How PCI Works

by Jeff Tyson

There are a lot of incredibly complex components in a computer. And all of these parts need to communicate with each other in a fast and efficient manner. Otherwise, the amazing speed and capabilities of each individual component is lost in the whole.

A typical PCI card

That's where the **bus** comes in. Essentially, a bus is the channel or path between the components in a computer. There are many different types of buses. In this edition of **HowStuffWorks**, you will learn about some of those buses. We will concentrate on the bus known as the **Peripheral Component Interconnect** (PCI). We'll talk about what PCI is, how it operates and how it is used, as well as about past and future bus technologies.

Hop on the Bus, Gus

Buses have grown and evolved over the years in an effort to match the performance of all the other computer components. Even so, the evolution of the bus has been surprisingly slow compared to other technologies. Most computers sold today still have an **Industry Standard Architecture** (ISA) bus that will accept computer cards developed for the original IBM PC in the early 1980s.

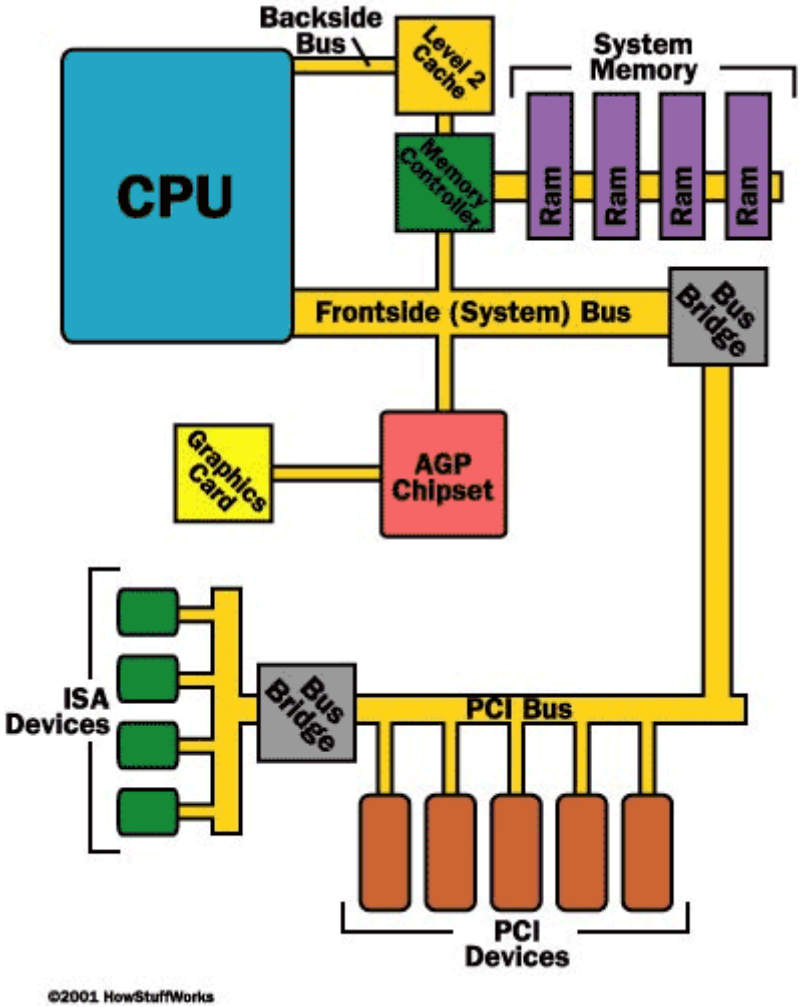
There have been a couple of key reasons for this bus longevity:

- There is a need for long-term compatibility with a large number of hardware manufacturers.
- Before the rise of multimedia, few hardware peripherals fully utilized the speed of the bus.

A typical computer has two key buses. The first one, known as the **system bus** or **local bus**, connects the microprocessor (central processing unit) and the system memory. Other buses, such as the ISA and PCI buses, connect to the system bus through a **bridge**, which is a part of the computer's chipset and acts as a traffic cop, integrating the data from the other buses to the system bus.

