Local Buses Poised to Enter PC Mainstream

Intel's PCI and VESA's VL-Bus Vie for Vendors' Attention

By Michael Slater and Mark Thorson

Last month, both VESA (the Video Electronics Standards Association) and Intel announced local-bus standards that will have a significant impact on PC system architecture in the coming years. The two standards are complementary in some ways, but they will ultimately be head-on competitors. In this article, we first explore the motivations for local-bus standards. Then, we delve into the details of the two specifications, and take a look at the role each is likely to play.

Local Bus Motivations

The IBM-compatible PC architecture is now over ten years old, and it is showing its age. Nowhere is the standard system architecture more glaringly out-of-date than in the graphics subsystem, whose performance has been hobbled by a slow communication channel to a clumsy frame-buffer interface. The increasing dominance of Windows has fueled a demand for higher-resolution displays, making the bottlenecks of the PC architecture painfully obvious.

There are several factors that limit PC display performance. The widespread VGA standard is one culprit, because it requires that the frame buffer be accessed through a 64-Kbyte window. The most severe limiter, however, is the low bandwidth of the I/O bus. The majority of the PC market still uses the AT (ISA) bus, which is limited to about 5 Mbytes/s—an inadequate rate for updating a high-resolution, bit-mapped display, especially when animation or real-time video is required.

The AT bus, which is limited by compatibility with add-in cards designed for the 1980-era IBM PC, is simply no longer an appropriate interface for today's PCs. The EISA and Micro Channel buses are considerably faster, but the cost of such systems is higher because of the more complex chip sets, more expensive connectors, and less-competitive markets. While performance can certainly be increased by using bus-master graphics boards on these buses, the resulting price premium is more than the mass market will bear.

One way to improve graphics performance is by using a graphics processor or drawing engine on the display card. This allows the host processor to make better use of the limited bus bandwidth, since a single command to the graphics processor can control the manipulation of many pixels.

Using an intelligent graphics card reduces the impact of the slow AT bus, but even with a graphics processor, performance would be even better if there were a faster communication path between the host processor and the graphics processor. Furthermore, the most cost-effective way to increase graphics performance for low-end systems is to eliminate the graphics processor entirely. If the host processor has high-bandwidth access to a linearly-addressed frame buffer, impressive performance can be achieved without any graphics acceleration hardware.

A few examples illustrate the need for a high-bandwidth interface to the frame buffer. For a 1280×1024 display with 24 bits/pixel, a single image requires almost 4 Mbytes. With the frame buffer on an AT bus, it would take nearly a full second to rewrite this image. This makes the system appear very sluggish and makes animation impossible. For another example, consider a window displaying a television-quality (NTSC) image. This requires rewriting a 320×240 image 30 times per second. With 24 bits/pixel, this requires 6.9 Mbytes/s—more than an AT bus can provide.

The key to improving graphics performance is to get the display controller off the slow AT bus and provide it with a high-bandwidth connection to the host processor. The idea is simple: just connect the display controller to the processor's local bus, instead of to the AT bus. This doesn't require any new standards, and existing graphics chips designed to connect to the AT bus can be interfaced to the local bus with glue logic.

Recently, several display controller chips have become available with direct 386/486 local-bus interfaces, reducing or eliminating the glue logic required. Often, the internal architecture of the display controller needs to be modified to take full advantage of the local-bus speed. Some PC chip sets also support local-bus graphics, which essentially means providing the option for the display controller accesses to be mapped to the processor bus instead of the I/O expansion bus.

While standards are not essential for local-bus graphics, they are desirable. Without a standard, chip-set and graphics-chip vendors don't have a well-defined common target for providing a glue-less interface. With each new generation of microprocessor, the CPU's local bus changes, requiring either a new generation of peripheral chips or a different layer of glue logic.

