**LETTERS TO THE EDITOR**

**Data format and the S-100 bus**

Editor:

In “Data Format and Bus Compatibility in Multiprocessors” (IEEE Micro, August 1983), Hubert Kirrmann insists that “all existing bus standards indirectly impose a data format for memory because they are overspecified.” It is very unfortunate that Kirrmann did not read the approved S-100 bus standard instead of the draft that appeared in the July 1979 issue of Computer magazine.

The situation is most easily cleared up by quoting from the foreword of the standard:

To be more specific, the original standard described 16-bit transfers in terms of a high and low (or most and least significant) byte. This presented a problem because CPU chip manufacturers chose to order the bytes different from each other, that is, some transfer the high byte on the lower half of the data bus, and some do just the opposite. It was clear that the working group had to settle the issue of where to place each byte. A fundamental problem was that the group was split fairly evenly on this issue. The solution that evolved is both clever and unique in that it made everybody happy.

It was decided that the standard should not dictate a significance of the bytes at all. Instead, the standard concerns itself with making sure that bytes read or written in an 8-bit mode . . . [are] read or written consistently in a 16-bit mode, and vice versa. Originally, data bits were called DATA16-DATA0 in a 16-bit mode, which inherently designates significance. Now the 16 data bits are thought of only as two bytes: an odd byte and an even byte—now called OD7-0 and ED7-0 (OD for odd data and ED for even data). The terms high and low have been replaced by odd and even, respectively.

Basically, the rule is as follows: byte data that is written or read with A0 = 1 appears on the OD7-0 lines during a 16-bit transfer. Byte data that is written or read with A0 = 0 appears on the ED7-0 lines during a 16-bit transfer. The nomenclature of OD and ED (odd data and even data) makes it easy to remember the rule when looking at a schematic. All one has to do is think about the fact that any address with A0 = 1 is odd, and any address with A0 = 0 is even.

At a meeting of the IEEE 696 Working Group at Wescon in September of 1979, David Gustavson and I had a lively discussion about imposing a data format. Eventually, as the standard shows, he agreed with my views, but on the way there he made a very good point about why one format is better than the other. Character strings are necessarily big-endian, and therefore, for consistency, it is best to choose the big-endian for all data formats.

Kirrmann’s statement that “processors such as the TI 9900 . . . cannot be integrated into a justified bus” is mystifying. The first 16-bit implementation on the justified S-100 bus was the Marinchip 9900 CPU card. Since the 9900 does only word transfers, there is no difficulty at all in adapting it to a justified bus.

Tim Paterson
Falcon Technology, Inc.
6644 South 196th Street
Suite T-101
Kent, WA 98032

Author’s reply:

I am sorry for the inaccuracy about the IEEE 696 (S-100) byte ordering scheme. At the time I wrote the article, I had only the 1979 draft. The article was already in press when I learned that there was a more recent spec of IEEE 696. The changes had been published in the November 1982 and February 1983 issues of IEEE Micro, which I received in Europe with some months of delay. The new, formally approved IEEE-Std 696-1983 is now available from the IEEE Computer Society Order Department, PO Box 80452, Worldpay Postal Center, Los Angeles, CA 90080.

This brings us back to the discussion about the publication of standards that Bob Stewart started in IEEE Micro. The problem is not that the 696 draft had been published but that I had been unable to keep track of the changes. I would suggest that IEEE Micro not only regularly list standards projects but also state which version of a draft standard is current and where it can be found.

The byte ordering scheme of the approved S-100 standard is indeed more

**LETTERS cont’d on p. 14**

---

**ABOUT THE COVER**

The cover drawing shows Athens’ Tower of the Winds, built in about 50 BC to house an astronomical clock. The engineers are checking final construction details with a portable computer. Of course, no Greek engineer had such a computer available to him (much less one with a keyboard and LCD display), but Greek astronomers did have a sophisticated analog computer that they could use to predict celestial motions, as explained below.

Cover art and design by Jay Simpson.

Did the Greeks invent the portable computer? Yes, they did—in fact, the first known computer of any type. And its dimensions did approximate those of today’s portable “lap” machines. The Antikythera mechanism, as the machine has come to be called, was a portable analog computer built in about 80 BC. Its corroded fragments were discovered near the Greek island of Antikythera in 1900. Though it was not used to help construct the Tower of the Winds, it did allow its user to “dial up” detailed astronomical information.

The Antikythera mechanism and the Tower of the Winds are connected by the work of Derek de Solla Price, who in this issue begins our celebration of the IEEE’s centennial year with a history of pre-electronic calculating machines (page 22). Price, who was the Avalon Professor of the History of Science at Yale University until his recent unexpected death, deciphered the architecture of both devices. Their significance to the history of calculation is explained by Price in his article; their operation is described by Associate Editor L. Robert Morris in his tribute to Professor Price (page 15).
foreign markets. Government plays an important role in ensuring both. This country must pursue an aggressive trade policy aimed at achieving free and fair trade. We should negotiate in a tough-minded fashion to break down the trade barriers erected by our trading partners. Also, we should focus and streamline our export controls on high tech products, so that we can prevent trade-related transfer of militarily critical technologies, but at the same time make exporting easier for US companies.