As the speed of central processing units (CPUs) and RAM increased, it became more important to isolate the path between processor and memory. A replacement for the standard system bus, called **Dual Independent Bus** (DIB), was created. DIB replaced the single system bus with a **frontside bus** and a **backside bus**. The backside bus had one purpose: to provide a direct, fast

channel between the CPU and the Level 2 cache. The frontside bus connected the system memory, via the memory controller, to the CPU, and the other buses to the CPU and system memory.



The illustration above shows how the various buses connect to the CPU.

The other main bus, the **shared bus**, is for connecting additional components to the computer. It is called a shared bus because it lets multiple devices access the same path to the CPU and system memory. These devices includes such items such as:

- Modem
- Hard drive
- Sound card
- Graphics card
- Controller card
- Scanner

The original PC bus operated at 4.77 MHz (million cycles per second) and was 8 bits wide, meaning it could process 8 bits of data in each cycle. In 1982, it improved to 16 bits at 8 MHz and officially became known as ISA. This bus design is capable of passing along data at a rate of up to 16 MBps (megabytes per second), fast enough even for many of today's applications.

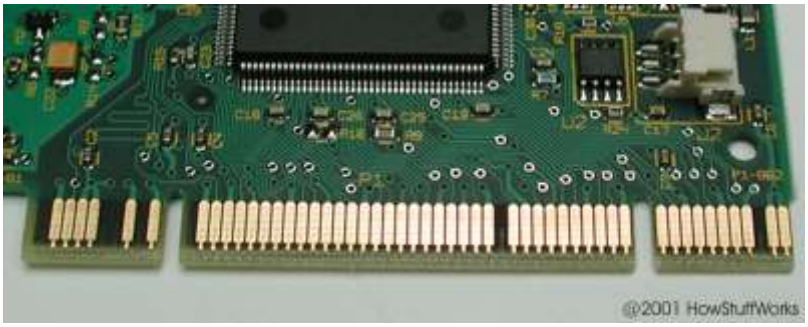
As technology advanced and the ISA bus grew long in the tooth, other buses were developed. Key among these were **Extended Industry Standard Architecture** (EISA) -- which was 32 bits

at 8 MHz-- and **Vesa Local Bus** (VL-Bus). The cool thing about VL-Bus (named after VESA, the Video Electronics Standards Association, which created the standard) is that it was 32 bits wide and operated at the speed of the local bus, which was normally the speed of the processor itself. The VL-Bus essentially tied directly into the CPU. This worked okay for a single device, or maybe even two. But connecting more than two devices to the VL-Bus introduced the possibility of interference with the performance of the CPU. Because of this, the VL-Bus was typically used only for connecting a graphics card, a component that really benefits from high-speed access to the CPU.

Along Comes PCI

During the early 1990s, Intel introduced a new bus standard for consideration, the **Peripheral Component Interconnect (PCI)**. PCI presents a hybrid of sorts between ISA and VL-Bus. It provides direct access to system memory for connected devices, but uses a bridge to connect to the frontside bus and therefore to the CPU. Basically, this means that it is capable of even higher performance than VL-Bus while eliminating the potential for interference with the CPU.

PCI can connect more devices than VL-Bus, up to five external components. Each of the five connectors for an external component can be replaced with two fixed devices on the motherboard. Also, you can have more than one PCI bus on the same computer, although this is rarely done. The PCI bridge chip regulates the speed of the PCI bus independently of the CPU's speed. This provides a higher degree of reliability and ensures that PCI-hardware manufacturers know exactly what to design for.



PCI cards use 47 pins.

PCI originally operated at 33 MHz using a 32-bit-wide path. Revisions to the standard include increasing the speed from 33 MHz to 66 MHz and doubling the bit count to 64. Currently, PCI-X provides for 64-bit transfers at a speed of 133 MHz for an amazing 1-GBps (gigabyte per second) transfer rate!

PCI cards use 47 pins to connect (49 pins for a **mastering** card, which can control the PCI bus without CPU intervention). The PCI bus is able to work with so few pins because of hardware **multiplexing**, which means that the device sends more than one signal over a single pin. Also, PCI supports devices that use either 5 volts or 3.3 volts.

