Chapter 5

Bus Slots and I/O Cards

At the heart of every system is the motherboard; you learned about various motherboards in the preceding chapter. A motherboard is made up of components. The major component that determines how the motherboard actually works is called the *bus*. In this chapter, you learn about system buses. Specifically, you learn the following:

- What a bus is and what types of buses exist
- Why expansion slots are needed
- What types of I/O buses are used in PC systems
- What system resources are
- How adapter cards use system resources
- How to resolve conflicts among system resources

What Is a Bus?

A bus is nothing but a common pathway across which data can travel within a computer. This pathway is used for communication and can be established between two or more computer elements. A PC has many kinds of buses, including the following:

- Processor bus
- Address bus
- Memory bus
- I/O bus

If you hear someone talking about the PC bus, chances are good that he or she is referring to the I/O bus, which also is called the *expansion bus*. Whatever name it goes by, this bus is the main system bus and the one over which most data flows. The I/O bus is the highway for most data in your system. Anything that goes to or from any device—including your video system, disk drives, and printer—travels over this bus. The busiest I/O pathway typically is to and from your video card.

Because the I/O bus is the primary bus in your computer system, it is the main focus of discussion in this chapter. The other buses deserve some attention, however, and they are covered in the following sections.

The Processor Bus

The *processor bus* is the communication pathway between the CPU and immediate support chips. This bus is used to transfer data between the CPU and the main system bus, for example, or between the CPU and an external memory cache. Figure 5.1 shows how this bus fits into a typical PC system.

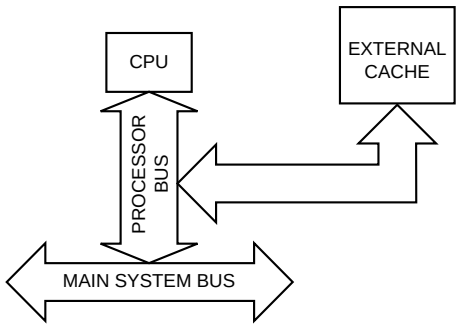

Fig. 5.1
The processor bus.

Not all PC systems have an external cache for the CPU; these caches typically are used only in high-end systems that use the faster 486 and Pentium chips. In many systems, therefore, the sole purpose of the processor bus is to communicate with the main system bus.

Because the purpose of the processor bus is to get information to and from the CPU at the fastest possible speed, this bus operates at a much faster rate than any other bus in your system; no bottleneck exists here. The bus consists of electrical circuits for data, for addresses (the address bus, which is discussed in the following section), and for control purposes. In a 486-based system, for example, the processor bus consists of 32 address lines, 32 data lines, and a few lines for control purposes. A Pentium system's processor bus has 64 data lines, 32 address lines, and associated control lines.

The processor bus operates at the same base clock rate as the CPU and can transfer one bit of data per data line every one or two clock cycles. Thus, a 486-based system can transfer 32 bits of data at a time, whereas a Pentium can transfer 64 bits of data.

To determine the transfer rate for the processor bus, you multiply the data width (32 bits for a 486 or 64 bits for a Pentium) by the clock speed of the bus (the same as the base clock speed of the CPU). If you are using a 66 MHz Pentium chip that can transfer a bit

of data each clock cycle on each data line, you have a maximum instantaneous transfer rate of 528M per second. You get this result by using the following formula:

66 MHz × 64 bits = 4,224 megabits/second

4,224 megabits/second ÷ 8 = 528M/second

This transfer rate, often called the *bandwidth* of the bus, represents a maximum. Like all maximums, this rate does not represent the normal operating bandwidth; you should expect lower average throughput, even up to 25 percent lower. A system that has a 128M/second bandwidth (such as a 33 MHz 486 system), for example, typically may operate at a rate of 100 M/second. Other limiting factors (such as how fast the system bus can feed information to the processor bus) can conspire to lower the effective bandwidth even more.

The Memory Bus

The *memory bus* is used to transfer information between the CPU and main memory—the RAM in your system. This bus is implemented by a dedicated chipset that is responsible for transferring information between the processor bus and the memory bus. This chipset typically is the same chipset that is responsible for managing the I/O bus. Figure 5.2 shows how the memory bus fits into your PC.

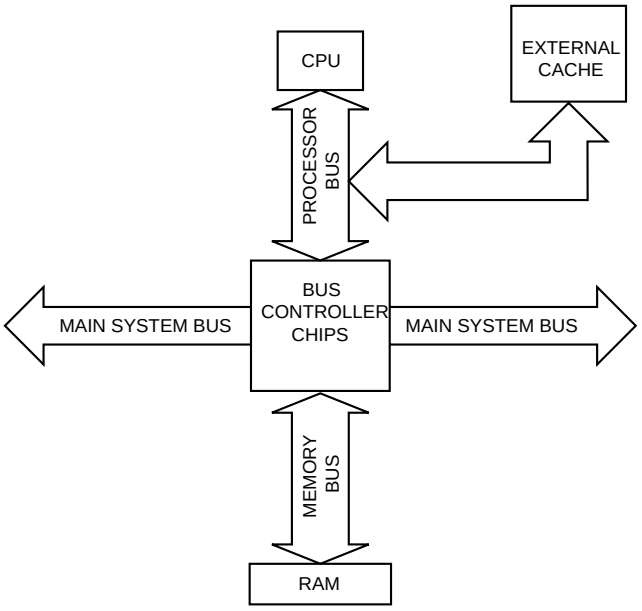

Fig. 5.2
The memory bus.

The information that travels over the memory bus is transferred at a much slower rate than the information on the processor bus, for two reasons: the memory bus has fewer data lines, and the memory chips cannot handle data as quickly as the CPU can. The chip sockets or the slots for memory SIMMs are connected to the memory bus in much the same way that expansion slots are connected to the I/O bus.

Caution

You should make sure that you purchase a system in which the data width of the memory bus matches the capabilities of the CPU. If a system has a 32-bit CPU, for example, you should insist that it also have a 32-bit memory bus. Likewise, in a Pentium system (which has a 64-bit CPU), you should insist on a 64-bit memory bus. Some Pentium systems, particularly the early ones, have a limited memory bus (32 bits); stay away from these systems. You typically can identify these systems by their names; they are touted as being "Pentium-ready" or "P5-ready."

The Address Bus

The *address bus* actually is a subset of the processor bus. Earlier in this chapter, you learned that the processor bus in a 486 or Pentium system consists of either 32 or 64 data lines, 32 address lines, and a few control lines. These address lines constitute the address bus; in most block diagrams, this bus is not broken out from the processor bus.

The address bus is used to perform operations on memory. The address bus indicates precisely where in memory the next operation will occur. The size of the memory bus is directly related to the amount of memory that the CPU can address directly.

The Need for Expansion Slots

The I/O bus enables your CPU to communicate with peripheral devices. The bus and its associated expansion slots are needed because basic systems cannot possibly satisfy all the needs of all the people who buy them. The I/O bus enables you to add devices to your computer to expand its capabilities. The most basic computer components, such as hard drive controllers and video adapter cards, can be plugged into expansion slots; you also can plug in more specialized devices, such as sound boards and CD-ROM drive interface cards.

Note

In some PC systems, some I/O capability is built into the motherboard. Some systems, for example, have the hard drive controller, SCSI port, serial ports, mouse port, and parallel printer ports built into the motherboard; in such a system, an expansion slot on the I/O bus is not needed. Nevertheless, these built-in controllers and ports still use the I/O bus to communicate with the CPU.

The number of expansion slots varies among computers. The original IBM PC had five expansion slots, for example, and the PC-XT had eight slots; to this day, virtually no PC systems have more than eight expansion slots. When the number of expansion slots increases, the distance between the slots decreases. Slots are approximately an inch apart in the original PC and only 0.8 inch apart in XT-series machines. Because of this design change, some of the extremely thick (*double-stacked*) expansion cards that fit well in a PC require two adjacent slots in most other systems.

Some PC systems provide only a single expansion slot on the motherboard. This slot typically is called a *riser-card slot*. The riser card in turn has expansion slots on its sides. Adapter cards are installed in the riser card, meaning that they are parallel to the motherboard.

Riser cards are used when a vendor wants to produce a computer that is shorter in height than normal. These computers usually are called "slimline," "low profile," or "ultra-thin." Even though this type of configuration may seem odd, the actual bus used in these systems is the same kind used in normal computer systems; the only difference is the use of the riser card.

Types of I/O Buses

Since the introduction of the first PC, many I/O buses have been introduced; most buses seem to have been introduced in the past several years. The reason is quite simple: faster I/O speeds are necessary for better system performance. This need for higher performance involves three main areas:

- Faster CPUs
- Increasing software demands
- Greater video requirements

Each of these areas requires the I/O bus to be as fast as possible. Surprisingly, a large number of PC systems shipped today (by some reports, as many as 75 percent) still use the same bus architecture as the IBM PC/AT. This situation is changing, however, as the types of systems sold demand different I/O bus structures, as the architectures mature, and as the cost of newer systems continues to plummet.

One of the primary reasons why new I/O-bus structures have been slow in coming is compatibility—that old catch-22 that anchors much of the PC industry to the past. One of the hallmarks of the PC's success is its standardization. This standardization spawned thousands of third-party I/O cards, each originally built for the early bus specifications of the PC. If a new, high-performance bus system is introduced, it must be compatible with the older bus systems so that the older I/O cards do not become obsolete. Therefore, bus technologies seem to evolve rather than make quantum leaps forward.

You can identify different types of I/O buses by their architecture. The main types of I/O architecture are:

- ISA
- Micro channel
- EISA
- Local bus
- VESA local bus
- PCMCIA bus

The differences among these buses consist primarily of the amount of data that they can transfer at one time and the speed at which they can do it. Each bus architecture is implemented by a chipset that is connected to the processor bus. Typically, this chipset also controls the memory bus (refer to fig. 5.2 earlier in this chapter). The following sections describe the different types of PC buses.

The ISA Bus

ISA, which is an acronym for *Industry Standard Architecture,* is the bus architecture that was introduced with the original IBM PC in 1982 and later expanded with the IBM PC/AT. ISA is the basis of the modern personal computer and the primary architecture used in the vast majority of PC systems on the market today. It may seem amazing that such an antiquated architecture is used in today's high-performance systems, but this seems to be true for reasons of reliability, affordability, and compatibility.

The ISA bus enabled thousands of manufacturers to build systems whose components (except for a few specialized parts) were interchangeable. Floppy drives that work in an IBM PC also work in IBM clones, for example, and video adapters that work in IBM ATs also work in IBM-compatible systems based on the 286 CPU chip.

Two versions of the ISA bus exist, based on the number of data bits that can be transferred on the bus at a time. The older version is an 8-bit bus; the newer version is a 16-bit bus. Both versions of the bus operate at an 8 MHz cycle rate, with data transfers requiring anywhere from two to eight cycles. Therefore, the theoretical maximum data rate of the ISA bus is 8M per second, as the following formula shows:

8 MHz × 16 bits = 128 megabits/second

128 megabits/second ÷ 2 cycles = 64 megabits/second

64 megabits/second ÷ 8 = 8M/second

The bandwidth of the 8-bit bus would be half this figure (4M per second). Remember, however, that these figures are theoretical maximums; because of I/O bus protocols, the effective bandwidth is much lower—typically by half.

The 8-Bit ISA Bus. This bus architecture is used in the original IBM PC computers. Although virtually nonexistent in new systems today, this architecture still exists in hundreds of thousands of PC systems in the field.

Physically, the 8-bit ISA expansion slot resembles the tongue-and-groove system that furniture makers once used to hold two pieces of wood together. An adapter card with 62 gold contacts on its bottom edge plugs into a slot on the motherboard that has 62 matching gold contacts. Electronically, this slot provides 8 data lines and 20 addressing lines, enabling the slot to handle 1M of memory.

Table 5.1 describes the pinouts for the 8-bit ISA bus.

Table 5.1 Pinouts for the 8-Bit ISA Bus

Pin	Signal Name	Pin	Signal Name
B1	Ground	A1	I/O Channel Check
B2	Reset Driver	A2	Data 7
B3	+5V	A3	Data 6
B4	IRQ 2	A4	Data 5
B5	–5V	A5	Data 4
B6	DMA Request 2	A6	Data 3
B7	–12V	A7	Data 2
B8	Card Selected (XT only)	A8	Data 1
B9	+12V	A9	Data 0
B10	Ground	A10	I/O Channel Ready
B11	Memory Write	A11	Address Enable
B12	Memory Read	A12	Address 19
B13	I/O Write	A13	Address 18
B14	I/O Read	A14	Address 17
B15	DMA Acknowledge 3	A15	Address 16
B16	DMA Request 3	A16	Address 15
B17	DMA Acknowledge 1	A17	Address 14
B18	DMA Request 1	A18	Address 13
B19	DMA Acknowledge 0	A19	Address 12
B20	Clock	A20	Address 11
B21	IRQ 7	A21	Address 10
B22	IRQ 6	A22	Address 9
B23	IRQ 5	A23	Address 8
B24	IRQ 4	A24	Address 7
B25	IRQ 3	A25	Address 6
B26	DMA Acknowledge 2	A26	Address 5
B27	Terminal Count	A27	Address 4

(continues)

Table 5.1 Continued

Pin	Signal Name	Pin	Signal Name
B28	Address Latch Enable	A28	Address 3
B29	+5 VDC	A29	Address 2
B30	Oscillator	A30	Address 1
B31	Ground	A31	Address 0

Figure 5.3 shows how these pins are oriented in the expansion slot.

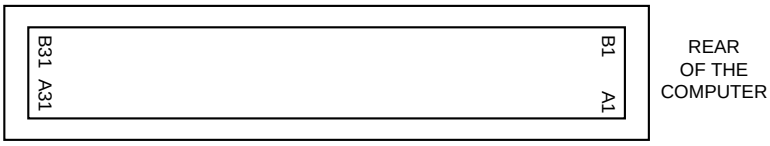

Fig. 5.3
The 8-bit ISA bus connector.

Although the design of the bus is simple, IBM never published full specifications for the timings of the data and address lines, so in the early days of IBM compatibles, manufacturers had to do their best to figure out how to make adapter boards. This problem was solved, however, as IBM-compatible personal computers became more widely accepted as the industry standard and manufacturers had more time and incentive to build adapter boards that worked correctly with the bus.

The dimensions of 8-bit ISA adapter cards are as follows:

4.2 inches (106.68mm) high

13.13 inches (333.5mm) long

0.5 inch (12.7mm) wide

In the XT or Portable PC, the eighth slot—the one closest to the power supply—is a special slot; only certain cards can be installed there. A card installed in the eighth slot must supply to the motherboard, on pin B8, a special card-selected signal, which few cards are designed to do. (The IBM asynchronous adapter card and the keyboard/timer card from a 3270 PC are two examples of cards that fit into the eighth slot.) Additionally, the timing requirements for the eighth slot are stricter.

The reason why this strange slot exists in the XT is that IBM developed the system to support a special configuration called the 3270-PC, which really is an XT with three to six special adapter boards installed. The eighth slot was designed specifically to accept the keyboard/timer adapter from the 3270 PC. This board needed special access to the motherboard because it replaced the motherboard keyboard circuitry. Special timing and the card-selected signal made this access possible.

Contrary to what many users believe, the eighth slot has nothing to do with the IBM expansion chassis, which was popular at the time of the PC-XT. The IBM expansion chassis is a box developed by IBM that looks like another system unit. Because the IBM XT has eight slots, one full-height floppy drive, and one full-height hard drive, the expansion chassis makes room for more expansion slots and for additional floppy and hard drives.

