Futurebus+ Coming of Age

High Bandwidth and Advanced Features are Key Attractions

By John Theus

This is the first of a three-part series on the Futurebus+ standard. The author is the president of the TheUs Group (Sherwood, OR), a consulting firm that specializes in high-speed interconnects with a focus on Futurebus+. John was the architect of the Futurebus and Futurebus+ parallel protocols, central arbitration, and Profile F. He is the chairman of P896.8, which is working to develop a next-generation desktop/mezzanine bus. John was employed by Tektronix, Inc. for 16 years where he was an architect and hardware designer in their workstation division. He led the development of the Tek XD88 series which featured a multiprocessor architecture built around the 88000 and Futurebus.

The BUSCON/92-West conference held in February marked the coming of age for Futurebus+. For the first time, several working Futurebus+ systems were shown, along with the infrastructure needed for industry-wide acceptance. The core specifications, officially known as *IEEE Std. 896.1-1991: Logical Protocol Specification* and *IEEE Std. 896.2-1991: Physical Layer and Profile Specifications* became approved standards last September. The draft specifications actually became stable during the first months of 1991, but the IEEE approval process added considerable delay, and the IEEE printing process has added still more delay (896.1 shipped on 3/13/92). However, while the IEEE process moves forward slowly, packaging, integrated circuit, board, system, and instrumentation manufacturers wasted little time in turning those unofficial specifications into implementations.

What makes Futurebus+ interesting is the combination of bandwidth and advanced features not found on any other standard bus. Key features include hierarchical cache coherency, split (also known as split response or write-only) transactions, split locks, broadcast, data snarfing, multiple packet transactions, fault tolerance, live insertion, incident wave switching, and completely metric packaging. (An expanded features list with explanations will be presented in part 2, along with how these features are applied.)

Bandwidth

Futurebus+ has a potential bandwidth of 4000 megabytes per second (MB/s). However, stating bandwidth in MB/s, like MIPS, is an abused measurement of performance. Although there are fewer variables involved in stating MB/s compared to stating MIPS, the context of the measurement is still critical. In the case above, this number is the maximum expected burst transfer rate (125 megatransfers/s) using the maximum data width (256 bits) with a very good electrical environment.

Futurebus+ allows implementations to use data widths from 32 to 256 bits, but it's expected that almost all applications will use either 64 or 128 bits. To eliminate the effects of this variable, Futurebus+ burst bandwidth is often stated in megatransfers/s (MT/s), which is independent of data width.

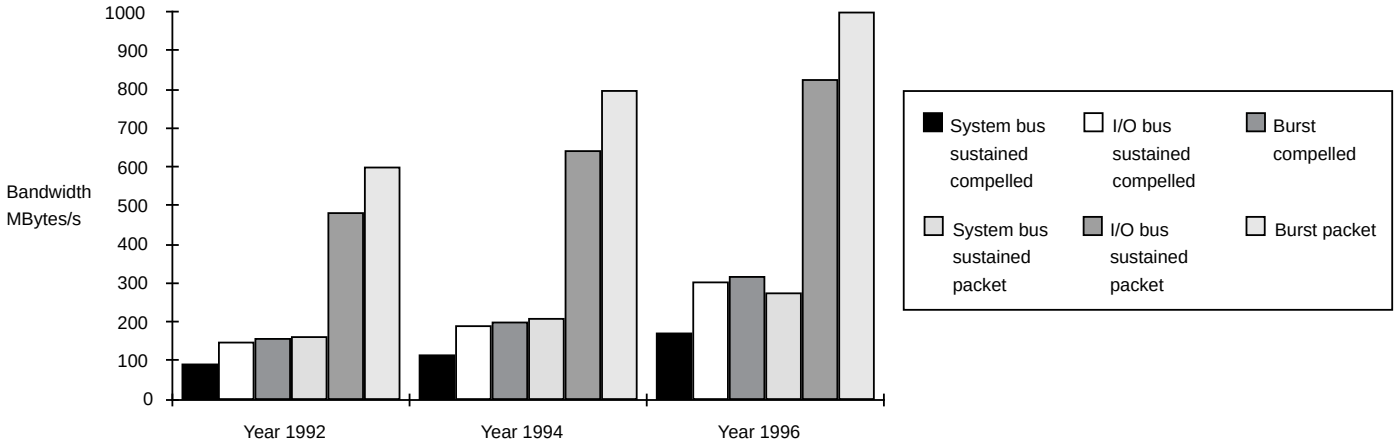

Figure1. Bandwidth anticipated for Futurebus+ using various protocols, assuming a 64-bit-wide data path.

