A BUS TOUR

Why the big controversy over bus architectures, and why should you care?

George White

If you own a personal computer, you are more or less familiar with the computer's bus. These days, debates rage over the relative merits and weaknesses of the IBM PC AT bus versus IBM's new Micro Channel architecture (MCA) or the yet-to-be-released Extended Industry Standard Architecture (EISA). New 32-bit buses, like the Mac II's NuBus, are touted as surpassing older, 8-bit buses in speed and memory capacity. However, if you crack open your computer, you may be hard-pressed to locate the bus, since it is simply a collection of signals and their protocols, which are used to communicate between boards.

A bus is physically embodied in the connectors that carry its signals, and the logic on each board that implements the bus protocol and connection. Essentially, the three major types of buses are the system bus, the I/O bus, and the memory bus.

System, I/O, and Memory

Minicomputers and supermicrocomputers are often designed around a central common bus to which the CPU memory and the high-performance I/O are connected. This arrangement qualifies as a system bus in that it forms the backbone of the computer.

You find I/O buses at both ends of the computer spectrum. Very large computers often have an I/O bus in addition to a system bus. They may use a proprietary and specialized system bus along with an industry-standard I/O bus that allows support of various peripherals. Personal computers often use only an I/O bus, with the CPU and memory having a close nonbus connection.

A slot that accepts manufacturer-specific memory-expansion boards is not really a bus. The signals that pass to and from this slot are merely an extension of the DRAM chip signals and provide no generality (i.e., the slots are good only for DRAM boards that are manufacturer-specific). All 80386 microcomputers have 32-bit CPU-to-memory pathways, which are important features of these machines and provide much of their performance. However, you cannot think of these pathways as buses, because they only provide a connection to a manufacturer-specific memory board.

Industrial-Strength Buses

Although it's getting harder to draw a line between personal computer buses and more "industrial" buses like Multibus and VMEbus, there are important distinctions. While multimaster capability is a novelty in personal computer buses, it's a necessity for industrial buses.

In any bus transaction, there is a master and a slave. The master initiates the transaction, and the slave responds. All industrial buses and the MCA provide general mechanisms to arbitrate the bus and turn mastership over to one of the boards in an add-in slot. The basic hardware is fairly simple; how the feature is used can vary widely. The basic use of a multimaster capability is to allow I/O cards to perform true direct memory access (DMA) and to access data from main memory independently of the central processor. In the XT and AT buses, there is generally only one master, the motherboard.

Outside the personal computer world, a bus without multimaster capability would not even be called a bus. On the other hand, the built-in DMA channels in personal computer buses are unheard of in industrial buses.

In general, the key distinction between an industrial or minicomputer system and a desktop system is the motherboard. Desktop systems have one—industrial systems do not. A VMEbus-based system starts out as an empty card cage. There is no presumption about what type of CPU the designers will use or whether they will construct a multiuser computer, RISC workstation, process-control system, or flight-simulator controller.

In the design of a personal computer, it makes sense to put as many functions as possible on the motherboard. Conversely, designers of industrial buses strive to minimize the centralized logic. Most industrial buses require only clock-generation logic. Futurebus manages to dispense with even this clock

continued

ILLUSTRATION: DAVID TILLINGHAST © 1989

generation and requires no centralized logic at all.

Cost has been another issue separating these bus categories. Personal computer users are cost-sensitive, while industrial system users are more concerned about performance and reliability. As personal computers become more powerful and are increasingly used as servers and multiuser systems, designers and users find the issues of industrial buses becoming more important.

Which Bus to Ride?

Current systems are built on a wide variety of buses (with more being created all the time), each having certain higher-level properties.

Although not a technical property, the degree of a bus's openness is one critical feature. Many buses like Multibus I and II, VMEbus, NuBus, and Futurebus are IEEE/ANSI standards. Other buses are "open," but their futures are controlled by one manufacturer (e.g., IBM's MCA). Still others are de facto industry standards that no company can continue to unilaterally influence (e.g., the PC AT bus, the so-called "industry standard architecture," or ISA).

While the subject of form factors may be mundane, designers obviously cannot put as much logic on a small board as they can on a larger board. Therefore, the size of usable board space on a bus's add-in cards may limit the number of boards that a user will have to choose from. The other real estate issue is the type of connector, or connectors, from the bus to the board (see photo). The industrial buses (including the NuBus used in the Mac II) have long since gone to two-piece connectors rather than the less-reliable edge-card connectors used in personal computers.

While performance is important, raw speed is not always the most meaningful bus criterion. How fast a bus can theoretically transfer data in a peak burst may not be indicative of real performance. Performance also depends on the speed of bus arbitration, whether or not arbitration can be overlapped with the previous data transfer, and whether existing cards run at maximum bus speeds.