The 16-Bit ISA Bus. The second-generation 80286 chip can handle 16 bits on the I/O bus at a time, compared with 8 bits in older CPUs. The introduction of this chip posed a problem for IBM in relation to its next generation of PCs. Should the company create a new I/O bus and associated expansion slots, or should it try to come up with a system that could support both 8- and 16-bit cards? IBM opted for the latter solution, and the PC/AT was introduced with a set of double expansion slots. You can plug an older 8-bit card into the forward part of the slot or a newer 16-bit card into both parts of the slot.

Note

The expansion slots for the 16-bit ISA bus also introduced access keys to the PC environment. An *access key* is a cutout or notch in an adapter card that fits over a tab in the connector into which the adapter card is inserted. Access keys typically are used to keep adapter cards from being inserted into a connector improperly. The ends of the two expansion slots of the 16-bit ISA bus, when butted up against each other, serve the same purpose as an access key.

The second part of each expansion slot adds 36 connector pins to carry the extra signals necessary to implement the wider data path. In addition, one or two of the pins in the base portion of the connector (used for ISA cards) serve different purposes.

Table 5.2 describes the pinouts for the full 16-bit ISA expansion slot.

Table 5.2 Pinouts for the 16-Bit ISA Bus

Pin	Signal Name	Pin	Signal Name
B1	Ground	A1	I/O Channel Check
B2	Reset Driver	A2	Data 7
B3	+5V	A3	Data 6
B4	IRQ 9	A4	Data 5
B5	–5V	A5	Data 4
B6	DMA Request 2	A6	Data 3
B7	–12V	A7	Data 2
B8	–0 WAIT	A8	Data 1
B9	+12V	A9	Data 0
B10	Ground	A10	I/O Channel Ready

(continues)

Table 5.2 Continued

Pin	Signal Name	Pin	Signal Name
B11	Memory Write	A11	Address Enable
B12	Memory Read	A12	Address 19
B13	I/O Write	A13	Address 18
B14	I/O Read	A14	Address 17
B15	DMA Acknowledge 3	A15	Address 16
B16	DMA Request 3	A16	Address 15
B17	DMA Acknowledge 1	A17	Address 14
B18	DMA Request 1	A18	Address 13
B19	DMA Acknowledge 0	A19	Address 12
B20	Clock	A20	Address 11
B21	IRQ 7	A21	Address 10
B22	IRQ 6	A22	Address 9
B23	IRQ 5	A23	Address 8
B24	IRQ 4	A24	Address 7
B25	IRQ 3	A25	Address 6
B26	DMA Acknowledge 2	A26	Address 5
B27	Terminal Count	A27	Address 4
B28	Address Latch Enable	A28	Address 3
B29	+5 VDC	A29	Address 2
B30	Oscillator	A30	Address 1
B31	Ground Access key	A31	Address 0 Access key
D1	Memory 16-bit chip select	C1	System bus high enable
D2	I/O 16-bit chip select	C2	Latch Address 23
D3	IRQ 10	C3	Latch Address 22
D4	IRQ 11	C4	Latch Address 21
D5	IRQ 12	C5	Latch Address 20
D6	IRQ 15	C6	Latch Address 19
D7	IRQ 14	C7	Latch Address 18
D8	DMA Acknowledge 0	C8	Latch Address 17
D9	DMA Request 0	C9	Memory Read
D10	DMA Acknowledge 5	C10	Memory Write
D11	DMA Request 5	C11	Data 8
D12	DMA Acknowledge 6	C12	Data 9
D13	DMA Request 6	C13	Data 10
D14	DMA Acknowledge 7	C14	Data 11
D15	DMA Request 7	C15	Data 12
D16	+5V	C16	Data 13
D17	Master 16-bit select	C17	Data 14
D18	Ground	C18	Data 15

Figure 5.4 shows how the pins are oriented in the expansion slot.

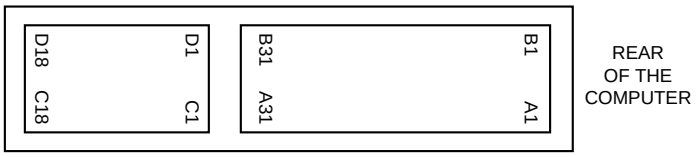

Fig. 5.4
The 16-bit ISA bus connector.

The extended 16-bit slots physically interfere with some adapter cards that have a *skirt*—an extended area of the card that drops down toward the motherboard just after the connector. To handle these cards, IBM left two expansion ports in the PC/AT without the 16-bit extensions. These slots, which are identical to the expansion slots in earlier systems, can handle any skirted PC or XT expansion card.

Note

Extended expansion slots were introduced years ago. Since then, virtually every manufacturer of expansion cards has modified its design so that the cards fit properly in the systems that are available today. Although card skirts are a thing of the past, you sometimes find them on older cards, such as those that you may pick up at a flea market or garage sale.

The dimensions of a typical AT expansion board are as follows:

4.8 inches (121.92mm) high

13.13 inches (333.5mm) long

0.5 inch (12.7mm) wide

Two heights actually are available for cards that are commonly used in AT systems: 4.8 inches and 4.2 inches (the height of older PC-XT cards). The shorter cards became an issue when IBM introduced the XT Model 286. Because this model has an AT motherboard in an XT case, it needs AT-type boards with the 4.2-inch maximum height. Most board makers trimmed the height of their boards; many manufacturers now make only 4.2-inch-tall boards so that they will work in systems with either profile.

32-Bit ISA Buses. After 32-bit CPUs became available, it was some time before 32-bit bus standards became available. Before MCA and EISA specs were released, some vendors began creating their own proprietary 32-bit buses, which were extensions of the ISA bus. Although the proprietary buses were few and far between, they do still exist.

The expanded portions of the bus typically are used for proprietary memory expansion or video cards. Because the systems are proprietary (meaning that they are nonstandard), pinouts and specifications are not available.

The Micro Channel Bus

The introduction of 32-bit chips meant that the ISA bus could not handle the power of another new generation of CPUs. The 386 chips can transfer 32 bits of data at a time, but the ISA bus can handle a maximum 16 bits. Rather than extend the ISA bus again, IBM decided to build a new bus; the result was the MCA bus. MCA (an acronym for *micro channel architecture*) is completely different from the ISA bus and is technically superior in every way.

IBM not only wanted to replace the old ISA standard, but also to receive royalties on it; the company required vendors that licensed the new proprietary MCA bus to pay IBM royalties for using the ISA bus in all previous systems. This requirement led to the development of the competing EISA bus (described later in this chapter) and hindered acceptance of the MCA bus. Another reason why MCA has not been adopted universally for systems with 32-bit slots is that adapter cards designed for ISA systems do not work in MCA systems.

Note

The MCA bus is not compatible with the older ISA bus, so cards designed for the ISA bus do not work in an MCA system.

MCA runs asynchronously with the main processor, meaning that fewer possibilities exist for timing problems among adapter cards plugged into the bus.

MCA systems produced a new level of ease of use, as anyone who has set up one of these systems can tell you. An MCA system has no jumpers and switches—neither on the motherboard nor on any expansion adapter. You don't need an electrical-engineering degree to plug a card into a PC.

The MCA bus also supports bus mastering. Through implementing bus mastering, the MCA bus provides significant performance improvements over the older ISA buses. (Bus mastering is also implemented in the EISA bus. General information related to bus mastering is discussed in the "Bus Mastering" section later in this chapter.) In the MCA bus mastering implementation, any bus mastering devices can request unobstructed use of the bus in order to communicate with another device on the bus. The request is made through a device known as the Central Arbitration Control Point (CACP). This device arbitrates the competition for the bus, making sure all devices have access and that no single device monopolizes the bus.

Each device is given a priority code to ensure that order is preserved within the system. The main CPU is given the lowest priority code. Memory refresh has the highest priority, followed by the DMA channels, and then the bus masters installed in the I/O slots. One exception to this is when an NMI (non-maskable interrupt) occurs. In this instance, control returns to the CPU immediately.

The MCA specification provides for four adapter sizes, which are described in table 5.3.

Table 5.3 Physical Sizes of MCA Adapter Cards

Adapter Type	Height	Length
Type 3	3.475"	12.3"
Type 3 half	3.475"	6.35"
Type 5	4.825"	13.1"
Type 9	9.0"	13.1"

Four types of slots are involved in the MCA design:

- 16-bit
- 16-bit with video extensions
- 16-bit with memory-matched extensions
- 32-bit

16-Bit MCA Slots. Every MCA slot has a 16-bit connector. This connector is the primary MCA slot design—the one used in all MCA systems. This 16-bit MCA slot has connectors that are smaller than the connectors in an ISA system. The slot itself is divided into two sections; one section handles 8-bit operations, and the other handles 16-bit operations.

Table 5.4 describes the pinouts for the 16-bit MCA connector. Pins B1/A1 through B45/A45 are responsible for 8-bit operations, and pins B48/A48 through B58/A58 handle 16-bit operations.

Table 5.4 Pinouts for the 16-Bit MCA Bus

Pin	Signal Name	Pin	Signal Name
B1	Audio Ground	A1	–CD Setup
B2	Audio	A2	MADE 24
B3	Ground	A3	Ground
B4	Oscillator	A4	Address 11
B5	Ground	A5	Address 10
B6	Address 23	A6	Address 9
B7	Address 22	A7	+5V
B8	Address 21	A8	Address 8
B9	Ground	A9	Address 7
B10	Address 20	A10	Address 6
B11	Address 19	A11	+5V
B12	Address 18	A12	Address 5
B13	Ground	A13	Address 4
B14	Address 17	A14	Address 3

(continues)

Table 5.4 Continued

Pin	Signal Name	Pin	Signal Name
B15	Address 16	A15	+5V
B16	Address 15	A16	Address 2
B17	Ground	A17	Address 1
B18	Address 14	A18	Address 0
B19	Address 13	A19	+12V
B20	Address 12	A20	–ADL
B21	Ground	A21	–PREEMPT
B22	–IRQ9	A22	–BURST
B23	–IRQ3	A23	–12V
B24	–IRQ4	A24	ARB 0
B25	Ground	A25	ARB 1
B26	–IRQ5	A26	ARB 2
B27	–IRQ6	A27	–12V
B28	–IRQ7	A28	ARB 3
B29	Ground	A29	ARB/–GNT
B30	–DPAREN	A30	–TC
B31	DPAR(0)	A31	+5V
B32	–CHCK	A32	–S0
B33	Ground	A33	–S1
B34	–CMD	A34	M/–IO
B35	CHRDYRTN	A35	+12V
B36	–CD SFDBK	A36	CD CHRDY
B37	Ground	A37	Data 0
B38	Data 1	A38	Data 2
B39	Data 3	A39	+5V
B40	Data 4	A40	Data 5
B41	Ground	A41	Data 6
B42	CHRESET	A42	Data 7
B43	–SD STROBE	A43	Ground
B44	–SDR(0)	A44	–DS 16 RTN
B45	Ground	A45	–REFRESH
B46	Access key	A46	Access key
B47	Access key	A47	Access key
B48	Data 8	A48	+5V
B49	Data 9	A49	Data 10
B50	Ground	A50	Data 11
B51	Data 12	A51	Data 13
B52	Data 14	A52	+12V
B53	Data 15	A53	DPAR(1)

Pin	Signal Name	Pin	Signal Name
B54	Ground	A54	–SBHE
B55	–IRQ10	A55	–CD DS 16#
B56	–IRQ11	A56	+5V
B57	–IRQ12	A57	–IRQ14
B58	Ground	A58	–IRQ15

Figure 5.5 shows how the connector for this card appears.

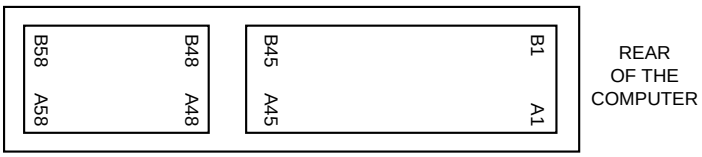

Fig. 5.5
A 16-bit MCA connector.

32-Bit MCA Slots. In addition to the basic 16-bit slot, MCA systems based on the 386DX or later CPU have several 32-bit slots that are designed to take advantage of the processors' increased communications and memory-addressing capabilities. Even though the 32-bit slot is an extension of the original MCA connector (as the 16-bit ISA slot is an extension of the 8-bit ISA design), this extension was designed at the same time as the rest of MCA. Therefore, because the extension connector was not an afterthought, the design is more integrated than the 16-bit extension in ISA systems.

Table 5.5 describes the pinouts for the 32-bit MCA connector. Pins B1/A1 through B58/A58 are exactly the same as in the 16-bit connector. Pins B59/A59 through B89/A89 are the 32-bit section.

Table 5.5 Pinouts for the 32-Bit MCA Bus

Pin	Signal Name	Pin	Signal Name
B1	Audio Ground	A1	–CD Setup
B2	Audio	A2	MADE 24
B3	Ground	A3	Ground
B4	Oscillator	A4	Address 11
B5	Ground	A5	Address 10
B6	Address 23	A6	Address 9
B7	Address 22	A7	+5V
B8	Address 21	A8	Address 8
B9	Ground	A9	Address 7
B10	Address 20	A10	Address 6

(continues)

Table 5.5 Continued

Pin	Signal Name	Pin	Signal Name
B11	Address 19	A11	+5V
B12	Address 18	A12	Address 5
B13	Ground	A13	Address 4
B14	Address 17	A14	Address 3
B15	Address 16	A15	+5V
B16	Address 15	A16	Address 2
B17	Ground	A17	Address 1
B18	Address 14	A18	Address 0
B19	Address 13	A19	+12V
B20	Address 12	A20	–ADL
B21	Ground	A21	–PREEMPT
B22	–IRQ9	A22	–BURST
B23	–IRQ3	A23	–12V
B24	–IRQ4	A24	ARB 0
B25	Ground	A25	ARB 1
B26	–IRQ5	A26	ARB 2
B27	–IRQ6	A27	–12V
B28	–IRQ7	A28	ARB 3
B29	Ground	A29	ARB/–GNT
B30	–DPAREN	A30	–TC
B31	DPAR(0)	A31	+5V
B32	–CHCK	A32	–S0
B33	Ground	A33	–S1
B34	–CMD	A34	M/–IO
B35	CHRDYRTN	A35	+12V
B36	–CD SFDBK	A36	CD CHRDY
B37	Ground	A37	Data 0
B38	Data 1	A38	Data 2
B39	Data 3	A39	+5V
B40	Data 4	A40	Data 5
B41	Ground	A41	Data 6
B42	CHRESET	A42	Data 7
B43	–SD STROBE	A43	Ground
B44	–SDR(0)	A44	–DS 16 RTN
B45	Ground	A45	–REFRESH
B46	Access key	A46	Access key
B47	Access key	A47	Access key
B48	Data 8	A48	+5V
B49	Data 9	A49	Data 10

Pin	Signal Name	Pin	Signal Name
B50	Ground	A50	Data 11
B51	Data 12	A51	Data 13
B52	Data 14	A52	+12V
B53	Data 15	A53	DPAR(1)
B54	Ground	A54	–SBHE
B55	–IRQ10	A55	–CD DS 16#
B56	–IRQ11	A56	+5V
B57	–IRQ12	A57	–IRQ14
B58	Ground	A58	–IRQ15
B59	Reserved	A59	Reserved
B60	Reserved	A60	Reserved
B61	–SDR(1)d	A61	Ground
B62	–MSDR	A62	Reserved
B63	Ground	A63	Reserved
B64	Data 16	A64	–SFDBKRTN
B65	Data 17	A65	+12V
B66	Data 18	A66	Data 19
B67	Ground	A67	Data 20
B68	Data 22	A68	Data 21
B69	Data 23	A69	+5V
B70	DPAR(2)	A72	Data 24
B71	Ground	A71	Data 25
B72	Data 27	A72	Data 26
B73	Data 28	A73	+5V
B74	Data 29	A74	Data 30
B75	Ground	A75	Data 31
B76	–BE0	A76	DPAR(3)
B77	–BE1	A77	+12V
B78	–BE2	A78	–BE3
B79	Ground	A79	–DS32 RTN
B80	TR32	A80	–CD DS32
B81	Address 24	A81	+5V
B82	Address 25	A82	Address 26
B83	Ground	A83	Address 27
B84	Address 29	A84	Address 28
B85	Address 30	A85	+5V
B86	Address 31	A86	–APAREN
B87	Ground	A87	–APAR(0)
B88	APAR(2)	A88	–APAR(1)
B89	APAR(3)	A89	Ground

Figure 5.6 shows how the connector for this card appears.