Identifier	Title	Status	Comments
896.1-1987	Futurebus Backplane	Standard	96-pin, Eurocard Standard
896.1-1991	Futurebus+ Logical Layer	Standard	Logical Protocols
896.2-1991	Futurebus+ Physical Layer and Profiles	Standard	Node Management, Profiles A, B, and F
P896.3	Futurebus+ Recommended Practices	In sponsor ballot	Guide to using FB+ Protocols in Secure, Real Time, and Fault Tolerant Systems
P896.4	FB+ Conformance Test	Working Group	
P896.5	FB+ Profile M	Working Group	Specification for Military Systems
P896.6	FB+ Profile T	Working Group	Specification for Telecom Systems
P896.7	FB+ Profile C	Working Group	Interconnect Between FB+ Systems
P896.8	FB+ Profile D	Working Group	FB+ Desktop Systems, Logical Protocols
P896.9	Fault Tolerance	Working Group	
P896.x	896.1 / 896.2 Errata	Working Group	
	FB+ Profile S	Study Group	Specifications for Spaceborn Systems

Table 1. Futurebus+-specific IEEE standards and development projects. Numbers with a "P" prefix are IEEE-authorized standards projects. Status definitions: Projects are often preceded by a "Study Group" that determines if there is sufficient interest in the proposed project and develops a project scope. A "Working Group" creates the specification and works to achieve consensus about its contents. A working group forwards the completed document to its sponsor (e.g., the BASC) which places it "In sponsor ballot." The sponsor forms a balloting body of interested individuals that represent a mix of users, manufactures, technical experts, etc. Documents that pass sponsor ballot are forwarded to the IEEE Standards Board, which meets quarterly, where they examine the procedures used to develop the specification. Approval by the standards board authorizes the publication of the document as an IEEE standard, which occurs after a round of IEEE-staff editing.

Futurebus+ uses a combination of fully compelled asynchronous protocols (all state changes require a handshake) along with speed-negotiated source-synchronous protocols. These protocols allow performance to be determined by the speed of the agents actually participating in the transfer. On the positive side, performance will grow along with technology. On the negative side, performance calculations hold only for specific implementations.

Finally, stating a burst transfer rate ignores all the overheads that are required to set up the transfers. Overhead comes from the bus protocols themselves and the mix of transaction types processed on the bus. The bus protocols introduce overhead from many areas, such as arbitration, passing control from one master to the next, and transferring address and command information. The transaction mix is critical since reads require more bus time to process than writes, even if the read access time is zero. If the application uses cache coherency transactions, then additional overhead comes from coherency operations, such as invalidate, which transfer no data.

The time lost due to protocol-related overhead is amortized across the length of a data transfer. Short transfers are less efficient than long transfers, and short transfers become more prevalent as the data width increases. This is due to the architectural limits placed on the number of bytes in a transfer. Some examples of limits are cache line size, page size, disk block size, LAN buffer size, etc.

In order to present some comparable numbers, Figure 1 shows some typical bandwidths in MB/s versus the year of implementation. The chart is further broken into two sets, based upon the two available data transfer methods. These sets are the compelled data protocol, which uses a master-slave handshake for each data word, and the packet data protocol, which transfers a 2^n-size block of data per handshake using a special protocol (described in part 3). Within each of these sets, burst bandwidth and sustained application-specific bandwidths are plotted. The year-to-year projections assume an increase in integration coupled with improving device propagation speeds and timing skew.

The system bus example assumes that all traffic is cache coherent with a single 64-byte cache line per transaction. It assumes that 70% of the transactions are reads, 20% are writes and 10% are cache operations that do not transfer a cache line. The I/O bus example has no coherent traffic and a traffic mix that results in an average data transfer length of 512 bytes. In this example, 70% of the transactions are reads and 30% are writes. Both applications use a 64-bit data path and a physical environment from 896.2.

