These products will sample to OEM's and be released-to-production in time for the fall and winter '97/'98 notebook product lines.

The method of programming the SSID/SSVID on TI CardBus controllers will be slightly different in the PCI1250 and PCI1220 in two respects. First, the SSID/SSVID registers will be read-only by default. In order to populate these fields, the PC BIOS will first unlock the registers using a TI-defined lock/unlock bit, write the proper Subsystem information, then re-lock the registers. This will ensure that the operating system will always find these registers as read-only. The second change in the PCI1250 and PCI1220 is that an EEPROM interface has been added to these devices, so that an EEPROM may be connected and read by the CardBus controller when it is taken out of power-on reset. The EEPROM contains the valid Subsystem information for the platform and is loaded automatically into the SSID/SSVID registers. The use of this EEPROM feature is optional, and intended for use in CardBus implementations in hot-dock-capable docking stations. Look for details on this EEPROM implementation and SSID/SSVID lock/unlock bits in the datasheets for these products.