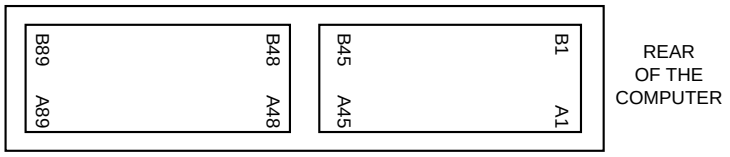

Fig. 5.6
The MCA connector.

Memory-Matched Extensions. Certain MCA systems (notably, IBM models 70 and 80) have bus extensions that support the operation of enhanced memory cards and support data transfer to those cards. These extensions are called *memory-matched extensions.*

The presence of memory-matched extensions varies from system to system. Some systems don't use the extensions at all; other systems include them on only one or a few slots. If one of your slots is equipped with these extensions, they will be on the end of the slot toward the rear of the motherboard (preceding pins B1 and A1 on your main MCA connector). You should refer to your system documentation to determine whether your system uses the extensions.

Table 5.6 details the additional pin-out specifications for the extensions.

Table 5.6 Additional Pinouts for the MCA Memory-Matched Extensions

Pin	Signal Name	Pin	Signal Name
BM4	Ground	AM4	Reserved
BM3	Reserved	AM3	–MM cycle command
BM2	–MM cycle request	AM2	Ground
BM1	Reserved	AM1	–MM cycle

Figure 5.7 shows how this connector appears in your system.

MCA Video Extensions. The third type of MCA slot is a standard 16-bit MCA connector with a special video-extension connector. This special slot appears in almost every MCA system. This slot is designed to speed your video subsystem.

The MCA video-extension connector is positioned at the end of the slot toward the rear of the motherboard (before pins B1 and A1 on the main MCA connector). The connector takes advantage of special VGA circuitry built into the motherboard. MCA provides compatible high-resolution video adapters that have special access to this motherboard circuitry, so that the special circuitry does not have to be duplicated on the video card itself.

No matter what new type of video board you add to an MCA system, all your programs will run, because you never lose the built-in VGA circuits. In addition, the built-in VGA circuits do not have to be disabled if you add a newer video card. Your new card can

coexist with the VGA circuits and even "borrow" some elements, such as the digital-to-analog converter. This arrangement (in theory) can make the add-on video boards less expensive, because they can rely on motherboard circuitry instead of providing their own.

Typically, only one slot in an MCA system has the video extensions. This arrangement makes sense, because a typical system has only one video card.

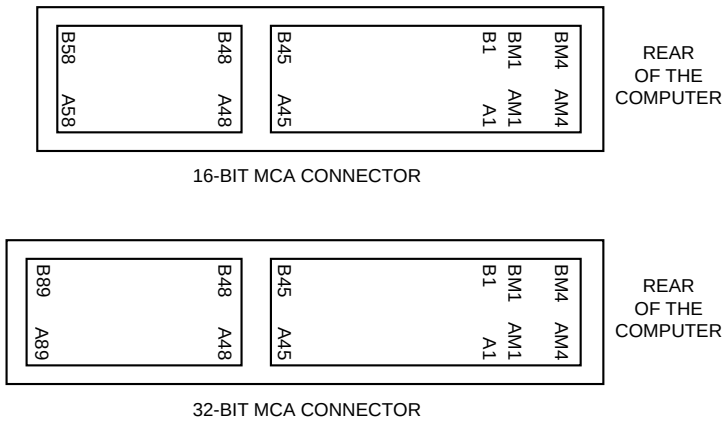

Fig. 5.7
The MCA connector with memory-matched extensions.

Table 5.7 describes the additional pinout connections for the slot with video extensions.

Table 5.7 Additional Pinouts for the MCA Video Extensions

Pin	Signal Name	Pin	Signal Name
BV10	Esync	AV10	Vsync
BV9	Ground	AV9	Hsync
BV8	P5	AV8	Blank
BV7	P4	AV7	Ground
BV6	P3	AV6	P6
BV5	Ground	AV5	Edclk
BV4	P2	AV4	Dclk
BV3	P1	AV3	Ground
BV2	P0	AV2	P7
BV1	Ground	AV1	Evideo

Figure 5.8 shows what this slot looks like in your system.

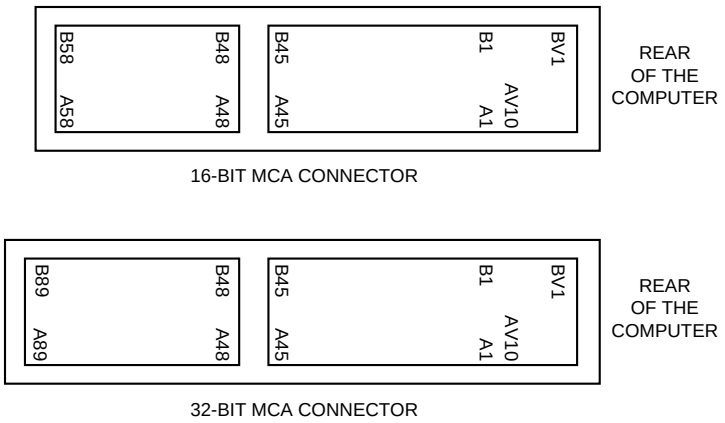

Fig. 5.8
The MCA connectors with video extensions.

The EISA Bus

EISA is an acronym for *extended industry standard architecture*. This standard was announced in September 1988 as a response to IBM's introduction of the MCA bus—more specifically, to the way that IBM wanted to handle licensing of the MCA bus. Vendors did not feel obligated to pay retroactive royalties on the ISA bus, so they turned their backs on IBM and created their own buses. The EISA standard was developed by leading computer manufacturers (minus IBM, of course) and supported by leading software companies. The first EISA machines started appearing on the market in 1989.

The EISA bus was designed as a successor to the ISA bus, although it has not turned out quite that way (as evidenced by the appearance of additional bus specifications). The EISA bus provides 32-bit slots for use with 386DX or higher systems. The EISA slot enables manufacturers to design adapter cards that have many of the capabilities of MCA adapters, but the bus also supports adapter cards created for the older ISA standard.

EISA provides markedly faster hard-drive throughput when used with devices such as SCSI bus-mastering hard drive controllers. Compared with 16-bit ISA system architecture, EISA permits greater system expansion with fewer adapter conflicts.

The EISA bus adds 90 new connections (55 new signals) without increasing the physical connector size of the 16-bit ISA bus. At first glance, the 32-bit EISA slot looks much like the 16-bit ISA slot. The EISA adapter, however, has two rows of connectors. The first row is the same kind used in 16-bit ISA cards; the other, thinner row extends from the 16-bit connectors.

To visualize the edge connectors on an EISA card, imagine that you are laying a 1-by-1-inch board on a 2-by-2-inch board in a lumberyard. The edge connector on an EISA board is about 0.2 inch longer than the connectors on an 8- or 16-bit ISA adapter board. The longest (and thinnest) connectors on an EISA card pass through the 16-bit part of the slot and make contact with the 32-bit connectors deeper in the slot.

The physical specifications of an EISA card are as follows:

5 inches (127mm) high

13.13 inches (333.5mm) long

0.5 inch (12.7mm) wide

The EISA specification calls for more than 45 watts at four different voltages to be available to each slot—a challenge to 200-watt or smaller power supplies, because it takes more than 325 watts to fully power the eight EISA slots in a system. Most EISA adapter cards, however, do not use the full 45 watts available to them; in fact, most cards use about the same amount of power as 8- and 16-bit ISA adapter boards.

The EISA bus can handle up to 32 bits of data at an 8.33 MHz cycle rate. Most data transfers require a minimum of two cycles, although faster cycle rates are possible if an adapter card provides tight timing specifications. The maximum bandwidth on the bus is 33M per second, as the following formula shows:

8.33 MHz × 32 bits = 266.56 megabits/second

266.56 megabits/second ÷ 8 = 33.32M/second

Data transfers through an 8- or 16-bit expansion card across the bus would be reduced appropriately. Remember, however, that these figures represent theoretical maximums. Wait states, interrupts, and other protocol factors can reduce the effective bandwidth—typically, by half.

Table 5.8 describes the pinouts for the EISA bus.

Table 5.8 Pinouts for the EISA Bus

Pin	Signal Name	Pin	Signal Name
B1	Ground	A1	I/O Channel Check
F1	Ground	E1	CMD
B2	Reset Driver	A2	Data 7
F2	+5V	E2	START
B3	+5V	A3	Data 6
F3	+5V	E3	EXRDY
B4	IRQ 9	A4	Data 5
F4	Unused	E4	EX32
B5	–5V	A5	Data 4
F5	Unused	E5	Ground
B6	DMA Request 2	A6	Data 3
F6	Access key	E6	Access key
B7	–12V	A7	Data 2

(continues)

Table 5.8 Continued

Pin	Signal Name	Pin	Signal Name
F7	Unused	E7	EX16
B8	Zero Wait State	A8	Data 1
F8	Unused	E8	SLBURST
B9	+12V	A9	Data 0
F9	+12V	E9	MSBURST
B10	Ground	A10	I/O Channel Ready
F10	Memory I/O	E10	W/R
B11	Memory Write	A11	Address Enable
F11	LOCK#	E11	Ground
B12	Memory Read	A12	Address 19
F12	Reserved	E12	Reserved
B13	I/O Write	A13	Address 18
F13	Ground	E13	Reserved
B14	I/O Read	A14	Address 17
F14	Reserved	E14	Reserved
B15	DMA Acknowledge 3	A15	Address 16
F15	BE3	E15	Ground
B16	DMA Request 3	A16	Address 15
F16	Access key	E16	Access key
B17	DMA Acknowledge 1	A17	Address 14
F17	BE2	E17	BE1
B18	DMA Request 1	A18	Address 13
F18	BE0	E18	LA31
B19	Refresh	A19	Address 12
F19	Ground	E19	Ground
B20	Clock	A20	Address 11
F20	+5V	E20	LA30
B21	IRQ 7	A21	Address 10
F21	LA29	E21	LA28
B22	IRQ 6	A22	Address 9
F22	Ground	E22	LA27
B23	IRQ 5	A23	Address 8
F23	LA26	E23	LA25
B24	IRQ 4	A24	Address 7
F24	LA24	E24	Ground
B25	IRQ 3	A25	Address 6
F25	Access key	E25	Access key
B26	DMA Acknowledge 2	A26	Address 5
F26	LA16	E26	LA15

Pin	Signal Name	Pin	Signal Name
B27	Terminal Count	A27	Address 4
F27	LA14	E27	LA13
B28	Address Latch Enable	A28	Address 3
F28	+5V	E28	LA12
B29	+5V	A29	Address 2
F29	+5V	E29	LA11
B30	Oscillator	A30	Address 1
F30	Ground	E30	Ground
B31	Ground	A31	Address 0
F31	LA10 Access key	E31	LA9 Access key
H1	LA8	G1	LA7
H2	LA6	G2	Ground
D1	Memory 16-bit chip select	C1	System Bus High Enable
H3	LA5	G3	LA4
D2	I/O 16-bit chip select	C2	LA23
H4	+5V	G4	LA3
D3	IRQ 10	C3	LA22
H5	LA2	G5	Ground
D4	IRQ 11	C4	LA21
H6	Access key	G6	Access key
D5	IRQ 12	C5	LA20
H7	Data 16	G7	Data 17
D6	IRQ 15	C6	LA19
H8	Data 18	G8	Data 19
D7	IRQ 14	C7	LA18
H9	Ground	G9	Data 20
D8	DMA Acknowledge 0	C8	LA17
H10	Data 21	G10	Data 22
D9	DMA Request 0	C9	Memory Read
H11	Data 23	G11	Ground
D10	DMA Acknowledge 5	C10	Memory Write
H12	Data 24	G12	Data 25
D11	DMA Request 5	C11	Data 8
H13	Ground	G13	Data 26
D12	DMA Acknowledge 6	C12	Data 9
H14	Data 27	G14	Data 28
D13	DMA Request 6	C13	Data 10
H15	Access key	G15	Access key

(continues)

Table 5.8 Continued

Pin	Signal Name	Pin	Signal Name
D14	DMA Acknowledge 7	C14	Data 11
H16	Data 29	G16	Ground
D15	DMA Request 7	C15	Data 12
H17	+5V	G17	Data 30
D16	+5V	C16	Data 13
H18	+5V	G18	Data 31
D17	Master 16-bit select	C17	Data 14
H19	/MACK0	G19	/MREQ0
D18	Ground	C18	Data 15

Figure 5.9, on the following page, shows the locations of the pins.

Bus Mastering. EISA uses a technology called *bus mastering* to speed the system. In essence, a bus master is an adapter with its own processor that can execute operations independently of the CPU. To work properly, bus-mastering technology relies on an *EISA arbitration unit*, most often called an *integrated system peripheral* (ISP) chip. The ISP enables a bus-mastered board to temporarily take exclusive control of the system, as though the board were the *entire* system. Because the board has exclusive control of the system, it can perform operations very quickly. A bus-mastering EISA hard drive controller, for example, achieves much greater data throughput with a fast drive than can controller cards that are not bus-mastered.

The ISP determines which device gains control by using a four-level order of priority. That order, in terms of priority, is:

- System-memory refresh
- DMA transfers
- The CPU itself
- Bus masters

A bus-mastering adapter board notifies the ISP when it wants control of the system. At the earliest possible time (after the higher priorities have been satisfied), the ISP hands control over to the bus-mastered board. The boards, in turn, have built-in circuitry to keep them from taking over the system for periods of time that would interfere with first-priority operations, such as memory refresh.

Automated Setup. EISA systems also use an automated setup to deal with adapter-board interrupts and addressing issues. These issues often cause problems when several different adapter boards are installed in an ISA system. EISA setup software recognizes potential conflicts and automatically configures the system to avoid them. EISA does, however, enable you to do your own troubleshooting, as well as to configure the boards through jumpers and switches.

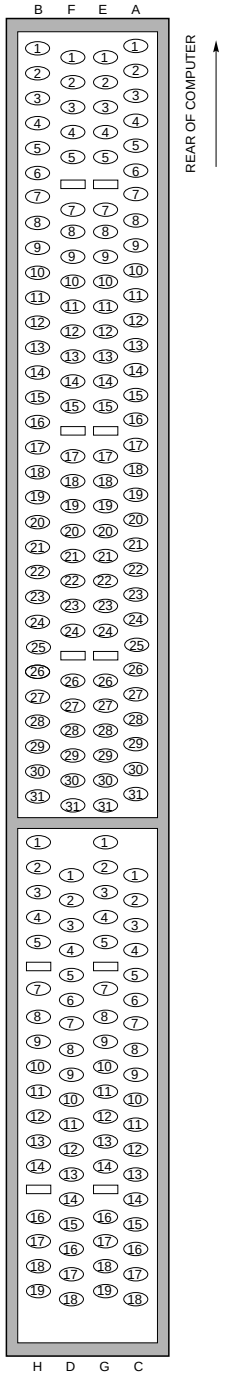

Fig. 5.9
The card connector for the EISA bus.