These examples show that sustained performance is very dependent upon traffic patterns. The performance of cached buses that use packet mode can be improved significantly by using an additional protocol, called multiple packet mode, which allows packets to be sent back-to-back with very little overhead.

A Brief History

The development of the Futurebus began in 1979 (prior to the VMEbus) when it became apparent that a standard bus was needed for 16- and 32-bit systems. The specification produced by this committee was in many respects similar to the VMEbus in its TTL nature

Identifier	Title	Status	Comments
P1014 Rev D	VMEbus Rev D	Working Group	VMEbus extension to 64 bit, SSBLT, etc..
P1014.1	VFEA to FB+ Bridge	Working Group	Bridge from VME64 to Futurebus+
P1101.3	Conduction Cooled	Working Group	Conduction Cooled Form Factors
P1101.4	SEM Mechanicals	Working Group	Standard Electronic Modules (Military)
P1156.1	Environmental Specifications for Computer Modules	In sponsor ballot	
P1156.2	Environmental Specifications for Systems	Working Group	
1194.1-1991	BTL Interface Circuits	Standard	
P1194.2	SCEM, Electrical	Working Group	Small Computer Expandability Module (Desktop)
1212-1991	Control and Status Register Architecture	Standard	
P1212.1	DMA Architectures	In sponsor ballot	
P1275	Open Boot	Working Group	
1301-1991	Metric Mechanicals	Standard	
1301.1-1991	Metric Mechanicals for 2mm Connector	Standard	
P1301.2	Metric Mechanicals Recommended Practice	Working Group	Guide to 1301
P1301.4	SCEM, Mechanical	Working Group	Desktop Form Factors
P1394	Serial Bus	Working Group	High Speed Serial Bus for peripherals and backplanes
P1496	SBus	Working Group	
	Modeling	Study Group	Recommendations for Machine Executable Specifications
	Multimedia Extensions	Study Group	Requirements for Multimedia Applications

Table 2. Futurebus+-related IEEE standards and development projects.

and protocols. The sponsoring committee felt it wasn't "good enough" and sent the specification back to the Futurebus committee. This action led to several resignations and the collapse of the working group.

The committee floundered for several years while the sponsors tried to find individuals interested in participating. Finally, interest started to build in 1983 and the foundation for a new specification was laid in 1984. During this period, Backplane Transceiver Logic (BTL) was designed as a TTL-friendly solution for incident wave switching in a backplane environment. Technology-independent fully-compelled protocols were analyzed and endorsed by the committee. The specifications were prototyped in 1985 and were shipped in a Tektronix NS32032-based production workstation in 1986. Along with the changes resulting from evaluating the prototypes, cache coherency hooks were added. A year of IEEE politics later, the specifications were approved and published as IEEE Std. 896.1-1987.

While the 1987 specifications were used in a few applications, it was far from a market success and it offered too little differentiation from the other available buses. Interest didn't expand until early 1989 when Futurebus was independently selected as the baseline specification by the VME International Trade Association (VITA) for its "Next Generation Architecture Bus Standard" (eventually called the "VME Futurebus+ Extended Architecture," VFEA), the telecom industry's "Rugged Bus" and the US Navy's "Next Generation Computer Resources" (NGCR) program for mission-critical computing. The agreement of these 3 groups to join with the original Futurebus committee brought together the critical mass necessary for success, and the creation of Futurebus+.

Core Specifications

Besides 896.1 and 896.2 mentioned above, several other IEEE specifications were created to support the requirements of the combined group (see Tables 1 and 2). In the packaging area, a new 2-mm grid connector was chosen for its higher pin density (twice that of a DIN 41612). Beginning in 1992, European equipment regulations would not allow the marriage of the "hard metric" (i.e., designed with metric dimensions) 2-mm connector with the "soft-metric" (i.e., designed with English dimensions and converted to metric) Eurocard-sized boards, so a new hard-metric circuit board standard was created. *IEEE 1301: Metric Equipment Practice for Microcomputers—Coordination Document,* and *1301.1: Metric Equipment Practice for Microcomputers—Convection Cooled with 2-mm Connectors* were created in record time. To address environmental issues, *1156.1: Environmental Specifications for Computer Modules* was created. In the electrical area, the BTL specification from the 1987 Futurebus specification was merged with the slightly different Pi bus specification into *IEEE 1194.1: Electrical Characteristics of Backplane Transceiver Logic (BTL) Interface Circuits.*

To conclude this first round of specifications, the software interfaces are specified in *1212: Control and Status Register (CSR) Architecture for Microcomputer Buses* and *1212.1: DMA Framework Architecture,* both of which were created in close coordination with the Scalable Coherent Interface (SCI), SerialBus, and Fu-

turebus+ working groups.

