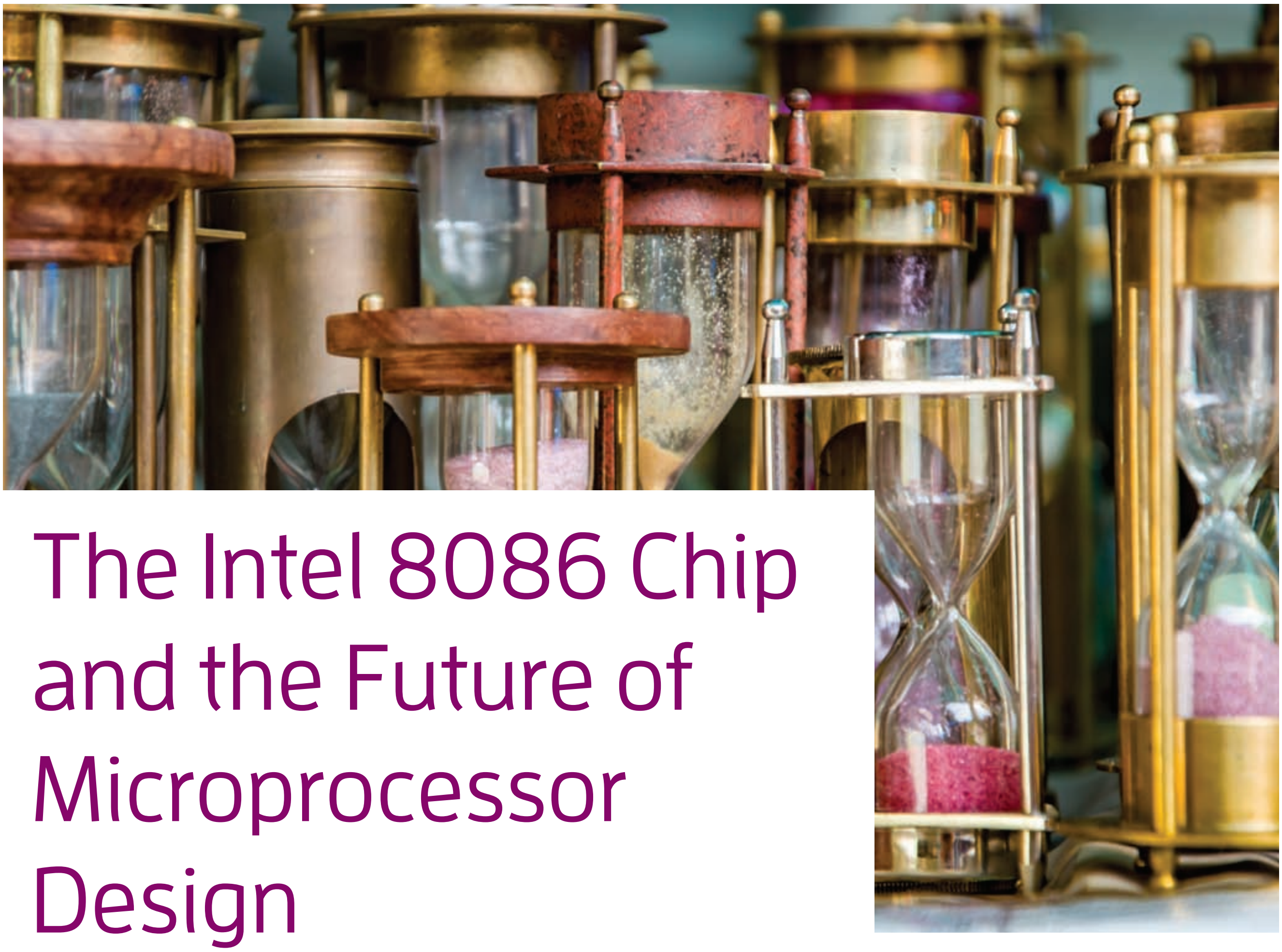
FROM THE ARCHIVES: *COMPUTER*'S LEGACY



The Intel 8086 Chip and the Future of Microprocessor Design

Stephen P. Morse

The author of a Computer article from 1978 reflects on his piece about the 8086 architecture that changed the future of microprocessor design.