Note

Although automated setup traditionally has not been available in ISA systems, it should be available in the near future with the advent of plug-and-play systems and components. Plug-and-play systems are discussed toward the end of this chapter in "The Future: Plug-and-Play Systems."

Local Bus

The I/O buses discussed so far (ISA, MCA, and EISA) have one thing in common: relatively slow speed. This speed limitation is a carryover from the days of the original PC, when the I/O bus operated at the same speed as the processor bus. As the speed of the processor bus increased, the I/O bus realized only nominal speed improvements, primarily from an increase in the bandwidth of the bus. The I/O bus had to remain at a slower speed, because the huge installed base of adapter cards could operate only at slower speeds.

Figure 5.10 shows a conceptual block diagram of the buses in a computer system.

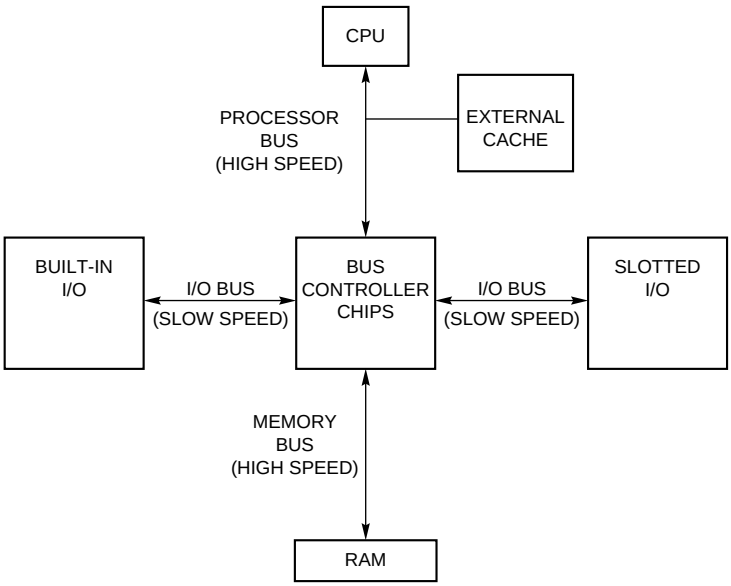

Fig. 5.10
Bus layout in a traditional PC.

The thought of a computer system running slower than it could is very bothersome to some computer users. Even so, the slow speed of the I/O bus is nothing more than a nuisance in most cases. You don't need blazing speed to communicate with a keyboard or a mouse, for example; you gain nothing in performance. The real problem occurs in subsystems in which you need the speed, such as video and disk controllers.

The speed problem became acute when graphical user interfaces (such as Windows) became prevalent. These systems required the processing of so much video data that the I/O bus became a literal bottleneck for the entire computer system. In other words, it did little good to have a CPU that was capable of 66 MHz speed if you could put data through the I/O bus at a rate of only 8 MHz.

An obvious solution to this problem is to move some of the slotted I/O to an area where it could access the faster speeds of the processor bus—much the same way as the external cache. Figure 5.11 shows this arrangement.

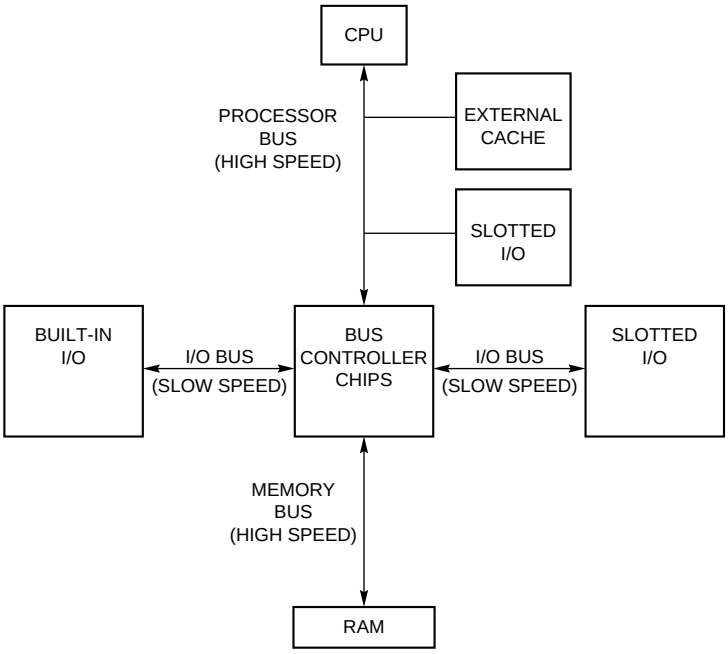

Fig. 5.11
How local bus works.

This arrangement became known as *local bus,* because external devices (adapter cards) now could access the part of the bus that was local to the CPU—the processor bus. Physically, the slots provided to tap this new configuration would need to be different from existing bus slots, to prevent adapter cards designed for slower buses from being plugged into the higher bus speeds that this design made accessible.

The first local-bus solution was introduced in 1992 as a joint effort between Dell Computer and Intel. Although the resulting system was very expensive (at first), it proved the feasibility of moving the video subsystem to the other side of the tracks, so to speak, where it could access the high speeds of the processor bus. This first pass at local bus became known, officially, as *i486 local-bus I/O.* By late 1992, the cost of local-bus systems began to decline, and many vendors began to offer similar systems.

Note

A system does not have to have a local-bus expansion slot to incorporate local-bus technology; instead, the local-bus device can be built directly into the motherboard. (In such a case, the local-bus-slotted I/O shown in fig. 5.11 would in fact be built-in I/O.) This built-in approach to local bus is the way the first local-bus systems were designed.

Local-bus solutions do not replace earlier standards, such as ISA and EISA; they are designed to augment those standards. Therefore, a typical system is based on ISA or EISA and has one or more local-bus slots available as well. Older cards still are compatible with the system, but high-speed adapter cards can take advantage of the local-bus slots as well.

Local-bus systems are especially popular with users of Windows and OS/2, because these slots are used for special 32-bit video accelerator cards that greatly speed the repainting of the graphics screens used in those operating systems. The performance of Windows and OS/2 suffers greatly from bottlenecks in even the best VGA cards connected to an ISA or EISA bus. Whereas regular VGA cards may be capable of painting 600,000 pixels per second on-screen, manufacturers of local-bus video adapters often claim that their cards can paint 50 million to 60 million pixels per second. Although performance typically is somewhat less in real-world situations, the increase in speed still is remarkable.

VESA Local Bus

At first, the local-bus slot primarily was used for video cards; this slot was where the bottleneck in top-of-the-line computer systems became apparent. Unfortunately, in late 1992, several competing local-bus systems were on the market, and each system was proprietary to its vendor. This lack of standardization hindered widespread acceptance of the local-bus solution.

To overcome this problem, the Video Electronics Standards Association (VESA) developed a standardized local-bus specification known as VESA Local Bus or simply VL-Bus. As in earlier local-bus implementations, the VL-Bus slot offers direct access to system memory at the speed of the processor itself. The VL-Bus can move data 32 bits at a time, enabling data to flow between the CPU and a compatible video subsystem or hard drive at the full 32-bit data width of the 486 chip. The maximum rated throughput of the VL-Bus is 128M to 132M per second. In other words, local bus went a long way toward removing the major bottlenecks that existed in earlier bus configurations.

Additionally, VL-Bus offers manufacturers of hard-drive interface cards an opportunity to overcome another traditional bottleneck: the rate at which data can flow between the hard drive and the CPU. The average 16-bit IDE drive and interface can achieve throughput of up to 5M per second, whereas VL-Bus hard drive adapters for IDE drives are touted as providing throughput of as much as 8M per second. In real-world situations, the true throughput of VL-Bus hard drive adapters is somewhat less than 8M per second, but VL-Bus still provides a substantial boost in hard-drive performance.

Despite all the benefits of the VL-Bus (and, by extension, of all local buses), this technology has a few drawbacks, which are described in the following list:

- *Dependence on a 486 CPU*. The VL-Bus inherently is tied to the 486 processor bus. This bus is quite different from that used by Pentium processors (and probably from those that will be used by future CPUs). A VL-Bus that operates at the full rated speed of a Pentium has not been developed, although stopgap measures (such as stepping down speed or developing bus bridges) are available.
- *Speed limitations*. The VL-Bus specification provides for speeds of up to 66 MHz on the bus, but the electrical characteristics of the VL-Bus connector limit an adapter card to no more than 50 MHz. If the main CPU uses a clock modifier (such as the kind that doubles clock speeds), the VL-Bus uses the unmodified CPU clock speed as its bus speed.
- *Electrical limitations*. The processor bus has very tight timing rules, which may vary from CPU to CPU. These timing rules were designed for limited loading on the bus, meaning that the only elements originally intended to be connected to the local bus are elements such as the external cache and the bus controller chips. As you add more circuitry, you increase the electrical load. If the local bus is not implemented correctly, the additional load can lead to problems such as loss of data integrity and timing problems between the CPU and the VL-Bus cards.
- *Card limitations*. Depending on the electrical loading of a system, the number of VL-Bus cards is limited. Although the VL-Bus specification provides for as many as three cards, this can be achieved only at clock rates of up to 40 MHz with an otherwise low system-board load. As the system-board load increases and the clock rate increases, the number of cards supported decreases. Only one VL-Bus card can be supported at 50 MHz with a high system-board load.

These drawbacks should not dissuade you from investing in a VL-Bus system; indeed, they provide an acceptable solution for high-speed computing. Many critics, however, complain that even though the VL-Bus is well suited to 486 systems, it is just that—a modified 486 system. These people contend that the VL-Bus is not adaptable or extensible, meaning that it does not work well for nonvideo needs and will not work well (without modification) for future generations of CPUs.

Caution

If you are contemplating the purchase of a local-bus system, you should make sure that it is a VL-Bus system. If the system instead uses a proprietary design, you may have a hard time finding adapter cards that take advantage of the local-bus slots. If you are convinced that another local-bus standard is superior, you should invest in the motherboard only after you ensure that you can buy video cards and hard drive adapters that take full advantage of the proprietary design.

Physically, the VL-Bus slot is an extension of the slots used for whatever type of base system you have. If you have an ISA system, the VL-Bus is positioned as an extension of your existing 16-bit ISA slots. Likewise, if you have an EISA system or MCA system, the VL-Bus slots are extensions of those existing slots. Figure 5.12 shows how the VL-Bus slots could be situated in an EISA system.

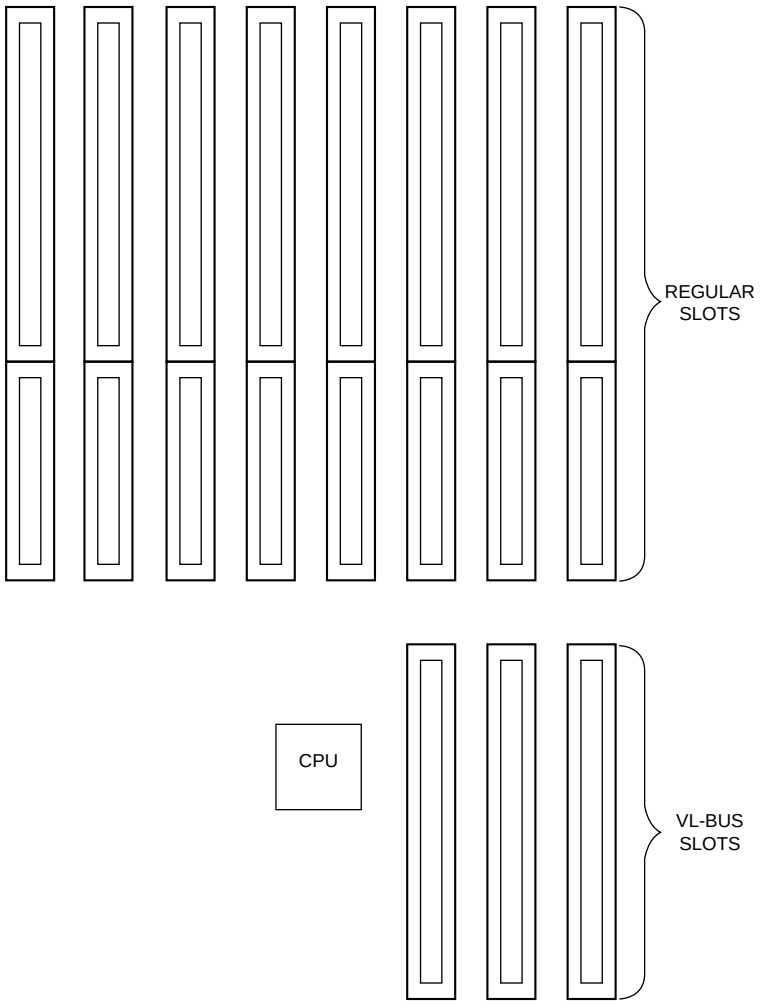

Fig. 5.12
An example of VL-Bus slots in an EISA system.

The VESA extension has 112 contacts and uses the same physical connector as the MCA bus. Table 5.9 describes the physical sizes of the various VL-Bus cards. (The sizes represent the main board only; the depths of the edge connectors are not included.)

Table 5.9 Size Specifications for Various VL-Bus Adapter Cards

System	Height	Length	Thickness
ISA	4.48"	13.4"	0.062"
MCA	2.95"	11.5"	0.063"
EISA	4.48"	13.4"	0.062"

The VL-Bus adds a total 116 pin locations to the bus connectors that your system already has. The complete pinout description for a VL-Bus card depends on which type of primary bus the VL-Bus is being used with. Table 5.10 lists the pinouts for only the VL-Bus connector portion of the total connector. (For pins for which two purposes are listed, the second purpose applies when the card is in 64-bit transfer mode.)