Furthermore, the lack of a standard local-bus connector makes it impossible for a third-party market for local-bus graphics cards to develop. System vendors that provide local-bus graphics have either included the graphics controller on the motherboard, which reduces

Name	Description
RESET#	System reset
LCLK	1X CPU clock (32 kHz–66 MHz)
RDYRTN#	Ready signal returned to the CPU
ID<4..0>	Host CPU identifier code (CPU type and speed)
ADR<31..2>	Address bus
DAT<31..0>	Data bus
BE<3..0>#	Byte enables
M/IO#	Memory vs. I/O cycle definition
W/R#	Write vs. read cycle definition
D/C#	Data vs. code (instruction prefetch) cycle definition
BLAST#	Burst last (next cycle is last cycle of burst)
ADS#	Address/data strobe
WBACK#	Write-back cache status
LEADS#	Local external address/data strobe
LGNT<2..0>#	Local bus grant (slot-specific)
LDEV#	Local device decode (slot-specific, asserted by bus slave to indicate selection)
LRDY#	Local ready (asserted until bus controller asserts RDYRTN#)
LBS16#	Local bus size 16 bits
BRDY#	Burst ready
LKEN#	Local cache enable
LREQ#	Local bus request (slot-specific)
IRQ9	Interrupt request 9

Table 1. VL-Bus signals.

flexibility, or they use a connector of their own design. Graphics board makers that hoped to sell after-market products would have to produce a different card for each vendor's systems.

A local bus is beneficial not only for graphics, but also for any peripheral requiring high bandwidth. The two most likely applications after graphics are disk controllers and network interfaces. High-performance SCSI drives can exceed the bandwidth of the AT bus. Ethernet and Token Ring networks still are far below even the AT bandwidth, but FDDI (at 100 Mbits/s) will require more bandwidth.

One of the first efforts to standardize a local-bus interface was launched by OPTi, a supplier of system-logic chip sets, which proposed using an EISA-type connector in an AT system. The second set of contacts, normally used in an EISA system for the EISA extensions, are used for the local bus instead. By providing one or more slots in a system with this special connector, those slots can be used for standard AT boards or for local-bus boards. This approach doesn't work with EISA or Micro Channel systems, however. About a dozen smaller system suppliers make systems using the OPTi local-bus connector, but it is unlikely to survive as a standard in the face of the VL-Bus and PCI efforts.

An even earlier local-bus effort was launched by S3, when it announced its system-logic chip set products and attempted to establish the ACI (Advanced Component Interconnect) bus as a chip-level standard. Unfortunately, S3 ran into difficulty getting its aggressive chip-set design into production and decided to abandon the ACI effort, along with the chip-set business, to focus on graphics chips.

VESA's VL-Bus

To establish a successful standard, the support of many graphics chip, board, and system makers is needed. One group that has such support is the Video Electronics Standards Association (VESA), an organization sponsored by IBM-compatible video board and graphics chip vendors. VESA formed a committee to work on a local-bus standard in December 1991, and the draft VESA Local Bus (VL-Bus) standard was released at PC Expo last month. The proposed standard must now be ratified by the group's members, a process that is expected to be completed in August.

A VL-Bus device can be resident on the system board or it can plug into a connector. Unlike Intel's PCI bus, the VL-Bus has defined connector and pin assignments. The connector is the type used for the 16-bit Micro Channel bus. It is in-line with the expansion board connectors, which can be AT bus, EISA, or Micro Channel. Mechanical specifications have been defined for boards that plug into both an expansion slot and the VL-Bus simultaneously, or into the VL-Bus alone.

The VL-Bus specification suggests that no more than two devices should be resident on an unbuffered VL-Bus; a buffered VL-Bus should have no more than three devices. An unbuffered VL-Bus is essentially a raw processor bus with the addition of a few signals.

Bus timing is defined from 32 kHz to 66 MHz. Stopping the clock and dynamic changes in clock speed are allowed. At 33 MHz or below, all of the VL-Bus devices may be resident on add-in boards or the motherboard. Above 33 MHz, VL-Bus buffers are not recommended. The specification advises a maximum of one connector in systems running at 40 MHz, and no connectors at 50 MHz or above. Some designers question the wisdom of exposing the CPU's pins directly on an expansion connector, since a malfunctioning or incorrectly installed add-in board could damage the expensive processor chip. Loading is also a concern.