Bus Type	Bus Width	Bus Speed	MB/sec
ISA	16 bits	8 MHz	16 MBps
EISA	32 bits	8 MHz	32 MBps
VL-bus	32 bits	25 MHz	100 MBps
VL-bus	32 bits	33 MHz	132 MBps
PCI	32 bits	33 MHz	132 MBps
PCI	64 bits	33 MHz	264 MBps

PCI	64 bits	66 MHz	512 MBps
PCI	64 bits	133 MHz	1 GBps

Although Intel proposed the PCI standard in 1991, it did not achieve popularity until the arrival of Windows 95 (in 1995). This sudden interest in PCI was due to the fact that Windows 95 supported a feature called **Plug and Play** (PnP), which we'll talk about in the next section.

Plug and Play

Plug and Play (PnP) means that you can connect a device or insert a card into your computer and it is automatically recognized and configured to work in your system. PnP is a simple concept, but it took a concerted effort on the part of the computer industry to make it happen. Intel created the PnP standard and incorporated it into the design for PCI. But it wasn't until several years later that a mainstream operating system, Windows 95, provided system-level support for PnP. The introduction of PnP accelerated the demand for computers with PCI, very quickly supplanting ISA as the bus of choice.

To fully implement PnP requires three things:

- **PnP BIOS** - The core utility that enables PnP and detects PnP devices. The BIOS also reads the ESCD for configuration information on existing PnP devices.
- **Extended System Configuration Data** (ECSD) - A file that contains information about installed PnP devices.
- **PnP operating system** - Any operating system, such as Windows 95/98/ME, that supports PnP. PnP handlers in the operating system complete the configuration process started by the BIOS for each PnP device.

PnP automates several key tasks that were typically done either manually or with an installation utility provided by the hardware manufacturer. These tasks include the setting of:

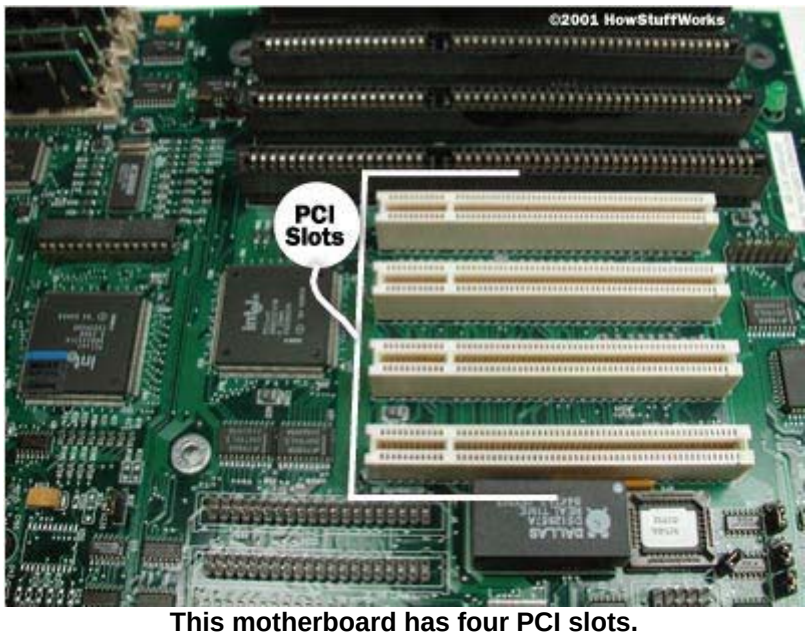
- **Interrupt requests** (IRQ) - An IRQ, also known as a hardware interrupt, is used by the various parts of a computer to get the attention of the CPU. For example, the mouse sends an IRQ every time it is moved to let the CPU know that it's doing something. Before PCI, every hardware component needed a separate IRQ setting. But PCI manages hardware interrupts at the bus bridge, allowing it to use a single system IRQ for multiple PCI devices.
- **Direct memory access** (DMA) - This simply means that the device is configured to access system memory without consulting the CPU first.
- **Memory addresses** - Many devices are assigned a section of system memory for exclusive use by that device. This ensures that the hardware will have the needed resources to operate properly.
- **Input/Output (I/O) configuration** - This setting defines the ports used by the device for receiving and sending information.

While PnP makes it much easier to add devices to your computer, it is not infallible. Variations in the software routines used by PnP BIOS developers, PCI device manufacturers and Microsoft have led many to refer to PnP as "Plug and *Pray*." But the overall effect of PnP has been to greatly simplify the process of upgrading your computer to add new devices or replace existing ones.

How It Works

Let's say that you have just added a new PCI-based sound card to your Windows 98 computer. Here's an example of how it would work.