Open Standards

What is unique about these specifications is that every single document listed here is a proactive standard. Prior to this time, almost all IEEE bus standards were proprietary specifications that were brought before the IEEE with an existing installed base. Independent manufacturers and users had little or no input into those specifications' foundation. The standardization process was mainly a codification of an existing specification.

Creating a proactive standard is not an easy process. As documented above, the first Futurebus standard took eight years to create. However, the participants learned from that effort how to manage a public process, and still gain the advantage of opinion and experience from a broad section of industry. Futurebus+ required 2.75 years. The next round of major proactive standards should take less than 2 years.

Efforts like these are always tagged with the label "designed by committee." Futurebus+ was not "designed by committee," it was "approved by committee." The design was executed by a small group of experienced individuals, all from different commercial or educational organizations, with one person in charge of each major technological area. Few organizations could match the thousands of hours of human and computer time that went into the creation and testing of these specifications, and probably no single organization could bring to bear the range of knowledge, experience and perspective that went into these developments.

The process of designing Futurebus and Futurebus+ led to the creation of several important technologies, many of which have been incorporated into non-Futurebus applications. Most noteworthy are BTL, jumper-free board configuration, the cache coherency model (MOESI), a hierarchical cache coherency protocol, and a packet-length, parallel-bit, skew-immune transport technique. It has been often stated that even if the Futurebus development effort had never led to a specification, the research it stimulated would have been well worth the effort.

Today's Organization

What is generally not realized is that the Futurebus+ organization has grown into a multifaceted umbrella organization, called the IEEE Bus Architecture Standards Committee (BASC). This committee sponsors the development of new bus-related standards. The BASC sponsors hardware standards in the areas of mechanical, electrical, and logical protocols, and software standards in the areas of system boot and drivers. Along with the large number of Futurebus+-related standards, independent standards development such as SBus and VMEbus are also finding a home here.

The BASC provides an efficient infrastructure for these committees to meet by sponsoring week-long bimonthly meetings. Typically, more than 100 participants conduct up to 40 half-day meetings. The convenient meeting arrangements have created an atmosphere for the cross fertilization of ideas. For example, experts in space-based applications can discuss ideas with commodity packaging experts.

New Developments

The BASC is sponsoring the development of many new specifications. Two Futurebus+-related efforts are particularly exciting: the Cable Interconnect and the Desktop/Mezzanine Bus.

The goal of the Cable Interconnect committee is to specify the logical and physical means to connect together backplanes. The logical specification is intended to be technology independent and will address a range of configurations from tightly-coupled cache coherent to networked. The physical portion will include a number of technologies, such as Fiber Channel, SCI, and to-be-created Futurebus+-specific parallel and fiber-serial channels.

The Desktop/Mezzanine Bus targets the replacement of buses such as today's SBus, Turbochannel, and Micro Channel for shipment in the 1995 time frame. The requirements for this bus are still being discussed, but the principal points are high-volume single-chip implementation, sustained bandwidth in the 500–1000 MB/s range for the first implementation, 64-bit data path, six to eight postcard-size stacked boards, few options, and a transaction set that is easy to convert to and from Futurebus+.

A Range of Applications

An often heard statement about Futurebus+ is that it's too complex, and it has too many options. This opinion is easily reached if one reads the core specifications without understanding their function. The underlying philosophy of the Futurebus+ core specifications is to create a toolbox containing mechanical, electrical, protocol, and software interface solutions. Figure 2 demonstrates how these tools fit together to form a protocol stack. A specific application chooses the tools that are appropriate for the job at hand, and specifies this collection in a document called an application profile. In some cases, the toolbox may not have all the necessary pieces required by the application, so the profile can either specify what it additionally needs or point to a new companion standard.