Table 1 shows the VL-Bus signals, which are a close superset of the signals on the 486 processor bus. The VL-Bus adds a set of ID signals, which are sampled at reset to determine the CPU type and clock speed. There is also a slot-specific pair of bus request/grant signals, used by a central arbiter to award control of the bus. The arbitration scheme is left to the designer; only the worst-case latency is defined.

The VL-Bus allows both 16- and 32-bit host processors (as selected by CPU ID). When a 16-bit bus is used, the BHE#, BLE, and A1 signals are translated into the appropriate byte enables. Sixteen-bit peripherals must steer data to the appropriate byte lanes. Devices

designed to the current specification are planned to be upward- and downward-compatible with future 64-bit VL-Bus motherboards and peripherals.

One interrupt line (IRQ9) is provided for the benefit of VL-Bus devices that don't mate with the expansion (AT, EISA, or Micro Channel) slot. DMA transfers using the system's DMA controller cannot be initiated from a VL-Bus slot. The VL-Bus does support bus masters, however, so a VL-Bus device can produce its own DMA cycles.

Figure 1 is a timing diagram showing cycles at different speeds. The faster cycle timing is allowed only at 33 MHz or below. Above 33 MHz, extra clock periods are inserted to allow for decoding and synchronization.

The cycle begins with the assertion of ADS# and a valid address. For cycles from the system DMA controller, LADS# is delayed by one clock.

The slave then asserts LDEV# within 20 ns to indicate address selection. Above 33 MHz, LDEV# is sampled on the second clock edge after sampling ADS# asserted. At 33 MHz or below, LDEV# may be sampled as soon as the clock edge immediately following the assertion of ADS#. If LDEV# is not asserted, the bus cycle defaults to the expansion bus.

When the slave is ready for the cycle to complete, it asserts LRDY#. Above 33 MHz, this results in the assertion of RDYRTN# (i.e., the ready signal returned to the CPU) after one clock period of synchronization delay through the bus controller. At 33 MHz or below, the bus controller may assert RDYRTN# immediately (after a small gate delay).

Up to 33 MHz, cycles can run with zero wait states on reads and one wait state on writes. From 40 to 66 MHz, reads and writes both require two wait states. Inside of a burst, data transfers can occur at the rate of one per clock period.

The VL-Bus specification supports four-word bursts using the 486-style addressing sequence. A VL-Bus slave device indicates that it supports bursts by asserting BRDY# instead of LRDY#.

A VL-Bus master takes control of the bus by asserting HOLD to the host CPU. Thus, the host processor is halted whenever a VL-Bus master is in control. Concurrent operation is not supported; when a VL-Bus master is accessing a VL-Bus slave, the host CPU cannot access main memory or an I/O bus device.

VESA is developing a "mezzanine" version of the VL-Bus that will be limited to 33 MHz but will support up to 10 devices. (The term mezzanine is used to imply that the bus is one step removed from the processor's local bus.) The mezzanine version will be compatible with the standard VL-Bus from the add-in card's perspective, so there will be a single standard for third-party boards. The mezzanine bus will allow concurrent operation of a VL-Bus master and the host processor.

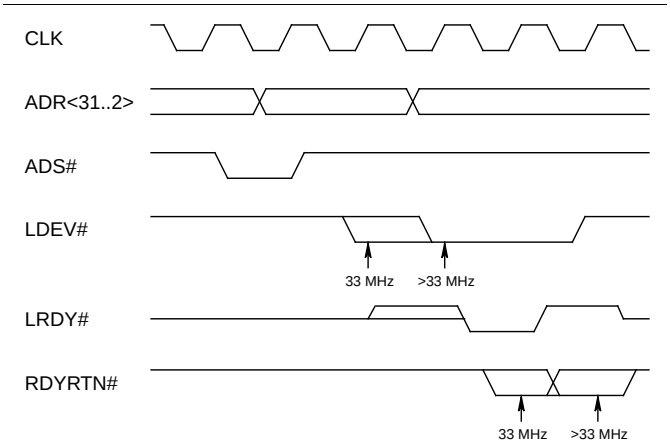

Figure 1. VL-Bus timing, showing fast and slow cycles.

VESA is also developing a 64-bit extension to the VL-Bus, which will support 32-bit VL-Bus peripherals as well. VESA plans to complete the mezzanine and 64-bit bus specifications by the end of August.