1. You open up your computer's case and plug the sound card into an empty PCI slot on the motherboard.



This motherboard has four PCI slots.

2. You close the computer's case and power up the computer.
3. The system BIOS initiates the PnP BIOS.
4. The PnP BIOS scans the PCI bus for hardware. It does this by sending out a signal to any device connected to the bus, asking the device who it is.
5. The sound card responds by identifying itself. The device ID is sent back across the bus to the BIOS.
6. The PnP BIOS checks the ESCD to see if the configuration data for the sound card is already present. Since the sound card was just installed, there is no existing ESCD record for it.
7. The PnP BIOS assigns IRQ, DMA, memory address and I/O settings to the sound card and saves the data in the ESCD.
8. Windows 98 boots up. It checks the ESCD and the PCI bus. The operating system detects that the sound card is a new device and displays a small window telling you that Windows has found new hardware and is determining what it is.
9. If it is able to determine what the device is, it displays the name of the device and attempts to install the **driver** (the software that enables the device to communicate with the operating system). You may be asked to insert a disk with the driver on it or tell Windows where to find the driver software. If Windows cannot determine what the device is, it provides a dialog window so that you can specify what type of device it is and load a driver to run it.
10. Once the driver is installed, the device should be ready for use. Some devices may require that you restart the computer before you can use them. In our example, the sound card is immediately ready for use.
11. You want to capture some audio from an external tape deck that you have plugged into the sound card. You set up the recording software that came with the sound card and begin to record.
12. The audio comes into the sound card via an external audio connector. The sound card converts the analog signal to a digital signal.
13. The digital audio data from the sound card is carried across the PCI bus to the bus controller. The controller determines which device on the PCI device has priority to send data to the CPU. It also checks to see if data is going directly to the CPU or to system memory.
14. Since the sound card is in record mode, the bus controller assigns a high priority to the

data coming from it and sends the sound card's data over the bus bridge to the system bus.

15. The system bus saves the data in system memory. Once the recording is complete, you can decide whether the data from the sound card is saved to a hard drive or retained in memory for additional processing.

What exactly does the future hold for PCI and PnP? Read on to see what's just around the corner.

All Aboard

As processor speeds steadily climb in the GHz range, many companies are working feverishly to develop a next-generation bus standard. Many feel that PCI, like ISA before it, is fast approaching the upper limit of what it can do.

All of the proposed new standards have something in common. They propose doing away with the shared-bus technology used in PCI and moving to a **point-to-point switching connection**. This means that a direct connection between two devices (nodes) on the bus is established while they are communicating with each other. Basically, while these two nodes are talking, no other device can access that path. By providing multiple direct links, such a bus can allow several devices to communicate with no chance of slowing each other down.

The first next-generation standard is already in place. **Infiniband** is a culmination of two previously proposed standards, Intel's **Next Generation I/O** (NGIO) and **Future I/O**, which was developed jointly by IBM, Hewlett Packard and Compaq. Infiniband is capable of a data path between two nodes with transfer rates up to about 4 GBps in the 12-channel version and 1.25 GBps in the four-channel version. Infiniband is already being integrated into enterprise-level servers, but is not expected to become a standard for PCs.

Will PCI be replaced by HyperTransport?

HyperTransport, a standard proposed by Advanced Micro Devices, Inc. (AMD), is touted by AMD as the natural progression from PCI. For each session between nodes, it provides two point-to-point links. Each link can be anywhere from 2 bits to 32 bits wide, supporting a maximum transfer rate of 6.4 GBps. HyperTransport is designed specifically for connecting internal computer components to each other, not for connecting external devices such as removable drives. The development of bridge chips will enable PCI devices to access the HyperTransport bus.

Intel has announced **3GIO** (Third Generation I/O), a proposed standard that industry experts believe will compete with HyperTransport to replace PCI. Intel's plans for 3GIO include:

- Speeds in excess of 10 GHz
- Full serial I/O architecture

- Point-to-point connections
- Low pin count

3GIO is still in the preliminary stages of specification and has a way to go before it can be considered a standard. But the emphasis with 3GIO, as well as with HyperTransport and Infiniband, is to move away from the bus-based system and toward a direct-connection system. Even so, ISA is still in use two decades after its inception, and PCI is expected to hang around for a long time yet.