Although a few systems are bottlenecked by the data transfer rate of the I/O bus, more are likely bottlenecked due to the lack of intelligence on the cards plugged into the bus. Bus features like multimastering can encourage the development of intelligent I/O controllers that can contribute more than raw transfer speed.

All the industrial buses have IEEE specifications. This means that not only is there a tight specification for designers to follow, but the evolution of the bus has been taken away from one or two manufacturers and placed in the hands of a democratic body. While committees don't have a history of successfully inventing new ideas, they have been useful at codifying technology and thus providing stability of bus definitions.

The tightness of a bus specification is directly related to how easy a bus is to design to and how likely boards of different manufacturers are to work together. For example, although there is no solid specification for the AT bus, the mass market has created an evolutionary process that weeds out computers or add-on cards that don't work well with the large installed base of AT clones.

Several industrial buses, notably NuBus and Futurebus, have had *processor independence* as important objectives. This means that they are designed not to favor one CPU interface style over another, but rather to provide a more general model of communication. In contrast, the XT and AT buses are simply decoded versions of the processor signals from an Intel microprocessor.

Standard Features and Optional Packages

The basic purpose of a bus is to get bytes moved from one board to another in an efficient and standard way. Many features can be wrapped around this basic "truth." Some features are key to creating reliable, fully functioning systems, while others are bells and whistles.

Broadly speaking, *protocol* refers to the types of transactions that a bus supports. The basics are reading and writing, and, in fact, these are quite sufficient for most systems. Others might be block reading and block writing, operations that transfer multiple data items in one burst transaction. The Futurebus defines *broadcast* as a write to multiple slave boards, and *broadcall* as a read that performs an OR on the data from multiple slave boards.

Data width is a fairly basic feature; essentially, it tells you how many wires the bus has, each one leading to a bit in an address. A bus is generally 8, 16, or 32 bits wide. Most VMEbuses are 32 bits wide, but an allowable subset is 16 bits. NuBus and Futurebus are 32 bits wide with no subset. While the MCA is billed as a 32-bit bus, most MCA slots are 16-bit only.

One AT bus limitation is the size of the address space. With 24 bits of address space, only 16 megabytes of physical memory (2^{24}) can be used. That storage capacity seemed like a lot in 1983, but it will soon be limiting. All the industrial buses have a 32-bit address space, although the actual size of the address space can vary (VMEbus can be either 24 or 32 bits).

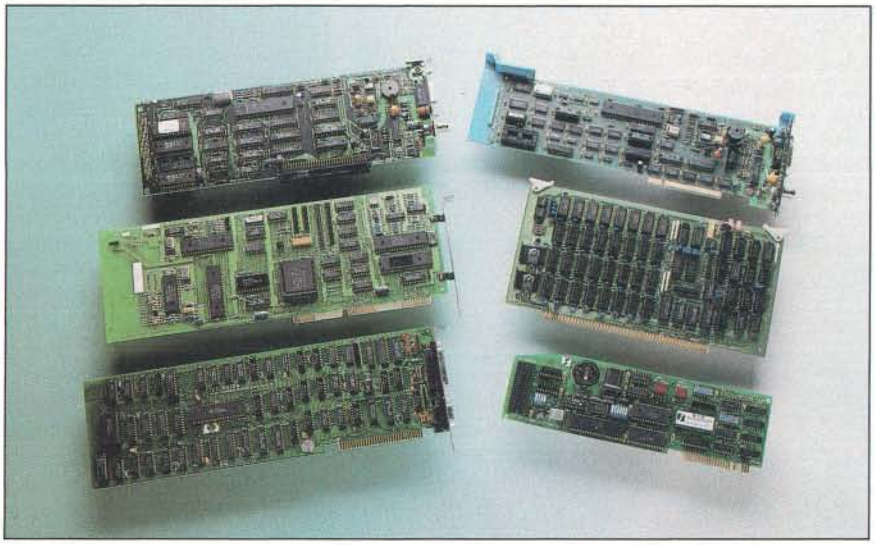
Formerly, when bus designers gathered, their most heated discussions concerned the issue of *synchronous* versus *asynchronous* buses. The first uses a single clock signal, propagated to all slots, to time all data and control information transfers. Typically, data and control lines are only valid on a certain clock edge. In an asynchronous bus, no central clock is used, and some form of handshake replaces the clock's function.