Intel Promotes PCI Bus

In parallel with VESA's efforts, Intel has been developing a standard aimed at solving a similar problem. Intel's Peripheral Component Interconnect (PCI) bus is intended to be a board-level interconnect bus for VLSI peripherals. While VESA's focus was on defining a standard connector for local-bus add-in cards, Intel's focus was on creating a standard for chip interfaces. The PCI bus has no defined connector, although it is anticipated that sockets or connectors for expansion modules will be implemented by some vendors. The electrical design anticipates up to two such connectors, but the specification leaves the selection of a connector and assignment of its pins to proprietary designs or future standardization efforts.

The PCI bus is one step further removed from the host CPU bus; although Intel calls it 486-like, it isn't. While a 486-to-VL-Bus interface can be little more than a handful of gates and some optional buffers, a typical 486-to-PCI interface is relatively complex. Intel does not expect PCI to be used in systems until it is directly supported by chip sets and peripherals.

Intel calls PCI an "intermediate local bus" to distinguish it from the CPU's pin bus, or a slightly modified version of the CPU pin bus such as the VL-Bus. VESA calls PCI a mezzanine bus, but Intel's designers avoid this term because they associate it with a physical packaging scheme where small add-in boards are mounted in parallel with a larger board, such as the iSBX bus used for Intel's MultiModules.

Figure 2 shows a block diagram for a typical PCI-bus system. The DRAM controller is shown as being associated with the PCI bridge because Intel expects most desktop designs to provide a dual-ported DRAM

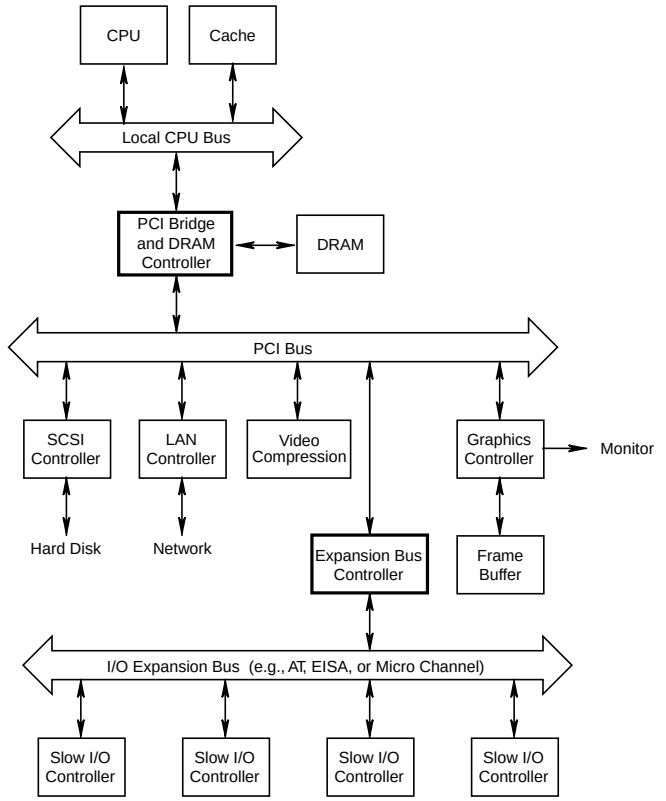

Figure 2. Block diagram of a typical PCI-bus system.

controller that allows access from either the CPU's local bus or the PCI bus. This allows the CPU to continue operating while the graphics controller or another PCI peripheral accesses the DRAM.

Both PCI and VL-Bus will be supported by system-logic chip sets that produce the required control signals and buffered address and data buses. In the long run, it seems likely that Intel will include a PCI interface as part of future processor designs, but Intel declined to comment on any such plans.

Intel's goal in creating the PCI bus was more than just accelerating graphics performance. Intel wants to continue to reduce the cost of building a PC while increasing performance, and the PCI bus promises to do this by eliminating most of the glue logic required for peripheral interfaces and providing a higher-bandwidth communication channel. Intel was disturbed by the high chip count in a high-end PC, as compared to a SPARCstation, which uses many fewer chips yet includes more functions. Indeed, the PCI bus is conceptually similar to Sun's SBus. PCI is intended to provide a constant interface for peripheral chip vendors so they can get off the "processor treadmill" of modifying their interfaces for each new generation of microprocessor.