Table 5.10 Pinouts for the VL-Bus

Pin	Signal Name	Pin	Signal Name
B1	Data 0	A1	Data 1
B2	Data 2	A2	Data 3
B3	Data 4	A3	Ground
B4	Data 6	A4	Data 5
B5	Data 8	A5	Data 7
B6	Ground	A6	Data 9
B7	Data 10	A7	Data 11
B8	Data 12	A8	Data 13
B9	VCC	A9	Data 15
B10	Data 14	A10	Ground
B11	Data 16	A11	Data 17
B12	Data 18	A12	VCC
B13	Data 20	A13	Data 19
B14	Ground	A14	Data 21
B15	Data 22	A15	Data 23
B16	Data 24	A16	Data 25
B17	Data 26	A17	Ground
B18	Data 28	A18	Data 27
B19	Data 30	A19	Data 29
B20	VCC	A20	Data 31
B21	Address 31 or Data 63	A21	Address 30 or Data 62
B22	Ground	A22	Address 28 or Data 60
B23	Address 29 or Data 61	A23	Address 26 or Data 58
B24	Address 27 or Data 59	A24	Ground
B25	Address 25 or Data 57	A25	Address 24 or Data 56

(continues)

Table 5.10 Continued

Pin	Signal Name	Pin	Signal Name
B26	Address 23 or Data 55	A26	Address 22 or Data 54
B27	Address 21 or Data 53	A27	VCC
B28	Address 19 or Data 51	A28	Address 20 or Data 52
B29	Ground	A29	Address 18 or Data 50
B30	Address 17 or Data 49	A30	Address 16 or Data 48
B31	Address 15 or Data 47	A31	Address 14 or Data 46
B32	VCC	A32	Address 12 or Data 44
B33	Address 13 or Data 45	A33	Address 10 or Data 42
B34	Address 11 or Data 43	A34	Address 8 or Data 40
B35	Address 9 or Data 41	A35	Ground
B36	Address 7 or Data 39	A36	Address 6 or Data 38
B37	Address 5 or Data 37	A37	Address 4 or Data 36
B38	Ground	A38	Write Back
B39	Address 3 or Data 35	A39	Byte Enable 0 or 4
B40	Address 2 or Data 34	A40	VCC
B41	Unused or LBS64#	A41	Byte Enable 1 or 5
B42	Reset	A42	Byte Enable 2 or 6
B43	Data/Code Status	A43	Ground
B44	Memory-I/O Status or Data 33	A44	Byte Enable 3 or 7
B45	Write/Read Status or Data 32	A45	Address Data Strobe
B46	Access key	A46	Access key
B47	Access key	A47	Access key
B48	Ready Return	A48	Local Ready
B49	Ground	A49	Local Device
B50	IRQ 9	A50	Local Request
B51	Burst Ready	A51	Ground
B52	Burst Last	A52	Local Bus Grant
B53	ID0	A53	VCC
B54	ID1	A54	ID2
B55	Ground	A55	ID3
B56	Local Clock	A56	ID4 or ACK64#
B57	VCC	A57	Unused
B58	Local Bus Size 16	A58	Loc/Ext Address Data Strobe

Figure 5.13 shows the locations of the pins.

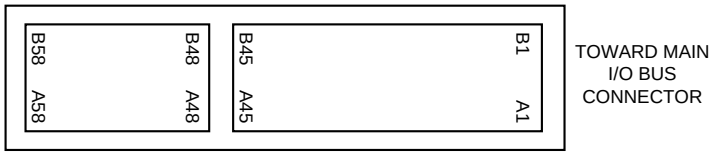

Fig. 5.13
The card connector for the VL-Bus.

The PCI Bus

In early 1992, Intel spearheaded the creation of another industry group which was formed with the same goals as the VESA group in relation to the PC bus. Recognizing the need to overcome weaknesses in the ISA and EISA buses, the PCI Special Interest Group was formed.

PCI is an acronym for *peripheral component interconnect bus*. The PCI bus specification, released in June 1992 and updated in April 1993, redesigned the traditional PC bus by inserting another bus between the CPU and the native I/O bus by means of bridges. Rather than tap directly into the processor bus, with its delicate electrical timing (as was done in local bus and VL-Bus), a new set of controller chips was developed to extend the bus, as shown in figure 5.14.

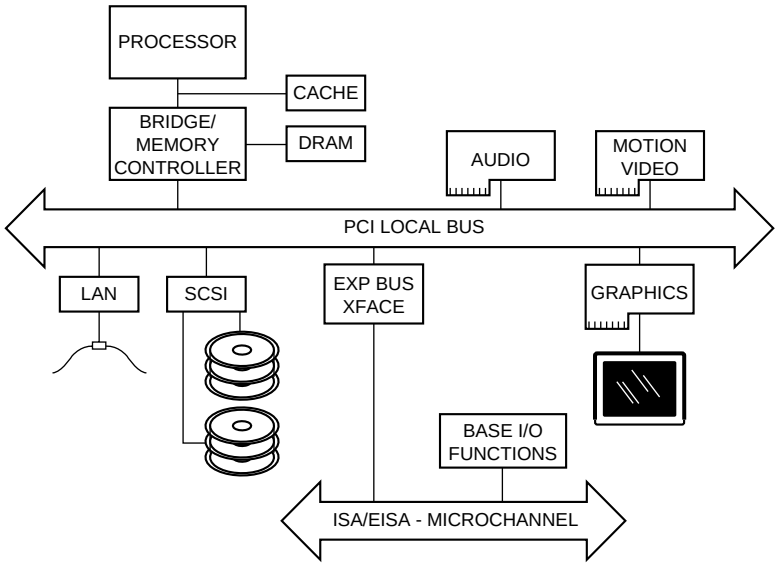

Fig. 5.14
Conceptual diagram of the PCI bus.

The PCI bus often is called a *mezzanine bus* because it adds another layer to the traditional bus configuration. PCI bypasses the standard I/O bus; it uses the system bus to increase the bus clock speed and take full advantage of the CPU's data path. Systems that integrate the PCI bus became available in mid-1993 and have since become the mainstay high-end systems.

Although the PCI bus is the clear choice for Pentium-based systems, some critics have contended that for 486 systems, the VL-Bus is the winner because it is cheaper to implement. These critics base their contention on the fact that extra chips and pins are necessary to implement the mezzanine configuration inherent to PCI. This argument is incomplete, however; in reality, connecting an I/O chip to the VL-Bus requires almost twice as many pins as connecting it to the PCI bus does (88, compared with 47). Therefore, PCI versions of chips (the ones built into the motherboard) should be less expensive than the VL-Bus versions.

Information is transferred across the PCI bus at 33 MHz, at the full data width of the CPU. When the bus is used in conjunction with a 32-bit CPU, the bandwidth is 132M per second, as the following formula shows:

33 MHz × 32 bits = 1,056 megabits/second

1,056 megabits/second ÷ 8 = 132M/second

When the bus is used with a 64-bit CPU, the bandwidth doubles, meaning that you can transfer data at speeds up to 264M per second. Real-life data-transfer speeds necessarily will be lower, but still will be much faster than anything else that is currently available. Part of the reason for this faster real-life throughput is the fact that the PCI bus can operate concurrently with the processor bus; it does not supplant it. The CPU can be processing data in an external cache while the PCI bus is busy transferring information between other parts of the system—a major design benefit of the PCI bus.

A PCI adapter card uses a standard MCA connector, just like the VL-Bus. This connector can be identified within a computer system because it typically is offset from the normal ISA, MCA, or EISA connectors (see fig. 5.15 for an example). The size of a PCI card can be the same as that of the cards used in the system's normal I/O bus.

The PCI specification identifies three board configurations, each designed for a specific type of system with specific power requirements. The 5-volt specification is for stationary computer systems, the 3.3-volt specification is for portable machines, and the universal specification is for motherboards and cards that work in either type of system.

Table 5.11 shows the 5-volt PCI pinouts, and figure 5.16 shows the pin locations. Table 5.12 shows the 3.3-volt PCI pinouts; the pin locations are indicated in figure 5.17. Finally, table 5.13 shows the pinouts, and figure 5.18 shows the pin locations, for a universal PCI slot and card. Notice that each figure shows both the 32-bit and 64-bit variations on the respective specifications.

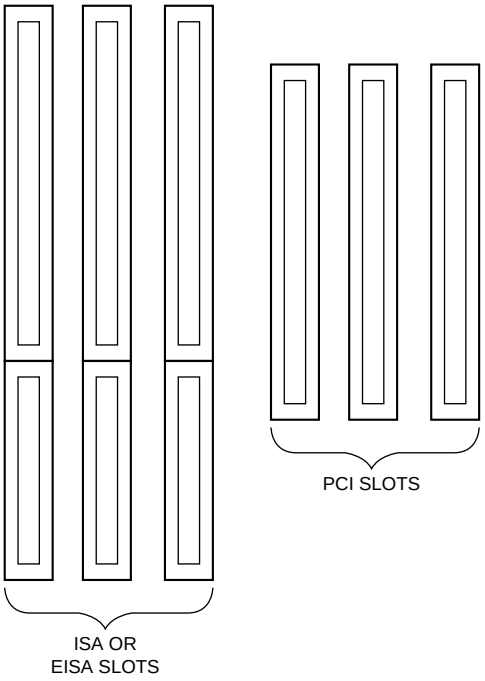

Fig. 5.15
Possible configuration of PCI slots in relation to ISA or EISA slots.

Note

If the PCI card supports only 32 data bits, it needs only pins B1/A1 through B62/A62. Pins B63/A63 through B94/A94 are used only if the card supports 64 data bits.

Table 5.11 Pinouts for a 5-Volt PCI Bus

Pin	Signal Name	Pin	Signal Name
B1	–12V	A1	Test Reset
B2	Test Clock	A2	+12V
B3	Ground	A3	Test Mode Select
B4	Test Data Output	A4	Test Data Input
B5	+5V	A5	+5V
B6	+5V	A6	Interrupt A
B7	Interrupt B	A7	Interrupt C
B8	Interrupt D	A8	+5V

(continues)

Table 5.11 Continued

Pin	Signal Name	Pin	Signal Name
B9	PRSNT1#	A9	Reserved
B10	Reserved	A10	+5V I/O
B11	PRSNT2#	A11	Reserved
B12	Ground	A12	Ground
B13	Ground	A13	Ground
B14	Reserved	A14	Reserved
B15	Ground	A15	Reset
B16	Clock	A16	+5V I/O
B17	Ground	A17	Grant
B18	Request	A18	Ground
B19	+5V I/O	A19	Reserved
B20	Address 31	A20	Address 30
B21	Address 29	A21	+3.3V
B22	Ground	A22	Address 28
B23	Address 27	A23	Address 26
B24	Address 25	A24	Ground
B25	+3.3V	A25	Address 24
B26	C/BE 3	A26	Init Device Select
B27	Address 23	A27	+3.3V
B28	Ground	A28	Address 22
B29	Address 21	A29	Address 20
B30	Address 19	A30	Ground
B31	+3.3V	A31	Address 18
B32	Address 17	A32	Address 16
B33	C/BE 2	A33	+3.3V
B34	Ground	A34	Cycle Frame
B35	Initiator Ready	A35	Ground
B36	+3.3V	A36	Target Ready
B37	Device Select	A37	Ground
B38	Ground	A38	Stop
B39	Lock	A39	+3.3V
B40	Parity Error	A40	Snoop Done
B41	+3.3V	A41	Snoop Backoff
B42	System Error	A42	Ground
B43	+3.3V	A43	PAR
B44	C/BE 1	A44	Address 15
B45	Address 14	A45	+3.3V
B46	Ground	A46	Address 13
B47	Address 12	A47	Address 11

Pin	Signal Name	Pin	Signal Name
B48	Address 10	A48	Ground
B49	Ground	A49	Address 9
B50	Access key	A50	Access key
B51	Access key	A51	Access key
B52	Address 8	A52	C/BE 0
B53	Address 7	A53	+3.3V
B54	+3.3V	A54	Address 6
B55	Address 5	A55	Address 4
B56	Address 3	A56	Ground
B57	Ground	A57	Address 2
B58	Address 1	A58	Address 0
B59	+5V I/O	A59	+5V I/O
B60	Acknowledge 64-bit	A60	Request 64-bit
B61	+5V	A61	+5V
B62	+5V Access key	A62	+5V Access key
B63	Reserved	A63	Ground
B64	Ground	A64	C/BE 7
B65	C/BE 6	A65	C/BE 5
B66	C/BE 4	A66	+5V I/O
B67	Ground	A67	Parity 64-bit
B68	Address 63	A68	Address 62
B69	Address 61	A69	Ground
B70	+5V I/O	A70	Address 60
B71	Address 59	A71	Address 58
B72	Address 57	A72	Ground
B73	Ground	A73	Address 56
B74	Address 55	A74	Address 54
B75	Address 53	A75	+5V I/O
B76	Ground	A76	Address 52
B77	Address 51	A77	Address 50
B78	Address 49	A78	Ground
B79	+5V I/O	A79	Address 48
B80	Address 47	A80	Address 46
B81	Address 45	A81	Ground
B82	Ground	A82	Address 44
B83	Address 43	A83	Address 42
B84	Address 41	A84	+5V I/O
B85	Ground	A85	Address 40

(continues)

Table 5.11 Continued

Pin	Signal Name	Pin	Signal Name
B86	Address 39	A86	Address 38
B87	Address 37	A87	Ground
B88	+5V I/O	A88	Address 36
B89	Address 35	A89	Address 34
B90	Address 33	A90	Ground
B91	Ground	A91	Address 32
B92	Reserved	A92	Reserved
B93	Reserved	A93	Ground
B94	Ground	A94	Reserved

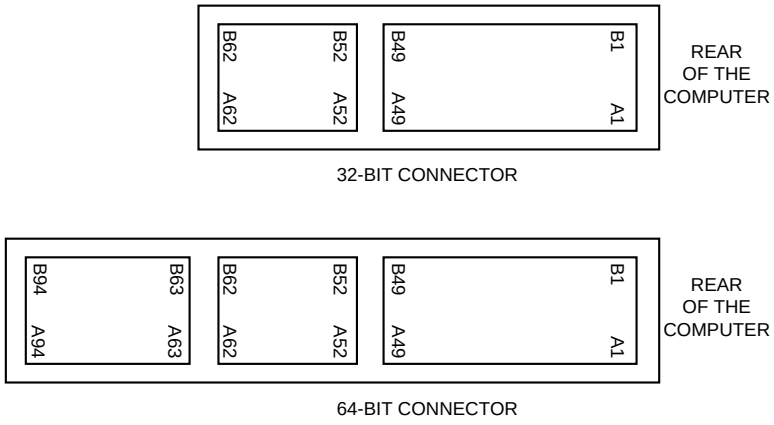

Fig. 5.16
The 5-volt PCI slot and card configuration.