In a nutshell, asynchronous bus operations set no upper limits on the bus speed, while synchronous buses may make it easier for designers to develop more reliable, high-performance systems. NuBus and Multibus II are synchronous; VMEbus and Futurebus are asynchronous. The asynchronous school holds that synchronous buses are inherently limited by contemporary technology. The synchronous school thinks that pure, reliable asynchronous buses are difficult to invent and design to and that, in reality, the promised future performance gain is slight.

Since interrupts seem fairly basic and critical, it may come as a surprise that neither Multibus II nor Futurebus has them, and NuBus's interrupt line was only grudgingly added late in its design cycle. The standard idea of an interrupt is that a board pulls on a wire when it needs service from a single CPU. But what if you have more than one CPU? It would be nice to have a way for an I/O card to direct an interrupt to one of many CPUs in a system. The CPUs also need a way to interrupt the I/O cards and, in some cases, other CPUs on coprocessor cards.

The conventional interrupt line that the I/O board drives is quite limiting in that all devices that need to interrupt the CPU must be multiplexed onto a single line. More-advanced buses use the standard bus write transaction to convey the information that one board wants attention from another. This makes interrupts a special case of a memory write transaction, provides flexibility and directability, and eliminates special signals and hardware that would otherwise be needed. Of course, today's personal computers typically have a single CPU, but multiprocessor microcomputers are coming on strong.

Direct memory access is a feature of both personal computers and larger machines. However, the name does not mean the same thing in both realms. On the VMEbus, a controller board



Add-in boards (specifically, their connectors and the logic paths) represent the physical embodiment of computer buses. Shown here, clockwise from upper left, are boards representing six of the most popular microcomputer buses: the Mac II NuBus; the MCA; the S-100; the Apple II; the IBM PC 8-bit, and the IBM PC 16-bit. The industrial buses (including the NuBus used in the Mac II) have long since gone to two-piece connectors rather than the less-reliable edge-card connectors used in personal computers.

that is said to do DMA could arbitrate for the bus and act as bus master in transferring data from itself to memory, with no intervention by the main processor board. This simple feat would be hailed as a breakthrough example of multimastering in the personal computer world.

Personal computers have a fixed number of DMA channels on the motherboard. "Indirect" memory access would be a better name, since personal computer DMA is not really performed by the I/O board as much as by DMA chips on the motherboard.

In minicomputer systems, controllers are often developed that read control blocks from memory, perform the function indicated, put status information back into memory, and optionally interrupt the controlling CPU. Multimaster buses make this type of operation possible in microcomputers as well.

The Magic of Multiprocessing

The most sophisticated systems made possible by multimaster buses are those with true multiprocessor capabilities. Some people confuse multimaster with multiprocessor. Multimaster operation is necessary, but far from sufficient, to create a true multiprocessor. A true multiprocessor bus should also have an interrupt scheme that lets any board interrupt any other board; a particularly efficient arbitration method; and provisions for supporting multiple boards with caches.

Arbitration is an operation that keeps all the masters from trying to use the bus at once. The schemes for accomplishing this differ from bus to bus. Multibus I and VMEbus use arbitration schemes that involve daisy-chained signals. This is somewhat awkward in that any unused slots must have special jumpers inserted to continue the daisy chain.

In most modern buses, arbitration for a subsequent data transfer is carried out on a separate set of lines from those used for data transfer. This allows the overlapping of arbitration operations with data transfer. As a result, the arbitration phase adds no time to the resulting operation. When one data transfer is completed, the next one can start immediately. The MCA is the exception to this practice, performing arbitration in series with the data transfer. Thus, the arbitration phase adds to the total transaction time.

Caches are becoming more important in both the personal computer and supermicrocomputer markets. Processors are so fast that DRAM cannot keep up. A cache of static RAM is the only way to keep the CPU fed with data. Caches can be complicated, and, in a multiprocessor system, they may be especially complicated. Some buses provide hardware support for what is called the cache coherency problem. Except for a handful of proprietary buses used in high-end computers, the Futurebus is the only open bus with this feature.

These are the features most often contrasted on current buses. If industrial buses and personal computer buses continue to converge, be prepared for the marketing of bus enhancements such as geographical addressing, broadcall transactions, and cache coherency.

A Bus Inventory

The S-100 was the first microcomputer bus used in machines from different manufacturers. It was used in systems such as those from CompuPro/Viasyn. The S-100 bus provided users with the ability to add both I/O and memory options to their systems and offered a sophisticated multimaster arbitration scheme not seen in personal computer buses until the MCA. In some ways, the S-100 was the precursor to both the industrial microcomputer buses (e.g., Multibus I) and the personal computer buses (e.g., Apple II).