The present specification calls for up to 10 devices on the PCI bus at speeds from 8 to 33 MHz, hence it is known as the "33-10" specification.

Name	Description
AD[31::0]	Multiplexed address/data bus
C/BE#[3::0]	Multiplexed bus command/byte enables
PAR	Parity for AD[31::0] and C/BE#[3::0]
FRAME#	Asserted by master during entire access (may include multiple transfers)
TRDY#	Target ready (indicates slave can accept current transfer)
IRDY#	Initiator ready (indicates master can complete current transfer)
STOP#	Asserted by slave to stop an access
DEVSEL#	Device select (indicates a slave is responding to the cycle)
IDSEL#	Initialization device select (used to select configuration space)
REQ#	Bus request (slot-specific)
GNT#	Bus grant (slot-specific)
CLK	Clock
RST	Reset
D[63::32]	Expansion of data bus to 64 bits
BE#[7::4]	Byte enables for upper 32 bits of data bus
PAR64	Parity for D[63::32] and BE#[7::4]
REQ64#	Asserted by master to request 64-bit transfers
ACK64#	Asserted by slave to acknowledge 64-bit compatibility
LOCK#	Resource lock
PERR#	Bus parity error
SERR#	System error
SBO#	Snoop backoff (causes bus cycle to be retried)
SDONE	Snoop done (indicates SBO# is valid)
TDI	Test data input (JTAG interface)
TDO	Test data output (JTAG interface)
TCK	Test clock (JTAG interface)
TMS	Test mode select (JTAG interface)
TRST#	Test reset (JTAG interface)

Table 2. PCI Bus signals; the top group is the minimum set.

Table 2 shows the signals that comprise the PCI bus. The basic 32-bit set is used by all bus devices, with optional extensions to handle 64-bit data width, bus masters, cache coherency, resource locks, error reporting, and a JTAG test interface. The minimum bus width is 32 bits. Bus sizing below 32 bits is not supported. The PCI bus is a multiplexed bus, unlike a 386 or 486 processor bus, to minimize the number of signals required—an important consideration for a chip-level bus, since pin count is often a limiting factor.

There are two slot-specific signals: the request/grant pair used to gain control of the bus. The bus arbitration scheme is left to the designer, subject to limitations on maximum bus latency. The interface between the processor's local bus and the PCI bus isolates the two, and the host processor can run concurrently with a PCI bus master. Interrupts and DMA are not part of the bus standard, but a PCI device (if resident on the system board) can be connected to the system interrupt or DMA controllers.

Figure 3 shows a timing diagram for the fastest burst-read transaction. The master begins the cycle by

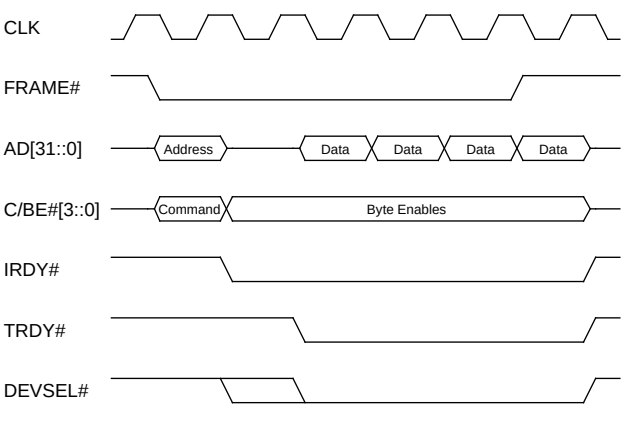

Figure 3. PCI bus timing for a maximum-speed burst read.

asserting FRAME#, along with the address and bus command. FRAME# stays asserted until the last data transfer is ready to complete.

The bus command encodes the direction of transfer and the destination address space. There are three address spaces: memory space, I/O space, and configuration space (a 256-byte, slot-specific address space). There are also command encodings for interrupt acknowledge cycles and other special cycle types.