For example, four of the major specified tools are cache coherency, message passing, fault tolerance, and live insertion. Applications exist, such as a typical workstation I/O bus, where none of these tools is

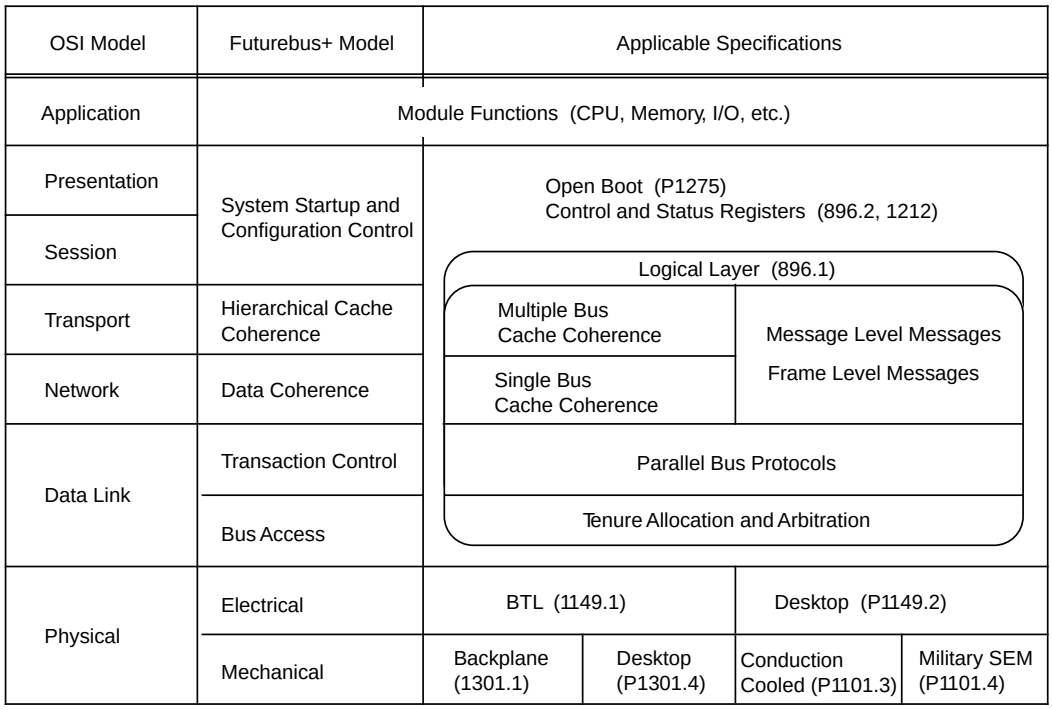

Figure 2. Futurebus+ is designed around a protocol stack that is patterned after the seven layer OSI model.

needed. A space-based application would want fault tolerance and possibly message passing. A military application might want all four tools. Many of the possible permutations of just this small list are likely to be needed by different applications.

Clearly, taken to the extreme, every system built around Futurebus+ could be incompatible with every other system. In practice however, the different industry segments have realized their common needs, and they have rallied around a small number of profiles. So while a board produced for one profile may not work in another profile, their mechanical, electrical or logical implementations may be in common, and the value of having a "standard" realized.

Principal Applications

Five applications areas have been addressed so far by Futurebus+ profiles. The three that are specified in 896.2 are for an I/O bus, a system bus, and a general-purpose bus. The other two, the military and telecommunications applications, are still in development, and they will be specified in separately numbered standards. A new application area — space-borne systems — is just starting and should shortly become an official profile-development project.

Each application area will be introduced here, but the specifics of each and the technologies they use will be covered in part 2 of this series. The I/O, system, and general-purpose applications have a common set of mechanical, electrical, and environmental specifications. As their names imply, they differ in their logical focus and their strictness.

The I/O bus application uses the smallest usable subset of the Futurebus+ protocols for a pure data movement environment. There is no support for cache coherency or lock operations. Some of the protocol's major stimulus-response events have their maximum permitted execution time specified. Protocol options are limited so that all boards designed for this specification will work together correctly across the entire bus-related feature set.

The system bus application adds to the I/O protocols cache coherency, locks, and the provisions for two different types of data transfer protocols. All protocol stimulus-response events have a maximum timing specification so that a guaranteed minimum performance calculation is possible. Protocol options are limited here also.