An 8-bit bus at first, the S-100 was extended to 16 bits. An IEEE working group ironed out several minor reliability and interoperability problems, a process that resulted in the IEEE 696 standard. Following the tradition of the S-100, most IEEE bus standards developed since then have been assigned numbers ending in 96: Multibus I is IEEE 796, Futurebus is IEEE 896, VMEbus is IEEE 996, Multibus II is IEEE 1096, and NuBus is IEEE 1196. The S-100 community is alive and well and exploring ways to extend its bus to 32 bits.

Like many buses, Multibus (now called Multibus I) started as the product of one company, became open and used by others, and then took on a life of its own. Various industrial systems and commercial computers were built around Multibus, including the original Sun boards from Stanford and later Sun Microsystems. Although not consciously processor-independent (having been developed by Intel), it was general enough that designers had no problem creating many 6800 Unix-based computer systems around Multibus.

Like the S-100, Multibus was originally an 8-bit bus, expanded to 16 bits in a cooperative effort between manufacturers and an IEEE committee. The Multibus market and user community became the model for others that followed.

continued

Although the Apple II bus was not noteworthy as a bus per se, it introduced two important features. First, each board had a ROM at a fixed address relative to the board's starting address, with both an input routine and an output routine for the particular board. This scheme provided a simple but elegant BIOS that allowed device-independent I/O operation. The second innovation was simply the shape of the board and the placement of the I/O connectors. Rather than being more or less square and sliding into card guides on both edges, it was rectangular and had its I/O connections on its outside edge. The same basic scheme was later used in the IBM PC.

In a chronology of microcomputer buses, putting Futurebus here seems odd. However, the Futurebus effort started in 1979, well before the IBM PC was announced and before the advent of the VMEbus. An IEEE Futurebus group was founded on the noble idea of developing a 32-bit bus *before* it was needed. The plan was to avoid the problems that come with an existing user base, a dilemma that faced the S-100 and Multibus I development groups. Those groups had to produce solidly engineered bus standards without unreasonably obsoleting the installed base. The Futurebus group started with a clean sheet of paper, was unencumbered by any installed-base compatibility constraints, and attracted input from bus experts worldwide.

Outside of the personal computer world, a bus without multimaster capability would not even be called a bus.

Futurebus has not yet been designed into any commercial machines, although a very early version of the specification was used as the basis of a bus in a workstation once produced by Tektronix. Many research laboratories around the world have built prototypes of various versions of the Futurebus specification. The IEEE committee that created the Futurebus standard is now developing a standard called Futurebus+, which is gaining wider support, including support from the developers of VMEbus and Multibus II.

The best feature of the IBM PC bus is that a lot were built and sold, so it was subsequently widely cloned. It is poorly specified, is not particularly fast, and has its interrupt lines upside down (i.e., an interrupt request is indicated by a low-to-high transition on an interrupt request line rather than the other way around). But the PC bus is adequate for its target applications and has admirably achieved a critical bus feature: wide usage.

IBM upgraded its original PC XT bus for use with the PC AT. The data path was widened to 16 bits, and more address lines and interrupt lines were added. The AT bus provides crude multimastering that is little-used because it is awkward to implement and not a very high-performance method.

Several companies (not including IBM) are now upgrading the AT bus again to the EISA bus. This 32-bit bus supports multiple masters and automatic system configuration. It's not a completely open bus, since those who want access to the specification must sign a nondisclosure agreement. An estimated 200 firms, however, have paid for the spec, and with the advent of the newly released Intel four-chip chip set, the bus wars are heating up.

VMEbus was announced in 1982 and soon became a winner in the industrial bus market. It's mainly a bus for supermicrocomputers, such as those from Sun Microsystems and MIPS Computer Systems. VMEbus has been used in industrial control applications and as the I/O bus for larger machines, such as those from Sequent Computer Systems. With the other buses now available, it's unlikely that standard office-environment PC-class machines will ever be built around VMEbus.

VMEbus used a two-piece connector with the Eurocard form factor. It had support for 32 bits, and three large organizations (Motorola, Signetics, and Mostek) endorsed it simultaneously. *Eurocard* is a term for a standard card-packaging system originally used in Europe. VMEbus, Multibus II, Futurebus, and the industrial version of NuBus all use Eurocard technology.

Although the VMEbus developers didn't have the lofty technical goals of the Futurebus developers, VMEbus filled a vacuum. There was a growing realization that the Eurocard packaging was superior to the standard edge-card scheme in general use in the U.S. and that a path to 32 bits would soon be needed. (In fact, VMEbus supports both 16- and 32-bit transfers. Early VMEbus systems used only the 16-bit option.)