Table 5.12 Pinouts for a 3.3-Volt PCI Bus

Pin	Signal Name	Pin	Signal Name
B1	–12V	A1	Test Reset
B2	Test Clock	A2	+12V
B3	Ground	A3	Test Mode Select
B4	Test Data Output	A4	Test Data Input
B5	+5V	A5	+5V
B6	+5V	A6	Interrupt A
B7	Interrupt B	A7	Interrupt C
B8	Interrupt D	A8	+5V
B9	PRSNT1#	A9	Reserved

Pin	Signal Name	Pin	Signal Name
B10	Reserved	A10	+3.3V
B11	PRSNT2#	A11	Reserved
B12	Access key	A12	Access key
B13	Access key	A13	Access key
B14	Reserved	A14	Reserved
B15	Ground	A15	Reset
B16	Clock	A16	+3.3V
B17	Ground	A17	Grant
B18	Request	A18	Ground
B19	+3.3V	A19	Reserved
B20	Address 31	A20	Address 30
B21	Address 29	A21	+3.3V
B22	Ground	A22	Address 28
B23	Address 27	A23	Address 26
B24	Address 25	A24	Ground
B25	+3.3V	A25	Address 24
B26	C/BE 3	A26	Init Device Select
B27	Address 23	A27	+3.3V
B28	Ground	A28	Address 22
B29	Address 21	A29	Address 20
B30	Address 19	A30	Ground
B31	+3.3V	A31	Address 18
B32	Address 17	A32	Address 16
B33	C/BE 2	A33	+3.3V
B34	Ground	A34	Cycle Frame
B35	Initiator Ready	A35	Ground
B36	+3.3V	A36	Target Ready
B37	Device Select	A37	Ground
B38	Ground	A38	Stop
B39	Lock	A39	+3.3V
B40	Parity Error	A40	Snoop Done
B41	+3.3V	A41	Snoop Backoff
B42	System Error	A42	Ground
B43	+3.3V	A43	PAR
B44	C/BE 1	A44	Address 15
B45	Address 14	A45	+3.3V
B46	Ground	A46	Address 13
B47	Address 12	A47	Address 11
B48	Address 10	A48	Ground

(continues)

Table 5.12 Continued

Pin	Signal Name	Pin	Signal Name
B49	Ground	A49	Address 9
B50	Ground	A50	Ground
B51	Ground	A51	Ground
B52	Address 8	A52	C/BE 0
B53	Address 7	A53	+3.3V
B54	+3.3V	A54	Address 6
B55	Address 5	A55	Address 4
B56	Address 3	A56	Ground
B57	Ground	A57	Address 2
B58	Address 1	A58	Address 0
B59	+3.3V	A59	+3.3V
B60	Acknowledge 64-bit	A60	Request 64-bit
B61	+5V	A61	+5V
B62	+5V Access key	A62	+5V Access key
B63	Reserved	A63	Ground
B64	Ground	A64	C/BE 7
B65	C/BE 6	A65	C/BE 5
B66	C/BE 4	A66	+3.3V
B67	Ground	A67	Parity 64-bit
B68	Address 63	A68	Address 62
B69	Address 61	A69	Ground
B70	+3.3V	A70	Address 60
B71	Address 59	A71	Address 58
B72	Address 57	A72	Ground
B73	Ground	A73	Address 56
B74	Address 55	A74	Address 54
B75	Address 53	A75	+3.3V
B76	Ground	A76	Address 52
B77	Address 51	A77	Address 50
B78	Address 49	A78	Ground
B79	+3.3V	A79	Address 48
B80	Address 47	A80	Address 46
B81	Address 45	A81	Ground
B82	Ground	A82	Address 44
B83	Address 43	A83	Address 42
B84	Address 41	A84	+3.3V
B85	Ground	A85	Address 40
B86	Address 39	A86	Address 38

Pin	Signal Name	Pin	Signal Name
B87	Address 37	A87	Ground
B88	+3.3V	A88	Address 36
B89	Address 35	A89	Address 34
B90	Address 33	A90	Ground
B91	Ground	A91	Address 32
B92	Reserved	A92	Reserved
B93	Reserved	A93	Ground
B94	Ground	A94	Reserved

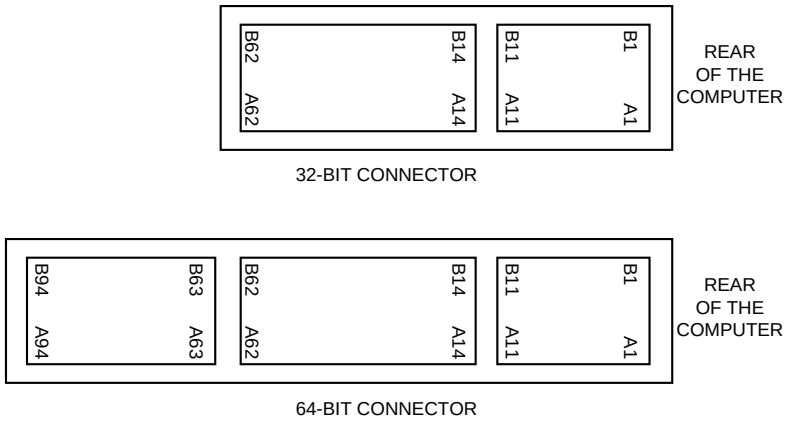

Fig. 5.17
The 3.3-volt PCI slot and card configuration.

Table 5.13 Pinouts for a Universal PCI Bus

Pin	Signal Name	Pin	Signal Name
B1	–12V	A1	Test Reset
B2	Test Clock	A2	+12V
B3	Ground	A3	Test Mode Select
B4	Test Data Output	A4	Test Data Input
B5	+5V	A5	+5V
B6	+5V	A6	Interrupt A
B7	Interrupt B	A7	Interrupt C
B8	Interrupt D	A8	+5V
B9	PRSNT1#	A9	Reserved
B10	Reserved	A10	+V I/O
B11	PRSNT2#	A11	Reserved

(continues)

Table 5.13 Continued

Pin	Signal Name	Pin	Signal Name
B12	Access key	A12	Access key
B13	Access key	A13	Access key
B14	Reserved	A14	Reserved
B15	Ground	A15	Reset
B16	Clock	A16	+V I/O
B17	Ground	A17	Grant
B18	Request	A18	Ground
B19	+V I/O	A19	Reserved
B20	Address 31	A20	Address 30
B21	Address 29	A21	+3.3V
B22	Ground	A22	Address 28
B23	Address 27	A23	Address 26
B24	Address 25	A24	Ground
B25	+3.3V	A25	Address 24
B26	C/BE 3	A26	Init Device Select
B27	Address 23	A27	+3.3V
B28	Ground	A28	Address 22
B29	Address 21	A29	Address 20
B30	Address 19	A30	Ground
B31	+3.3V	A31	Address 18
B32	Address 17	A32	Address 16
B33	C/BE 2	A33	+3.3V
B34	Ground	A34	Cycle Frame
B35	Initiator Ready	A35	Ground
B36	+3.3V	A36	Target Ready
B37	Device Select	A37	Ground
B38	Ground	A38	Stop
B39	Lock	A39	+3.3V
B40	Parity Error	A40	Snoop Done
B41	+3.3V	A41	Snoop Backoff
B42	System Error	A42	Ground
B43	+3.3V	A43	PAR
B44	C/BE 1	A44	Address 15
B45	Address 14	A45	+3.3V
B46	Ground	A46	Address 13
B47	Address 12	A47	Address 11
B48	Address 10	A48	Ground
B49	Ground	A49	Address 9
B50	Access key	A50	Access key

Pin	Signal Name	Pin	Signal Name
B51	Access key	A51	Access key
B52	Address 8	A52	C/BE 0
B53	Address 7	A53	+3.3V
B54	+3.3V	A54	Address 6
B55	Address 5	A55	Address 4
B56	Address 3	A56	Ground
B57	Ground	A57	Address 2
B58	Address 1	A58	Address 0
B59	+5 I/O	A59	+V I/O
B60	Acknowledge 64-bit	A60	Request 64-bit
B61	+5V	A61	+5V
B62	+5V Access key	A62	+5V Access key
B63	Reserved	A63	Ground
B64	Ground	A64	C/BE 7
B65	C/BE 6	A65	C/BE 5
B66	C/BE 4	A66	+V I/O
B67	Ground	A67	Parity 64-bit
B68	Address 63	A68	Address 62
B69	Address 61	A69	Ground
B70	+V I/O	A70	Address 60
B71	Address 59	A71	Address 58
B72	Address 57	A72	Ground
B73	Ground	A73	Address 56
B74	Address 55	A74	Address 54
B75	Address 53	A75	+V I/O
B76	Ground	A76	Address 52
B77	Address 51	A77	Address 50
B78	Address 49	A78	Ground
B79	+V I/O	A79	Address 48
B80	Address 47	A80	Address 46
B81	Address 45	A81	Ground
B82	Ground	A82	Address 44
B83	Address 43	A83	Address 42
B84	Address 41	A84	+V I/O
B85	Ground	A85	Address 40
B86	Address 39	A86	Address 38
B87	Address 37	A87	Ground
B88	+V I/O	A88	Address 36

(continues)

Table 5.13 Continued

Pin	Signal Name	Pin	Signal Name
B89	Address 35	A89	Address 34
B90	Address 33	A90	Ground
B91	Ground	A91	Address 32
B92	Reserved	A92	Reserved
B93	Reserved	A93	Ground
B94	Ground	A94	Reserved

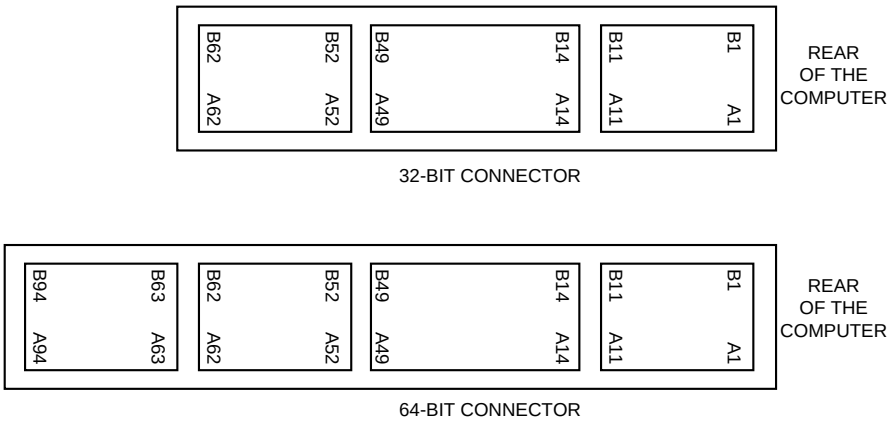

Fig. 5.18
The universal PCI slot and card configuration.

Notice that the universal PCI board specifications effectively combine the 5-volt and 3.3-volt specifications. For pins for which the voltage is different, the universal specification labels the pin simply *V I/O*. This type of pin represents a special power pin for defining and driving the PCI signaling rail.

The PCMCIA Bus

In an effort to give laptop and notebook computers the kind of expandability that users have grown used to in desktop systems, the Personal Computer Memory Card International Association (PCMCIA) has established several standards for credit-card-size expansion boards that fit into a small slot on laptops and notebooks.

The PCMCIA standards, which were developed by a consortium of more than 300 manufacturers (including IBM, Toshiba, and Apple), have been touted as being a revolutionary advance in mobile computing, because PCMCIA laptop and notebook slots enable you to add memory expansion cards, fax/modems, SCSI adapters, local-area-network cards, and many other types of devices. The idea behind PCMCIA is to enable you to plug any manufacturer's PCMCIA peripheral into your notebook computer.

The promise of the 2.1- by 3.4-inch PCMCIA 68-pin cards is enormous. Marketers envision not only memory expansion cards, tiny hard drives, and wireless modems (which already are available), but also wireless LAN connectors, PCMCIA sound cards, CD-ROM controllers, tape backup drives, and a host of other peripherals. Current PCMCIA devices cost considerably more than the same devices for ISA desktop systems. In version 2 of the PCMCIA card standard, devices can be longer, which will help manufacturers design some advanced peripheral cards.

PCMCIA has just one drawback: the standard has been followed loosely by manufacturers of computers and peripheral devices. If you purchase a laptop or notebook computer with an eye toward expandability, you must do your homework before buying the system; some devices that are advertised as being fully PCMCIA-compatible do not work with other systems that advertise themselves as being fully PCMCIA-compatible. If you already have a PCMCIA-bus computer, the only safe way to buy PCMCIA cards is to contact the device manufacturer and determine whether the device has been tested in your computer. Before you purchase a PCMCIA-compatible computer, you should get from the manufacturer a list of devices with which the computer will work.

In an effort to solve these compatibility problems, the PCMCIA has continued to establish standards. At this writing, in fact, four standards exist: PCMCIA Type I through Type IV. Even with all the PCMCIA types, compatibility problems remain, mostly because PCMCIA standards are voluntary; some manufacturers do not fully implement these standards before advertising their products as being PCMCIA-compatible. The standards have, however, helped make more and more PCMCIA computers and peripherals compatible with one another.

The following list describes the major features of the PCMCIA standards:

- *PCMCIA Type I*. The original PCMCIA standard, now called Type I slots, can handle cards that are 3.3mm thick. These slots work only with memory expansion cards. If you are shopping for a PCMCIA memory expansion card, check with your system manufacturer before buying to ensure that the card you buy will work with your system.
- *PCMCIA Type II*. PCMCIA Type II slots accommodate cards that are 5mm thick but otherwise have the same physical form factor as Type I cards. PCMCIA Type II slots also can be used with Type I cards because the guides that hold the cards are the same thickness; the center portion of the slot provides more room. Type II cards support virtually any type of expansion device (such as a modem) or LAN adapter.
- *PCMCIA Type III*. In late 1992, PCMCIA Type III was introduced. These slots, intended primarily for computers that have removable hard drives, are 10.5mm thick, but they also are compatible with Type I and Type II cards.
- *PCMCIA Type IV*. PCMCIA Type IV slots are intended to be used with hard drives that are thicker than the 10.5mm Type III slot allows. The exact dimensions of the slot have not yet been determined, but Type IV slots are expected to be compatible with Types I through III.

Table 5.14 shows the PCMCIA pinouts.

Table 5.14 Pinouts for a PCMCIA Card

Pin	Signal Name
1	Ground
2	Data 3
3	Data 4
4	Data 5
5	Data 6
6	Data 7
7	–Card Enable 1
8	Address 10
9	–Output Enable
10	Address 11
11	Address 9
12	Address 8
13	Address 13
14	Address 14
15	–Write Enable/–Program
16	Ready/–Busy (IREQ)
17	+5V
18	Vpp1
19	Address 16
20	Address 15
21	Address 12
22	Address 7
23	Address 6
24	Address 5
25	Address 4
26	Address 3
27	Address 2
28	Address 1
29	Address 0
30	Data 0
31	Data 1
32	Data 2
33	Write Protect (–IOIS16)
34	Ground
35	Ground
36	–Card Detect 1
37	Data 11

Pin	Signal Name
38	Data 12
39	Data 13
40	Data 14
41	Data 15
42	–Card Enable 2
43	Refresh
44	RFU (–IOR)
45	RFU (–IOW)
46	Address 17
47	Address 18
48	Address 19
49	Address 20
50	Address 21
51	+5V
52	Vpp2
53	Address 22
54	Address 23
55	Address 24
56	Address 25
57	RFU
58	RESET
59	–WAIT
60	RFU (–INPACK)
61	–Register Select
62	Battery Voltage Detect 2 (–SPKR)
63	Battery Voltage Detect 1 (–STSCHG)
64	Data 8
65	Data 9
66	Data 10
67	–Card Detect 2
68	Ground

System Resources

In various chapters in Parts II–IV, you learn about system resources. At their lowest level, these resources typically include the following:

- I/O port addresses
- IRQ numbers

- DMA channels
- Memory

These resources are required and used by many different components of your system. Adapter cards need these resources to communicate with your system and to accomplish their purposes. Not all adapter cards have the same resource requirements. A serial communications port, for example, needs an I/O port address and an IRQ number, whereas a sound board needs these resources and at least one DMA channel.

As your system increases in complexity, the chance for resource conflicts increases dramatically. You naturally have a conflict if you have two or three cards, each of which needs an IRQ number or an I/O port address. So that you can resolve conflicts, most adapter cards allow you to modify resource assignments by setting jumpers or switches on the cards.

I/O Port Addresses

Your computer's I/O ports enable you to attach a large number of important devices to your system to expand its capabilities. A printer attached to one of your system's LPT (parallel) ports enables you to make a printout of the work on your system. A modem attached to one of your system's COM (serial) ports enables you to use telephone lines to communicate with computers thousands of miles away. A scanner attached to an LPT port or a SCSI host adapter enables you to convert graphics or text to images and type that you can use with the software installed on your computer.

Most systems come configured with at least two COM (serial) ports and one LPT (parallel printer) ports. The two serial ports are configured as COM1 and COM2, and the parallel port as LPT1. The basic architecture of the PC provides for as many as four COM ports (1 through 4) and three LPT ports (1 through 3).

Caution

Theoretically, each of the four COM ports in a system can be used to attach a device, such as a mouse or modem, but doing so may lead to resource conflicts. For more information, see the discussion of resolving IRQ conflicts in "IRQ Conflicts" later in this chapter.

Every I/O port in your system uses an I/O address for communication. This address, which is in the lower memory ranges, is reserved for communication between the I/O device and the operating system. If your system has multiple I/O cards, each card must use a different I/O address; if not, your system will not be able to communicate with the devices reliably.