Above all, high technology enterprises, as well as all businesses, can achieve their potential only within a good domestic economic climate. That means we must have lower interest rates and low inflation. People are unwilling to make investments, to make long-term commitments, or to borrow the funds needed for expansion in a climate of high interest rates and inflation. We must reduce significantly the substantial projected federal budget deficits for the next several years in order to remove the upward pressure on interest rates and inflation. We need a monetary policy that accommodates economic growth, a tax policy that encourages savings and investment, and the discipline needed to sharply curtail the growth of government spending.

High technology is perhaps our most valuable national resource. We must preserve it. However, innovation cannot be forced. It can only be fostered. It is fostered by creating an environment that emphasizes freedom of scientific and industrial activities and that offers incentives to the innovators, entrepreneurs, and investors who have the talent and resources to advance technology. It is fostered in a healthy economic environment and by trade policies that provide expanding opportunities for our technological products. Promoting such an environment should be a primary objective of America’s national industrial policy.

Ed Zschau represents California’s 12th District—which comprises much of Silicon Valley—in the US House of Representatives. First elected in 1982, he has served as a member of the Foreign Affairs Committee, as chairman of the Task Force on High Technology Initiatives for the Research Committee of the House Republican Conference, and as a member of the Republican Executive Committee.

For four years prior to his Congressional service, he was president of his own company, System Industries, a Santa Clara-based disk memory manufacturing firm which he founded in 1968. At the time of his departure, the company employed 550 people and had annual sales of more than $60 million. Earlier, he spent five years as an assistant professor at the Stanford Graduate School of Business and at Harvard Business School.

Zschau has been active in a number of volunteer projects in support of technological innovation, small business, and economic growth. He was co-chairman of President Reagan’s Business Advisory Panel, a member of the President’s Task Force on Entrepreneurship and Innovation, and delegate to the White House Conference on Small Business—all in 1980. Earlier, he served as a director of the American Electronics Association from 1974 to 1979 and as chairman of the board of that organization during 1978. He was also director of the American Council for Capital Formation from 1979 to 1982.

Zschau received an AB in philosophy, mathematics, and physics from Princeton, and an MS in statistics, an MBA, and a PhD in business administration from Stanford.

Letters cont’d from p. 11

processor-independent than that of the draft. I proposed the same scheme to the P896 committee in January 1983—that each byte lane be assigned to a byte address, leaving the weight of the byte unspecified. The lanes would have been called 0, 1, 2, and 3 depending on whether the address would end with 00, 01, 10, and 11. This would have corresponded to the odd and even lanes on the 16-bit S-100 bus. I thought that my proposal would resolve the problem, but I have been unable to convince the committee of it, since it was argued that my scheme again favors little-endian processors.

The reason for this is quite simple: practically all processors number the address lines the little-endian way, with A0 as least significant bit (LSB). As I pointed out in my article, there is a strong practical reason for this practice. If, for instance, an address is 24 bits wide, the LSB is A0 in a little-endian numbering but A23 in a big-endian one. If the address is extended to 32 bits, the same LSB is still A0 in the little-endian numbering but it must be renamed to A31 in the big-endian scheme. All lines must therefore be renumbered when one extends to a big-endian address. Since one lives with addresses of differing sizes, the logical choice is that the address be numbered in such a way that the least significant bits keep their names, i.e., in the little-endian way.

The problem arises with multiplexed, inconsistent big-endian processors like the Z8000. They are big-endians for 16-bit words, but they number the address in the little-endian way. Since the address appears on the same lines as the data, one must install a bidirectional byte swapper which either crosses the data byte lanes but not the address or the reverse.

Therefore, the Z8000 is penalized by the S-100 ordering scheme. The MC68000 is not affected since the crossing of the data lanes is static, except for the strange effect of having to cross the byte lanes, e.g., connect D0 of the processor to the bus line AD8.

So, the P896 committee decided that just four unassigned lanes with the names X, Y, W, and Z should be specified. I consider this a mistake: it only institutionalizes chaos. Since the inconsistency is caused by the Z8000, there is no reason to penalize all other processors.

So you see that even the current S-100 is not free of little-endian prejudices...

Now, about the TI 9900. It is no wonder that this processor can perform well on a justified bus: it does not make use of justification at all; all its transfers are 16 bits wide. As long as it only works with 16-bit devices, nothing bad should happen. But it cannot access an 8-bit peripheral, which is connected only to the justified byte lane and which responds to odd addresses. Hence, the processor board mentioned by Paterson does not comply to the S-100 specification, and it therefore cannot be fully integrated into a justified bus, since the very purpose of justification is to allow communication with 8-bit devices.

I encountered a similar problem when connecting our LSI-11/2 to a justified bus (an earlier P896). Everything worked as expected until we removed the byte swapper. To our surprise, the system kept on passing tests successfully. So we read the manuals more carefully and watched the traces on the scope. The LSI-11/2 was not doing byte operations but was using read-modify-writes on 16-bit transfers. The byte swapper had never been used since we didn’t have 8-bit peripherals.

Hubert Kirrmann
Brown, Boveri & Cie
Research Center
CH-5405 Baden, Switzerland