The original "closed" Macintoshes (the 128K, 512K, and Mac Plus), which have no bus, demonstrated the desperate need for buses. Third parties developed a wide variety of add-in products, including memory expansion, coprocessors, and internal disks. These were installed in machines against Apple's wishes and in violation of factory warranties. The ingenuity and fearlessness displayed in providing Macs with these and other capabilities illustrate the importance of open buses.

Originally designed for high-end workstations and superminicomputer applications, NuBus has found its greatest success in the Mac II (a modified NuBus is also used in the NeXT computer). NuBus was created at MIT in 1978 as a bus for a high-end reconfigurable workstation. Later, a group at Western Digital transformed NuBus into its present state (except for its form factor). Texas Instruments subsequently bought the project and used the bus in its Explorer Lisp machine. NuBus was also used in the Lambda AI computer made by the now-defunct Lisp Machine, Inc.

Its use in the Mac II and NeXT computers puts NuBus at the intersection of the industrial and desktop buses. Although used in personal computers, it has the raw speed and features of Multibus II.

While PC-clone makers are developing EISA, IBM has bet on its MCA, an architecture that has proven to be controversial. The MCA's strong and weak points are the same: its incompatibility with the AT bus. In most technologies or markets, there is a time to break with the past in order to achieve an improvement in performance and features. The given in this process, however, is that the old must really be holding you back and the new must be a significant step forward. This is still an open question regarding the MCA.

The MCA's "new" features are primarily standard elements in industrial buses: multimaster arbitration, burst transfers, and sensible interrupts. Today, the MCA is being used predominantly in the IBM PS/2 product line.

Lining Up the Buses

Although the AT bus lacks auto-configuration and high-performance multimaster capabilities, it is adequate for most desktop applications. There has been a real need for bandwidth between

continued

the CPU and memory, but ad hoc manufacturer-specific CPU-to-memory paths have solved this problem. A few more power and ground pins would be nice, as would interrupt signals that aren't upside down. A written bus specification would also be helpful. However, in spite of these limitations, board and system designers have produced a wide variety of interoperative, reliable, satisfactorily performing products.

The MCA does offer advances over the AT bus. It has reasonable interrupt lines that are not upside down, allows multiple masters (as anything called a bus should), and has a reasonable number of ground signals. Although the MCA is a technical advance over the AT bus, from the point of view of the rest of the bus world, that isn't saying much.

Auto-configuration of the MCA is possible because of the Programmable Option Select registers that are addressed on a slot basis. On Futurebus, Multibus II, and NuBus, the equivalent to POS is called *geographical addressing*—a portion of a board's physical address space is tied to the slot where that board is physically located. Optionally, the MCA is a 32-bit bus. However, a board would generally be designed to plug into either a 16-bit or 32-bit slot, and since most MCA slots are 16-bit, the majority of MCA add-in cards are 16-bit, also.

Multimastering has real advantages if add-in cards make use of it. While not strictly needed for intelligent I/O cards (there are many for the AT bus), it does make intelligent I/O somewhat cleaner.

While NuBus is now viewed as a desktop machine bus, it was conceived as addressing the same technical needs and objectives as Multibus II and VMEbus. This concept gives it a unique position as the only bus designed for high-end applications that is also used in a mass-marketed product. Technically, it is a 32-bit, multimaster, DIN-connector, IEEE/ANSI-standard bus with auto-configuration. One missing feature is built-in support for cache coherency in multiprocessor, write-back cache systems. Of the buses mentioned, only Futurebus has such support.

The Bus Stops Here

Future high-end personal computers will have two conflicting needs: (1) advanced performance and features to support multiple processors and higher-bandwidth I/O, and (2) the availability of a wide variety of I/O options. The widest array of available options is provided by staying with the status quo, but additional performance and features require extra effort and bring up the possibility of incompatibility. The MCA takes one path through this problem, EISA another.

Some future needs, such as support for true multiprocessing, can be accommodated by specialized CPU-to-CPU-to-memory buses that can be independent of the I/O bus. A dual-bus approach can offer the benefits of both a popular I/O bus—which is not particularly fast—and an optimized intra-CPU and CPU-to-memory path.

The only example of a crossover bus is NuBus. Originally developed for supermicrocomputers or high-end workstations, it is now at home in the Macintosh and the NeXT machine. While several concepts from industrial buses, such as auto-configuration, two-piece connectors, and cache coherency, are likely to reach personal computers, the generality, form factor, and inherent additional costs of these buses will probably keep them off the desktop. ■

George White is a cofounder and president of Corollary, Inc. (Irvine, CA), a maker of multiprocessor PC systems. He was the chairperson of the IEEE 1196 NuBus committee. He can be reached on BIX c/o "editors."