The I/O addresses that your ports use depend on the type of ports. Table 5.15 shows the I/O addresses expected by the various standard ports in a PC system.

Table 5.15 Standard I/O Addresses for Serial and Parallel Ports

Port	I/O Address
COM1	03F8
COM2	02F8
COM3	03E8
COM4	02E8
LPT1	03BC
LPT2	0378
LPT3	0278

Besides your serial and parallel ports, other adapters in your system use I/O addresses. Quite truthfully, the I/O addresses for the serial and parallel ports are fairly standard; it is unlikely that you will run into problems with them. The I/O addresses used by other adapters are not standardized, however, and you may have problems finding a mix of port addresses that works reliably. Later in this chapter, in "Resolving Resource Conflicts," you learn some of the techniques that you can use to solve this problem.

Interrupts (IRQs)

Interrupt request channels (IRQs), or *hardware interrupts,* are used by various hardware devices to signal the motherboard that a request must be fulfilled. This procedure is the same as a student raising his hand to indicate that he needs attention.

These interrupt channels are represented by wires on the motherboard and in the slot connectors. When a particular interrupt is invoked, a special routine takes over the system, which first saves all the CPU register contents in a stack and then directs the system to the interrupt vector table. This vector table contains a list of memory addresses that correspond to the interrupt channels. Depending on which interrupt was invoked, the program corresponding to that channel is run.

The pointers in the vector table point to the address of whatever software driver is used to service the card that generated the interrupt. For a network card, for example, the vector may point to the address of the network drivers that have been loaded to operate the card; for a hard disk controller, the vector may point to the BIOS code that operates the controller.

After the particular software routine finishes performing whatever function the card needed, the interrupt-control software returns the stack contents to the CPU registers, and the system then resumes whatever it was doing before the interrupt occurred.

Through the use of interrupts, your system can respond to external events in a timely fashion. Each time that a serial port presents a byte to your system, an interrupt is generated to ensure that the system reads that byte before another comes in.

Hardware interrupts are prioritized by their numbers; the highest-priority interrupts have the lowest numbers. Higher-priority interrupts take precedence over lower-priority interrupts by interrupting them. As a result, several interrupts can occur in your system concurrently, each interrupt nesting within another.

If you overload the system—in this case, by running out of stack resources (too many interrupts were generated too quickly)—an internal stack overflow error occurs, and your system halts. If you experience this type of system error, you can compensate for it by using the STACKS parameter in your CONFIG.SYS file to increase the available stack resources.

The ISA bus uses edge-triggered interrupt sensing, in which an interrupt is sensed by a signal sent on a particular wire located in the slot connector. A different wire corresponds to each possible hardware interrupt. Because the motherboard cannot recognize which slot contains the card that used an interrupt line and therefore generated the interrupt, confusion would result if more than one card were set to use a particular interrupt. Each interrupt, therefore, usually is designated for a single hardware device, and most of the time, interrupts cannot be shared.

A device can be designed to share interrupts, and a few devices allow this; most cannot, however, because of the way interrupts are signaled in the ISA bus. Systems with the MCA bus use level-sensitive interrupts, which allow complete interrupt sharing to occur. In fact, in an MCA system, all boards can be set to the same interrupt with no conflicts or problems. For maximum performance, however, interrupts should be staggered as much as possible.

Because interrupts usually cannot be shared in an ISA bus system, you often run out of interrupts when you are adding boards to a system. If two boards use the same IRQ to signal the system, the resulting conflict prevents either board from operating properly. The following sections discuss the IRQs that any standard devices use, as well as what may be free in your system.

8-Bit ISA Bus Interrupts. The PC and XT (the systems based on the 8-bit 8086 CPU) provide for eight different external hardware interrupts. Table 5.16 shows the typical uses for these interrupts, which are numbered 0 through 7.

Table 5.16 8-Bit ISA Bus Default Interrupt Assignments

IRQ	Function	Bus Slot
0	System Timer	No
1	Keyboard Controller	No
2	Available	No
3	Serial Port 2 (COM2)	Yes (8-bit)
4	Serial Port 1 (COM1)	Yes (8-bit)
5	Hard Disk Controller	Yes (8-bit)
6	Floppy Disk Controller	Yes (8-bit)
7	Parallel Port 1 (LPT1)	Yes (8-bit)

External hardware interrupts often are referred to as *maskable interrupts*, which simply means that the interrupts can be masked or turned off for a short time while the CPU is used for other critical operations. It is up to the programmer to manage interrupts properly and efficiently for the best system performance.

If you have a system that has one of the original 8-bit ISA buses, you will find that the IRQ resources provided by the system present a severe limitation. Installing several devices that need the services of system IRQs in a PC/XT-type system can be a study in frustration because the only way to resolve the interrupt-shortage problem is to remove the adapter board that you need the least.

16-Bit ISA, EISA, and MCA Bus Interrupts. The introduction of the AT, based on the 80286 processor, was accompanied by an increase in the number of external hardware interrupts that the bus would support. The number of interrupts was doubled, to 16, by using two interrupt controllers, piping the interrupts generated by the second one through the unused IRQ 2 in the first controller. This arrangement effectively means that only 15 IRQ assignments are available, because IRQ 2 and IRQ 9 mirror each other.

Table 5.17 shows the typical uses for interrupts in the 16-bit ISA, EISA, and MCA buses.

Table 5.17 16-Bit ISA, EISA, and MCA Default Interrupt Assignments

IRQ	Standard Function	Bus Slot
0	System Timer	No
1	Keyboard Controller	No
2	Second IRQ Controller	No
3	Serial Port 2 (COM2)	Yes (8-bit)
4	Serial Port 1 (COM1)	Yes (8-bit)
5	Parallel Port 2 (LPT2)	Yes (8-bit)
6	Floppy Disk Controller	Yes (8-bit)
7	Parallel Port 1 (LPT1)	Yes (8-bit)
8	Real-Time Clock	No
9	Available (appears as IRQ 2)	Yes (8-bit)
10	Available	Yes (16-bit)
11	Available	Yes (16-bit)
12	Motherboard Mouse Port	Yes (16-bit)
13	Math Coprocessor	No
14	Hard Disk Controller	Yes (16-bit)
15	Available	Yes (16-bit)

Because IRQ 2 now is used directly by the motherboard, the wire for IRQ 9 has been rerouted to the same position in the slot that IRQ 2 normally would occupy. Therefore, any board you install that is set to IRQ 2 actually is using IRQ 9. The interrupt vector table has been adjusted accordingly to enable this deception to work. This adjustment to the system provides greater compatibility with the PC interrupt structure and enables cards that are set to IRQ 2 to work properly.

Notice that interrupts 0, 1, 2, 8, and 13 are not on the bus connectors and are not accessible to adapter cards. Interrupts 8, 10, 11, 12, 13, 14, and 15 are from the second interrupt controller and are accessible only by boards that use the 16-bit extension connector,

because this is where these wires are located. IRQ 9 is rewired to the 8-bit slot connector in place of IRQ 2, which means that IRQ 9 replaces IRQ 2 and therefore is available to 8-bit cards, which treat it as though it were IRQ 2.

Note

Although the 16-bit ISA bus has twice as many interrupts as systems that have the 8-bit ISA bus, you still may run out of available interrupts, because only 16-bit adapters can use any of the new interrupts.

The extra IRQ lines in a 16-bit ISA system are of little help unless the adapter boards that you plan to use enable you to configure them for one of the unused IRQs. Some devices are hard-wired so that they can use only a particular IRQ. If you have a device that already uses that IRQ, you must resolve the conflict before installing the second adapter. If neither adapter enables you to reconfigure its IRQ use, chances are that you cannot use the two devices in the same system.

IRQ Conflicts. One of the most common areas of IRQ conflict involves serial (COM) ports. You may have noticed in the preceding two sections that two IRQs are set aside for two COM ports. IRQ 3 is used for COM2, and IRQ 4 is used for COM1. The problem occurs when you have more than two serial ports in a system—a situation that is entirely possible, because a PC can support up to four COM ports.

The solution used by the PC—assigning IRQs among different COM ports—historically has been unsatisfactory. In reality, IRQ 3 is used for *even* COM ports, and IRQ 4 is used for *odd* COM ports. Therefore, COM2 and COM4 both share IRQ 3, and COM1 and COM3 both share IRQ 4. As a result, you cannot use COM1 at the same time as COM3; neither can you use COM2 at the same time as COM4. Why? If more than one device shares the same IRQ, neither one can signal the CPU to indicate that it needs attention.

When you install devices that use COM ports, you must ensure that no two devices will be using the same IRQ at the same time. This issue can be tricky. If a mouse is configured for COM3 (and IRQ 4), for example, you cannot use COM1 (which also uses IRQ 4) for another device, such as a modem. In this example, the mouse driver, when it is installed in memory from CONFIG.SYS or AUTOEXEC.BAT, takes control of the IRQ for which it is configured. If another device attempts to use the IRQ that the mouse uses, the resulting conflict could destroy data or lock up your system.

DMA Channels

DMA (*direct memory access*) channels are used by high-speed communications devices that must send and receive information at high speed. A serial or parallel port does not use a DMA channel, but a network adapter often does. DMA channels sometimes can be shared if the devices are not of the type that would need them simultaneously. For example, you can have a network adapter and a tape backup adapter sharing DMA channel 1, but you cannot back up while the network is running. To back up during network operation, you must ensure that each adapter uses a unique DMA channel.

8-Bit ISA Bus DMA Channels. In the 8-bit ISA bus, four DMA channels support high-speed data transfers between I/O devices and memory. Three of the channels are available to the expansion slots. Table 5.18 shows the typical uses of these DMA channels.

Table 5.18 8-Bit ISA Default DMA-Channel Assignments

DMA	Standard Function	Bus Slot
0	Dynamic RAM Refresh	No
1	Available	Yes (8-bit)
2	Floppy disk controller	Yes (8-bit)
3	Hard disk controller	Yes (8-bit)

Because most systems typically have both a floppy and hard disk drive, only one DMA channel is available in 8-bit ISA systems.

16-Bit ISA DMA Channels. Since the introduction of the 80286 CPU, the ISA bus has supported eight DMA channels, with seven channels available to the expansion slots. Like the expanded IRQ lines described earlier in this chapter, the added DMA channels were created by cascading a second DMA controller to the first one. DMA channel 4 is used to cascade channels 0 through 3 to the microprocessor. Channels 0 through 3 are available for 8-bit transfers, and channels 5 through 7 are for 16-bit transfers only. Table 5.19 shows the typical uses for the DMA channels.

Table 5.19 16-Bit ISA, EISA, and MCA Default DMA-Channel Assignments

DMA	Standard Function	Bus Slot
0	Available	Yes (8-bit)
1	Available	Yes (8-bit)
2	Floppy disk controller	Yes (8-bit)
3	Available	Yes (8-bit)
4	First DMA controller	No
5	Available	Yes (16-bit)
6	Available	Yes (16-bit)
7	Available	Yes (16-bit)

EISA. Realizing the shortcomings inherent in the way DMA channels are implemented in the ISA bus, the creators of the EISA specification created a specific DMA controller for their new bus. They increased the number of address lines to include the entire address bus, thus allowing transfers anywhere within the address space of the system. Each DMA channel can be set to run either 8-, 16-, or 32-bit transfers. In addition, each DMA channel can be separately programmed to run any of four types of bus cycles when transferring data:

- *Compatible*. This transfer method is included to match the same DMA timing as used in the ISA bus. This is done for compatibility reasons; all ISA cards can operate in an EISA system in this transfer mode.
- *Type A*. This transfer type compresses the DMA timing by 25 percent over the compatible method. It was designed to run with most (but not all) ISA cards and still yield a speed increase.
- *Type B*. This transfer type compresses timing by 50 percent over the compatible method. Using this method, most EISA cards function properly, but only a few ISA cards will be problem-free.
- *Type C*. This transfer method compresses timing by 87.5 percent over the compatible method; it is the fastest DMA transfer method available under the EISA specification. No ISA cards will work using this transfer method.

EISA DMA also allows for special reading and writing operations referred to as *scatter write* and *gather read*. Scattered writes are done by reading a contiguous block of data and writing it to more than one area of memory at the same time. Gathered reads involve reading from more than one place in memory and writing to a device. These functions are often referred to as *buffered chaining* and they increase the throughput of DMA operations.

MCA. You might assume that, because MCA is a complete rebuilding of the PC Bus structure, DMA in an MCA environment would be better constructed; this is not so. In the MCA specification, systems were designed around one DMA controller with the following issues:

- The DMA controller can only connect to two 8-bit data paths. Thus, it can only transfer one or two bytes per bus cycle.
- The DMA controller is connected to address lines 0 through 23 on the address bus. This means the DMA controller can only address up to 16M of memory.
- The DMA controller runs at 10 MHz.

The inability of the DMA controller to address more than two bytes per transfer severely cripples this otherwise powerful bus. Perhaps this is a part of the MCA specification that will be modified in future versions.

Memory

All adapters, to one extent or another, use memory. The type of memory that an adapter uses depends on the type of adapter card and what it is designed to accomplish. Some cards (such as video cards) consume part of your system memory as video buffers; other cards require memory in the ROM portion of your system. The following sections discuss these types of memory use.

System-Memory Use by Video Cards. A video adapter installed in your system uses some of your system's memory to hold graphics or character information for display.

Some adapters, including the VGA, also have on-board BIOS mapped into the system space reserved for such types of adapters (discussed in the following section). Generally, the higher the resolution and color capability of the video adapter, the more system memory that the video adapter uses. Most VGA or Super VGA adapters have additional on-board memory that is used to handle the information that is currently displayed on-screen and to speed screen refresh.

In the standard system-memory map, a total 128K is reserved for use by the video card to store currently displayed information. The reserved video memory is located in segments A000 and B000; each segment is 64K. The video adapter ROM uses additional upper-memory space in segment C000. The following list describes how each type of video adapter affects your use of memory:

- *Monochrome Display Adapter (MDA).* This adapter uses only a 4K portion of the reserved video RAM from B0000 through B0FFF. Because the ROM code that is used to operate this adapter actually is a portion of the motherboard ROM, no additional ROM space is used in segment C000.
- *Color Graphics Adapter (CGA).* The CGA uses a 16K portion of the reserved video RAM from B8000 through BBFFF. Because the ROM code that is used to operate this adapter is a portion of the motherboard ROM, no additional ROM space is used in segment C000.
- *Enhanced Graphics Adapter (EGA).* This adapter uses all 128K of the video RAM from A0000 through BFFFF. The ROM code that is used to operate this adapter is on the adapter itself and consumes 16K of memory from C0000 through C3FFF.
- *Video Graphics Array (VGA).* Standard VGA cards use all 128K of the video RAM from A0000 through BFFFF, as well as 32K of ROM space from C0000 through C7FFF.
- *PS/2 Video Adapter.* Certain models of the PS/2 incorporate a VGA adapter. This adapter uses all 128K of video RAM from A0000 through BFFFF. The ROM code that operates this adapter is on the adapter itself and consumes 24K of memory from C0000 through C5FFF.

Notice that the IBM PS/2 Display Adapter uses additional memory, called *scratch-pad memory*. This memory use was not documented clearly in the technical-reference information for the adapter. In particular, the 2K of memory used at CA000 can cause problems with other cards if they are addressed in this area. If you try to install a hard disk controller with a 16K BIOS at C8000, for example, the system locks up during bootup because of the conflict with the video-card memory. You can solve the problem by altering the start address of the disk controller BIOS to D8000.