The general-purpose bus application removes most restrictions on protocol options. There are also no timing constraints. Vendors can choose any set of logical functions deemed necessary for their application. The specification does not guarantee that any two boards can do useful work together, so the system integrator is left with that responsibility.

The military application specifies a standard set of protocols across a wide range of physical and environmental implementations, ranging from the office to avi-

onics. Fault tolerance and high availability have a very high priority for this application. The ability to develop solutions using commercial grade hardware and then to convert those solutions into mil-spec hardware is very important in this application.

The telecommunications application is differentiated by its strict requirements for high availability, fault tolerance, and maintainability for systems that operate continuously. Live insertion will probably be used to help meet these requirements. Maintaining consistency with the world-wide telecommunications equipment standards (ETSI and NEBS) is also a high priority.

Current Status

Several manufacturers are now offering the full gamut of hard metric mechanical pieces that board and systems builders require. Included in this list are enclosures, backplanes, connectors, prototype boards with transceivers, and test fixtures.

BTL transceivers are available from several manufacturers, including National Semiconductor, Signetics, and Texas Instruments, in configurations ranging from 4-bit to 9-bit parts. These same companies plus Newbridge Microsystems are either shipping or have announced protocol controllers for several of the application areas.

A small number of processor, memory, and I/O boards have just become available from several companies. Boards from different vendors have been demonstrated working together in a single backplane. Systems companies have shown working systems with all internally designed boards, and with a mix of internally and externally designed boards.

Data acquisition boards for two of the top selling logic analyzers have been demonstrated. Each of the respective boards interfaces between the logic analyzer's probes and the bus, while maintaining the required Futurebus+ electrical environment. Both analyzers can display either timing or state information with transaction information decoded and presented as text.

Conclusions

Futurebus+ has arrived. Sufficient investment has been made to insure its success in several different markets. A solid technical foundation has been laid that combines newly invented technologies with some of the best of previously proven technologies. This foundation will support several generations of designs across an increasingly wide range of applications. While far from perfect in terms of both its design and its implementation, this time around Futurebus+ is "good enough."♦

Nimbus SPARC Chip Set

Continued from page 15

never completed its SBus interface chip. Second, Tera did not use the MBus, and was thus cut off from the emerging generation of processors. Finally, Tera built a relatively large organization with high cash requirements, while Nimbus has a fraction of Tera's overhead.

Market Potential

Nimbus plans to sell only the board kit, which consists of the seven Nimbus ASICs and a PC board that fits into the standard SPARCstation pizza-box enclosure, making system design as easy as possible for Nimbus' customers. A port of SunOS to the Nimbus system design is nearing beta release from Interactive Systems. In a striking departure from previous MBus chip set announcements, Nimbus already has a working version running OpenWindows on their hardware.

Nimbus claims its system will perform about 15% better than a SPARCstation-2 because cache misses are handled in 17 processor cycles, instead of 26 processor cycles for the SS-2. This is due, in large part, to the fact that the SS-2 does not use the MBus, and the DRAM is interfaced to the slower SBus. The performance claim assumes a cache miss rate of 2.5%, so the actual performance difference will depend on the application.

Since the Nimbus chip set implements the MBus level-2 protocol, system performance can be increased with dual-processor MBus modules. Capacitive loading problems prevent use of more than one MBus connector with the current chip set. Nimbus is planning a second generation that will handle more than one MBus module and operate much faster (between 50 and 60 MHz).

At long last, it appears that a variety of MBus SPARC processors will be available (see p. 1). This will give clone makers using the Nimbus chip set a simple way to differentiate their systems in the marketplace and yet offer easy upgradeability. Nimbus says the level of interest in MBus has increased dramatically because the SuperSPARC and hyperSPARC announcements make it clear to system makers that MBus and SPARC are finally "real." System makers can use the existing Cypress/Ross MBus module, based on the older 7C601 processor, and upgrade to SuperSPARC or hyperSPARC when those products become available.

Nimbus' chip set and board seem to be a simple, high-performance way for system makers to test the SPARC-clone waters. If the interest is really there among system makers and if the market is really ready to buy SPARCstation clones in large numbers, Nimbus is in an excellent position to reap some profit and set the direction for the next generation of SPARC-based computers. ♦