The address and command are multiplexed with the data and byte enables. Byte enables can change on the fly, though some devices might report bus errors for some combinations of byte enables. One clock period of non-overlap is enforced whenever one driver hands off to another driver, as is the case in this example when the master quits driving the address so that the slave can drive data on the same bus lines. If this were a burst write, data could follow in the clock period immediately after the address.

Parity is applied to the multiplexed address/data bus and command/byte enables bus taken together. For bus slaves, generating parity is mandatory, while reporting parity errors is optional.

The burst order consists of linearly incrementing addresses. However, the standard does not exclude a master and slave from having a private burst order (such as the unique burst order used by the 486). The standard allows so-called "sideband" signals for making extensions beyond the basic protocols, such as a signal to indicate 486-style burst ordering. Such auxiliary signals provide flexibility, but they also invite incompatibilities if they are not standardized.

Both the master and the slave have ready signals that can insert wait states. The ready signal to the CPU is not asserted until both IRDY# (initiator ready) and TRDY# (target ready) have been asserted.

The DEVSEL# signal is asserted by slaves to indicate that they are responding to the decoded address. If no device asserts DEVSEL#, in most systems the transaction will default to the expansion bus.

	PCI	VL-Bus
Glue logic cost for 486 compatibility	High	Low
Protocol	Multiplexed	Non-multiplexed
Interrupts	None (can be added)	One (IRQ9)
Data bus width	32 or 64	16 or 32 (64 planned)
Connector	not specified	16-bit Micro Channel type
Maximum number of bus devices	10	3 (10 for mezzanine version)
Maximum bus clock frequency (MHz)	33	66
Number of bus signals	45-96	88
Max. bandwidth (Mbytes/sec)		
Single-word transfers	33	67
Four-word burst	76	106
100-word burst	128	106
Concurrent bus master/host CPU operation supported	Yes	No (Yes for mezzanine version)

Table 3. Key differences between the PCI bus and VL-Bus. Bandwidth figures assume no wait states, 32-bit width, and 33-MHz clock.

Each PCI device must implement a mandatory subset of the 256-byte (per device) configuration space. The space consists of a 64-byte header in a fixed format common to all devices and a flexible, 192-byte device-specific area. All PCI-compliant devices must implement four 16-bit words in the header that define the vendor ID, device ID, command register, and status register. Some devices may need to implement other header registers, such as the cache line size register, if they contain functions described by these registers. Features such as the configuration registers would be expensive to implement with discrete logic; Intel's intent is that these features will not be implemented in glue logic, but that new PCI-specific peripherals will be introduced.

Comparisons and Conclusions

Table 3 shows the key differences between the VL-Bus and the PCI bus. One major difference is that the VL-Bus does not allow concurrent operation of the host processor and a VL-Bus master, although the mezzanine version of the VL-Bus will add this capability.

Because the VL-Bus has an option for a 16-bit data path, it can easily be used in a 386SX-based system. The PCI bus requires a 32-bit data path; while it could be used in a 386SX system by having a PCI bridge that performed the size translation, it is unlikely to be cost-effective for this class of system. It seems likely that 386 support was low on Intel's priority list, and the PCI bus was no doubt designed with the P5 in mind.

The maximum bandwidth of the two buses is comparable. The VL-Bus allows for faster transfers when not using burst mode, but this speed cannot be achieved with today's peripheral chips because of the very fast

For More Information

The draft VL-Bus standard is available only to VESA members; when it is released in final form, it will be openly available for a nominal charge. Membership in VESA is open to all interested companies for a fee of $1000 to $4000 per year, depending on company size. Contact VESA, 2150 North First Street, Suite 440, San Jose, CA 95131-2020; 408/435-0333; fax 408/435-8225.

Intel has established the PCI Special Interest Group (SIG) to administer the PCI specification and control its evolution. A copy of the specification can be purchased for $100, and companies planning to build PCI products can join the PCI SIG for $2500 per year. Members vote for the nine-member steering committee, get updates to the specification, may submit revision proposals, are assigned a vendor ID#, and get technical support. Contact the PCI SIG, c/o Intel Corp., 5200 Elam Young Pkwy, HF3-15A, Hillsboro, OR 97124; 503/696-2000.

access time required. With long bursts, the PCI bus has a slight speed edge, since the current VL-Bus specification does not support bursts longer than four words. What really matters is not the peak bandwidth of the bus but the effective bandwidth for information transfer between the processor and the peripherals. Since this is likely to be limited by the speed of the peripherals in most systems, the peak bandwidth differences between the two buses are probably not significant.