In PS/2 systems that have the Video Graphics Array (VGA) or MultiColor Graphics Array (MCGA), the built-in display systems also use all 128K of the reserved video RAM space. Because these display systems are built into the motherboard, however, the control BIOS code is built into the motherboard ROM BIOS and needs no space in segment C000.

Using Video Memory

Some programs, such as memory managers, may play tricks with your system, such as trying to move drivers and other system software into memory that typically is used by the video adapter. Whether this situation causes problems depends on how heavily you tax your video card. If you use the high-resolution video modes available with many of the modern video cards, no memory in the video area may be left for anything else. Make sure that you know how your video adapter will be using memory before you arbitrarily decide that your memory manager can have it.

ROM BIOS Memory Areas. Many types of adapter cards use memory areas for their ROM BIOS controllers. These areas are necessary for programs that control the way that the adapter does its work. Typically, these areas begin at a segment address somewhere above C000.

The amount of memory consumed by an adapter's ROM is entirely up to the adapter. For this reason, you could install in your system two adapters that attempt to use the same ROM areas. In such a case, you need to change the address used by one of the cards (many cards now permit you to do this) or remove one of the cards.

Table 5.20 describes the typical ROM addresses used by various types of adapters. (You should not assume that this table lists all the possible addresses, however; it always seems that one more board out there uses a different area.)

Table 5.20 Default Memory Use by Various Adapter Boards

Adapter	Starting Segment	Amount Used
IBM EGA ROM	C000	16K
PS/2 VGA ROM	C000	24K
PS/2 VGA scratch RAM	CA00	24K
Most EGA/VGA ROMs	C000	32K
IBM XT 10M hard disk controller ROM	C800	8K
IBM XT 20M hard disk controller ROM	C800	4K
Most other hard disk controller ROMs	C800	16K
Token Ring network adapter ROM	CC00	8K
Token Ring network adapter RAM	D800	16K
LIM expanded memory adapter RAM	D000	64K

Notice that some of the listed adapter boards would use the same memory addresses as other boards, which is not allowed in a single system. If two adapters have overlapping ROM or RAM addresses, neither board operates properly. Each board functions if you remove or disable the other one, but the boards do not work together.

On many adapter boards, you can use jumpers, switches, or driver software to change the actual memory locations to be used. This procedure may be necessary to make the two boards coexist in one system.

A conflict of this type can cause problems for troubleshooters. If you install a Token Ring network card and an Intel Above Board (an EMS board) in the same system, and then use the factory-set default configuration with each one, neither adapter card works. You must read the documentation for each adapter to find out what memory addresses the adapter uses and how to change the addresses to allow coexistence with another adapter. Most of the time, you can work around these problems by reconfiguring the board or by changing jumpers, switching settings, or software-driver parameters. For the Token Ring card, you can change the software-driver parameters in the CONFIG.SYS file to move the memory that the card uses from D8000 to something closer to C4000. This change enables the two boards to coexist and stay out of each other's way.

Resolving Resource Conflicts

The resources in a system are limited. Unfortunately, the demands on those resources seem to be unlimited. As you add more and more adapter cards to your system, you will find that the potential for resource conflicts increases. If your system does not have a bus that resolves conflicts for you (such as an MCA or EISA bus), you need to resolve the conflicts manually.

How do you know whether you have a resource conflict? Typically, one of the devices in your system stops working. Resource conflicts can exhibit themselves in other ways, though. Any of the following events could be diagnosed as a resource conflict:

- A device transfers data inaccurately.
- Your system frequently locks up.
- Your sound card doesn't sound quite right.
- Your mouse doesn't work.
- Garbage appears on your video screen for no apparent reason.
- Your printer prints gibberish.
- You cannot format a floppy disk.

In the following sections, you learn some of the steps that you can take to head off resource conflicts or to track them down when they occur.

Caution

Be careful in diagnosing resource conflicts; a problem may not be a resource conflict at all, but a computer virus. Many computer viruses are designed to exhibit themselves as glitches or as periodic problems. If you suspect a resource conflict, it may be worthwhile to run a virus check first to ensure that the system is clean. This procedure could save you hours of work and frustration.

Resolving Conflicts Manually

Unfortunately, the only way to resolve conflicts manually is to take the cover off your system and start changing switches or jumper settings on your adapter cards. Each of these changes then must be accompanied by a system reboot, which implies that they take a great deal of time. This situation brings us to the first rule of resolving conflicts: when you set about ridding your system of resource conflicts, make sure that you allow a good deal of uninterrupted time.

Also make sure that you write down your current system settings before you start making changes. That way, you will know where you began and can go back to the original configuration (if necessary).

Finally, dig out the manuals for all your adapter boards; you will need them. If you cannot find the manuals, contact the manufacturers to determine what the various jumper positions and switch settings mean.

Now you are ready to begin your detective work. As you try various switch settings and jumper positions, keep the following questions in mind; the answers will help you narrow down the conflict areas.

- *When did the conflict first become apparent?* If the conflict occurred after you installed a new adapter card, that new card probably is causing the conflict. If the conflict occurred after you started using new software, chances are good that the software uses a device that is taxing your system's resources in a new way.
- *Are there two similar devices in your system that do not work?* If your modem and mouse do not work, for example, chances are good that these devices are conflicting with each other.
- *Have other people had the same problem, and if so, how did they resolve it?* Public forums—such as those on CompuServe, Internet, and America Online—are great places to find other users who may be able to help you solve the conflict.

Whenever you make changes in your system, reboot and see whether the problem persists. When you believe that you have solved the problem, make sure that you test all your software. Fixing one problem often seems to cause another to crop up. The only way to make sure that all problems are resolved is to test everything in your system.

As you attempt to resolve your resource conflicts, you should work with and update a system-configuration template, as discussed in the following section.

Using a System-Configuration Template

A system-configuration template is helpful, simply because it is easier to remember something that is written down than it is to keep it in your head. To create a configuration template, all you need to do is start writing down what resources are used by which parts of your system. Then, when you need to make a change or add an adapter, you can quickly determine where conflicts may arise.

To create a system-configuration template, mark eight columns on a piece of paper, and label each column as follows:

- Subsystem
- Use
- Slot
- IRQ Line
- I/O Address
- DMA Channel
- Memory Used
- Comments

Each row in the template represents a different system component. You definitely want to include standard components, such as video, COM1, COM2, LPT1, floppy controller, and hard drive controller. Don't forget to include other common components, such as tape drive, CD-ROM, scanner, SCSI port, and sound card.

Next, fill in each column of the template to reflect which resources are used by that component. Figure 5.19 shows an example of a filled-in system-configuration template. Remember that because every system is different, your system-configuration template probably will differ from the one in this example.

SYSTEM CONFIGURATION TEMPLATE

SYSTEM: JOHN SMITH
DATE LAST CHANGED: 10/4/94

SUBSYSTEM	USE	SLOT	IRQ LINE	I/O ADDRESS	DMA CHANNEL	MEMORY USED	COMMENTS
VIDEO	SUPER VGA	1				A0000 - BFFFF	
COM1		BUILT-IN	4	03F8			
COM2	MOUSE	BUILT-IN	3	02F8			
COM3	MODEM	4	4	03E8			INTERNAL MODEM CARD
COM4							
LPT1	LASER PRINTER	BUILT-IN	7	03BC			IRQ CONFLICT OK; NEVER USE BOTH AT ONCE
LPT2	DOT-MATRIX	BUILT-IN	7	0378			IRQ CONFLICT OK; NEVER USE BOTH AT ONCE
LPT3							
FLOPPY DRIVE	FLOPPY DRIVE	2	6	03F0 - 03F7	2	C8000	SAME CARD AS HARD DRIVE CONTROLLER
HARD DRIVE	HARD DRIVE	2	14	01F0 - 01F7		C8000	SAME CARD AS FLOPPY DRIVE CONTROLLER
SCSI PORT	CD-ROM, SCANNER	6	11	0330 - 0333	5	DC000	
SOUND CARD	SOUND BLASTER	7	5	0220	1		CAN'T CHANGE I/O ADDRESS
NETWORK ADAPTER	ETHERNET	5	10	02A0 - 02BF			MUST RECOMPILE DRIVERS IF SPECS CHANGED

Fig. 5.19
A sample system-configuration template.

Heading Off Problems: Special Boards

A number of devices that you may want to install in a computer system require IRQ lines or DMA channels, which means that a world of conflict could be waiting in the box that the device comes in. As mentioned in the preceding section, you can save yourself problems if you use a system-configuration template to keep track of the way that your system is configured.

You also can save yourself trouble by carefully reading the documentation for a new adapter board *before* you attempt to install it. The documentation details the IRQ lines that the board can use, as well as its DMA-channel requirements. In addition, the documentation will detail the adapter's upper-memory needs for ROM and adapter RAM.

The following sections describe some of the conflicts that you may encounter when you install today's most popular adapter boards. Although the list of adapter boards covered in these sections is far from comprehensive, the sections serve as a guide to installing complex hardware with minimum hassle. Included are tips on sound boards, SCSI host adapters, and network adapters.

Sound Boards. Most sound boards require two kinds of hardware communication channels: one or more IRQ lines, and exclusive access to a DMA channel. If you take the time to read your sound board's documentation and determine its communications-channel needs, compare those needs to the IRQ lines and DMA channels that already are in use in your system, and then set the jumpers or switches on the sound board to configure it for available channels, your installation will go quickly and smoothly.

One example of a potential sound-board conflict is the Pro Audio Spectrum 16 sound card. This card needs access to as many as two DMA channels—one for use by its own circuitry, and the other for SoundBlaster compatibility—and as many as two IRQ lines. Before installing this board, you must determine not only how you plan to use the board (thereby determining the IRQ lines and DMA channels that you will need), but also the available IRQ lines and DMA channels that the board can be configured to use.

The Pro Audio Spectrum 16 sound card is not singled out here because there is something unusual or particularly complex about the card; many high-end sound cards require a great deal of system resources.

SCSI Adapter Boards. SCSI adapter boards, like other advanced add-in devices for modern PCs, require considerable system resources. A SCSI host adapter board, for example, may require an IRQ line, a DMA channel, or a large chunk of unused upper memory for its ROM and RAM use.

Before installing a SCSI adapter, be sure to read the documentation for the card, and make sure that any IRQ lines, DMA channels, and upper memory that the card needs are available. If the card needs system resources that are already in use, use your system-configuration template to determine how you can free the needed resources before you attempt to plug in the adapter card. In addition, remember to set the jumpers or switches detailed in the card's documentation, and run any setup software that the device requires.

Multiple-COM-Port Adapters. Because COM ports are required for so many peripherals that connect to the modern PC, and because the number of COM ports that can be used is strictly limited by the IRQ setup in the basic IBM system design, special COM-port cards are available that enable you to assign a unique IRQ to each of the four COM ports on the card. For example, you can use such a card to leave COM1 and COM2 configured for IRQ 4 and IRQ 3, respectively, but to configure COM3 for IRQ 10 and COM4 for IRQ 11.

With a multiple-COM-port adapter card installed and properly configured for your system, you can have devices hooked to four COM ports, and all four devices can be functioning at the same time. For example, you can use a mouse, modem, plotter, and serial printer at the same time.

The Future: Plug-and-Play Systems

What does the future hold? Perhaps the most exciting development looming on the horizon is plug-and-play systems. The specifications for these systems are available, and the systems themselves should start becoming available late in 1994 or early in 1995. Specifications exist for ISA, PCI, SCSI, IDE CD-ROM, MCA, and PCMCIA systems and components.

Most users of MCA or EISA systems already know what *plug-and-play* means: it means that your computer senses any new adapter card on the I/O bus and makes the necessary adjustments to share resources without conflict. You never need to worry about I/O addresses, DMA channels, or IRQ settings.

As mentioned earlier, this sort of capability has been built into EISA and MCA buses for some time; the problem is that these are not the only buses on the market. Other buses (all of which are described earlier in this chapter) enjoy the lion's share of the PC market. As a result, most computer users must use the trial-and-error method of making their systems work properly. The plug-and-play specifications already in existence have perhaps the greatest impact on users of systems that have ISA buses. Their implementation promises a great boon for most PC users.

Plug-and-play is not only a hardware issue. For plug-and-play to work, three components are required:

- Hardware
- BIOS
- Operating system

Each of these components needs to be plug-and-play-compatible, meaning that it complies with the plug-and-play specifications.

The Hardware Component. The *hardware component* refers to both computer systems and adapter cards. The term does not mean, however, that you will not be able to use your older ISA adapter cards (referred to as *legacy cards*) in a plug-and-play system. You can use these cards; you just won't receive the benefits of automatic configuration, as you would if the cards were compatible.

Plug-and-play adapter cards will be able to communicate with the system BIOS and the operating system to convey information about what system resources are needed. The BIOS and operating system, in turn, will resolve conflicts (wherever possible) and inform the adapter cards which specific resources it should use. The adapter card then can modify its configuration to use the specified resources.

The BIOS Component. The *BIOS component* means that most users of existing PCs will need to update their BIOSs or purchase new machines that have plug-and-play BIOSs. For a BIOS to be compatible, it must support 13 additional system function calls, which can be used by the operating-system component of a plug-and-play system. The BIOS specification developed for plug-and-play was developed jointly by Compaq, Intel, and Phoenix Technologies. The latest version is 1.0A, dated May 5, 1994.

The plug-and-play features of the BIOS are implemented through an expanded power-on self-test (POST). The BIOS is responsible for identification, isolation, and possible configuration of plug-and-play adapter cards. The BIOS accomplishes these tasks by performing the following steps:

1. Disables any configurable devices on the motherboard or on adapter cards.
2. Identifies any plug-and-play ISA devices.
3. Compiles an initial resource-allocation map for ports, IRQs, DMAs, and memory.
4. Enables I/O devices.
5. Scans the ROMs of ISA devices.
6. Configures initial-program-load (IPL) devices, which are used later to boot the system.
7. Enables configurable devices by informing them which resources have been assigned to them.
8. Starts the bootstrap loader.
9. Transfers control to the operating system.

The Operating-System Component. The *operating-system component* can be implemented by a brand-new version (such as an update of DOS or Windows, or the upcoming Chicago version of Windows) or through DOS extensions. Extensions of this type should be familiar to most DOS users; extensions have been used for years to provide support for CD-ROM drives. Some sort of extensions are expected to be available before the end of 1994 to ensure plug-and-play compatibility.

It is the responsibility of the operating system to inform users of conflicts that cannot be resolved by the BIOS. Depending on the sophistication of the operating system, the user then can configure the offending cards manually (on-screen) or turn the system off and set switches on the physical cards. When the system is restarted, the system is checked for remaining (or new) conflicts, any of which are brought to the user's attention. Through this repetitive process, all system conflicts are resolved.

Summary

This chapter covered a great deal of ground, going from the general to the specific and from the conceptual to the actual. The information in this chapter enables you to understand how your system works, physically and logically. You learned the following:

- How buses are used in a computer system
- What a processor bus, memory bus, address bus, and I/O bus are
- How expansion slots fit into the architecture of your system
- How the major types of I/O buses (ISA, MCA, EISA, local bus, VL-Bus, and PCI) compare
- That system resources (including I/O addresses, IRQ lines, DMA channels, and memory areas) are limited
- How adapter cards rely on system resources to communicate with your PC and to do their work
- That system-resource conflicts can stop your adapter cards or your entire system from functioning properly
- That the MCA bus, the EISA bus, and (in the future) plug-and-play greatly simplify allocation of system resources
- That if your system does not resolve system-resource conflicts automatically, you must resolve them manually