FROM THE EDITOR

As part of our 50th anniversary celebration, I welcome you to the first installment of a new special feature focusing on influential *Computer* articles from the past 50 years. This month I highlight the article "The Intel 8086 Microprocessor: A 16-bit Evolution of the 8080," from the June 1978 issue. One of the original authors revisits the topic and discusses the original 8086 design's influence on the industry.
–Ron Vetter, Editor in Chief Emeritus

It's hard to believe that I've been asked to write an update to a paper that I co-wrote nearly 40 years ago (S.P. Morse, W.B. Pohlman, and B.W. Ravenel, "The Intel 8086 Microprocessor: A 16-bit Evolution of the 8080," *Computer*, vol. 11, no. 6, 1978, pp. 18–27). In those days, 8-bit microprocessors were the state of the art, and Intel was facing a real threat when Zilog Inc. implemented an enhanced version of Intel's flagship 8-bit 8080 processor. Intel was betting its future on a high-end microprocessor, but that was still years away. To counter Zilog, Intel developed a stop-gap processor and called it the 8086.

0018-9162/17/$33.00 © 2017 IEEE

It was intended to be short-lived and not have any successors, but that's not how things turned out.

The high-end processor ended up being late to market, and when it did come out, it was too slow. So the 8086 architecture lived on—it evolved into a 32-bit processor and eventually into a 64-bit one. The names kept changing (80186, 80286, i386, i486, Pentium), but the underlying instruction set remained intact. Because it was the brains inside the PC, this instruction set has been executed by more instances of computers than any other computer instruction set in history.

No one was more surprised by this turn of events than I was. I never could have predicted this outcome in the mid-1970s when I wrote the architectural specifications for the 8086. I've been asked what I would have done differently if I'd realized how significant this architecture would be for years to come. For one thing, I would have abandoned the backwards ordering of bytes in a word, changing history so we'd never have the need for terms like big-endian and little-endian. For another, I probably wouldn't have introduced a segmented architecture. And I likely would have gone with a more symmetric register structure. But the segmented architecture and the lack of total register symmetry were important back then because they were compatible with the 8080, which was a way for Intel to lock in their existing customer base.

One change I should have seen coming was the dramatic drop in memory costs. Back in the days of the 8080, the address space was only 16 Kbytes. This is because memory was expensive and no one in their right mind would have considered using that much memory with a microprocessor that was thought of as a toy. The 1-Mbyte address space of the 8086 was a far cry from 16 Kbytes, but it probably wasn't far enough (although it was substantially more than the 128-Kbyte requirement placed on the 8086). Today it's not uncommon to have memories in the terabyte range. Another effect of the low cost of memory is that program size is no longer an issue and the processor can have fixed-length instructions, allowing for smaller and faster instruction decoders.

It's interesting to look back and see how the x86 architecture has evolved from the original 8086 design. The drop in memory costs made it feasible for the processor to address a much larger memory space. But memory speed hasn't kept up with processor speed, requiring new architectures to incorporate elaborate caching schemes. The 8086 didn't allow an OS to protect itself against malevolent or buggy applications. Modern processors have user/supervisor modes, rings of protection, or similar devices, as well as paged virtual memories and support for virtual machines.

As instruction-set architectures have become frozen, innovation has moved to microarchitecture. For example, Intel's hardware executes x86 code by translating it into RISC microinstructions, dispatching these microinstructions out of order to multiple functional execution units operating in parallel, and then re-ordering the writing of results to registers and memory so as to hide the out-of-order execution. With elaborate microarchitectural designs such as this, Intel's engineers have been able to overcome many of the performance disadvantages of the 8086 architecture.

ARCHIVED ARTICLES

The impact and significance of the 8086 architecture and instruction set remains highly relevant across the computing industry. In fact, the original paper published in *Computer* remains very popular as indicated by the number of downloads it receives from the IEEE Computer Society Digital Library. All of the original articles mentioned in this special column are free to view at www.computer.org/computer-magazine/from-the-archives-computers-legacy.

ACKNOWLEDGMENTS

My thanks to Douglas Albert for his many suggestions on this piece.

STEPHEN P. MORSE is the architect of the Intel 8086 processor and creator of the "One Step" website (stevemorse.org) containing search tools used by genealogists. He has received numerous awards for his website, including a lifetime achievement award. Morse received a PhD in electrical engineering from New York University and has held numerous research, development, and teaching positions during his long career. Contact him at steve@stevemorse.org.