Intel believes that a CPU-to-local-bus bridge with FIFO buffers is important for peak performance. While there is no reason why a VL-Bus system could not implement such a scheme, most VL-Bus systems are likely to have much simpler interfaces.

Since the VL-Bus is quite similar to the 486 bus, it is easy to adapt existing system-logic chip sets and local-bus peripherals for use with it. As a result, VL-Bus systems and peripherals will appear quite quickly. In fact, numerous prototypes have been demonstrated, and the first chip sets and motherboards supporting the VL-Bus have already been announced. The PCI bus, on the other hand, requires more complex logic in the PCI bridge and in the (on-chip) peripheral interfaces. PCI products will require more new silicon design, so products will emerge more slowly. Intel expects PCI technology demonstrations by the end of this year, with the first silicon support and systems appearing in the first half of 1993.

PCI systems will also require system software (BIOS) modifications, since new configuration registers have been defined. The VL-Bus, on the other hand, has been designed to be transparent to existing software. This highlights a key difference in attitude between the two groups—VL-Bus was conceived as a straightforward upgrade to existing system designs, while the PCI bus is intended to serve as the heart of an entirely new PC system architecture.

The fact that the VL-Bus has specified a standard connector will facilitate a third-party market for VL-Bus boards, including not only display controllers but also disk and LAN interfaces. This is especially important for graphics (and other) add-in card vendors, who could see their businesses drastically reduced if most system vendors provided high-performance peripherals on the motherboard or on an adapter board using a proprietary connector.

Some system makers would prefer that local-bus connectors remain proprietary, since this allows them to be the only supplier of boards that plug into those slots. This is probably one reason why Intel has not specified a standard connector for its PCI bus; to do so would have been a politically dangerous thing to do, considering the importance of gaining the support of major system vendors. Intel has left the selection of a standard connector, if any, up the PCI committee, so whatever decision is reached will come from the members, not from Intel.

While Intel has been careful to emphasize that the PCI bus should not be thought of as a replacement for the AT, EISA, or Micro Channel buses, this positioning, like the lack of a standard connector, is in part an attempt to protect the short-term interests of the supporting system vendors. In time, however, PCI or VL-Bus could indeed lead to the majority of systems being built without any of today's standard I/O buses.

With a graphics adapter, SCSI interface, and LAN controller all on the local bus, very few users would need any other I/O cards (assuming that the standard complement of serial ports, parallel ports, mouse interface, etc. are included on the motherboard). To be sure, there will always be some users with laboratory applications or other unique requirements that will continue to require systems that can support the widest variety of I/O cards, but such users are in the minority. Replacing the AT, EISA, or Micro Channel bus with a few local-bus connectors and a built-in SCSI port would reduce system cost while increasing performance—surely a winning combination.

Press reports have played up the rivalry between VESA and Intel's PCI group. It is indeed unfortunate that the two buses were announced simultaneously and will compete for the attention of system, chip-set, peripheral chip, and add-in board makers.

Table 4 shows the announced VL-Bus supporters, and Table 5 lists the PCI advocates. Note that there is considerable overlap between the two lists; many chip companies will support both, and some system makers will produce VL-Bus systems this year and add PCI systems in the future.

System and Motherboard OEMs
- Advanced Integration Research
- Alpha Research
- CMS Enhancements
- CompuAdd
- Diamond Computer Systems
- DFI
- Digital Air Systems
- Elanex PLC
- Everex
- Genoa Systems
- Hyundai
- Micronics
- NexGen Microsystems
- Northgate
- Oakleigh Systems
- Point Corporation

Graphics Chip Vendors
- Appian Technology
- ATI Technologies
- Avance Logic
- Chips and Technologies
- Cirrus Logic
- Integrated Information Technology
- NCR
- Oak Technology
- Primus Technology
- S3
- Sierra Semiconductor
- S-MOS Systems
- Trident Microsystems
- Tseng Labs
- Weitek
- Zeos

Graphics Adapter Vendors
- Actix Systems
- ATI Technologies
- Advanced Integration Research
- Diamond Computer Systems
- Genoa Systems Corp.
- Matrox
- NexGen MicroSystems
- Orchid Technology
- Point Corporation
- RasterOps
- Sigma Designs
- STB Systems

Hard Disk Adapter and SCSI Chip Vendors
- Alpha Research
- Data Technology Corp.
- NexGen Microsystems
- NCR
- Point Corporation
- Ultrastor Corporation

System-Logic Chip Set Vendors
- Appian Technology
- Chips and Technologies
- OPTi

Others
- AMD
- Anigma
- Apcal
- Austek Microsystems
- C-Cube Microsystems
- Metagraphics
- Microsumit K.K.
- Seiko Instruments

Table 4. Companies planning to provide chips, boards, or systems using the VL-Bus.

Intel has an impressive list of first- and second-tier PC vendors backing PCI, while the VL-Bus supporters (announced so far) among system makers are predominantly smaller companies. According to VESA committee members, several large computer makers—including ones on Intel's PCI list—plan to produce VL-Bus systems but are not yet willing to reveal this fact publicly. The VL-Bus also has the support of graphics board makers, for whom PCI does not yet have anything to offer—without a standard connector, there is no PCI add-in board business. PCI appears to have been driven by system makers, while VL-Bus was driven by graphics chip and board makers.

Both efforts have been underway for some time, and each group has made a different set of tradeoffs. Despite efforts to avoid a bus war, VESA could not realistically drop its proposal and adopt PCI, because PCI won't provide a solution soon enough. Intel, on the other hand, has put a great deal of effort into PCI, with a different focus, and it has the backing of many major system makers; Intel was not going to drop PCI in favor of the VL-Bus.

System and Motherboard OEMs
- Acer
- ALR
- AMI
- AST
- Compaq
- DEC
- Dell
- Epson
- Fujitsu
- Gateway 2000
- HP
- IBM
- Intel
- Micronics
- Mitsubishi
- NCR
- NEC
- Oki
- Olivetti
- Siemens
- Tandy
- Unisys
- Zenith Data Systems

Graphics Chip Vendors
- ATI Technologies
- Cirrus Logic
- Headland
- Intel
- Matrox
- NCR
- Tseng Labs
- S3
- Western Digital

SCSI Chip Vendors
- Adaptec
- NCR

LAN Chip Vendors
- Intel
- Texas Instruments

System-Logic Chip Set Vendors
- Headland
- Intel
- VLSI Technology
- Western Digital

Other Chip Vendors
- Intel (DVI)
- National
- NCR

Table 5. Companies planning to provide chips, boards, or systems using the PCI bus.

The two organizations did discuss a collaborative approach that would have put both buses under the auspices of the VESA organization, positioning the VL-Bus as today's solution for 386 and 486 systems and the PCI bus as a longer-term, higher-end approach for 486, P5, and future systems. Intel backed out of this proposal, however, causing bitterness among some VESA members. According to one source, some of Intel's major system partners preferred to keep the PCI bus under the control of a separate organization. VESA members wanted the VL-Bus to have a long-term role by including both buses in PCI systems, but Intel didn't support this approach.

The PCI supporters probably hope to ignore VL-Bus until they have PCI systems ready, allowing them to focus on a single approach. If VL-Bus takes off as quickly as its proponents expect, however, this could leave PCI supporters in a difficult competitive position for the six-month or longer period between production of VL-Bus and PCI bus systems.

Despite attempts to downplay the "bus war" aspect of the VL-Bus/PCI controversy, the reality is that there will be a market battle between the two standards. VL-Bus has the early advantage, but when PCI silicon and systems become widely available, the magnitude of support Intel has rallied behind PCI bus could give it the long-term edge. In any case, the days of graphics controllers and other high-speed peripherals on the antiquated AT bus, the baroque EISA bus, or the unpopular Micro Channel are clearly numbered, and the PC architecture will be better for it.♦