Reed Electronics Group

MICROPROCESSOR REPORT

www.MPRonline.com

THE INSIDER'S GUIDE TO MICROPROCESSOR HARDWARE

THE INTEL 4004'S 35TH ANNIVERSARY

Engineers Celebrate the World's First Commercial Microprocessor

By Tom R. Halfhill {12/11/06-01}

On November 15, 1971, Intel introduced the world's first standard-part microprocessor, the 4004. It was a four-bit CPU with 2,250 transistors, and it ran at a clock speed of 740kHz. Intel manufactured the chip in a 10-micron PMOS process on two-inch silicon wafers and furnished the device in a 16-pin ceramic dual-in-line package.

By its specifications, the 4004 may seem like ancient technology, a primitive relic from a bygone era. Yet it's only 35 years old—people born the same year don't even consider themselves middle-aged. The 4004 is a historic baseline from which the semiconductor industry can measure the astonishing progress of microprocessor technology.

At the time it appeared, of course, contemporary engineers viewed the 4004 not as a baseline but as a logical development in an evolutionary process that began with the invention of the transistor in 1947. The first discrete semiconductors led to planar transistors, integrated circuits, programmable logic arrays, multichip processing engines, and, finally, to the single-chip microprocessor.

Although the 4004 is widely recognized as the first single-chip microprocessor sold commercially as a standard part, it may not be the first microprocessor. Some historians point to the MP944 chipset in the Central Air Data Computer built by Garrett AiResearch in 1970. But the MP944 was a six-part chipset supplied only to the U.S. Navy for flight control systems in F-14 Tomcat fighters. The MP944 wasn't available to the merchant market. Indeed, it remained classified until 1997.

Another little-known contender is the AL1 processor designed by Four Phase Systems in the late 1960s. It shipped inside data terminals in 1969 and was described in the April 1970 issue of *Computer Design* ("Four-Phase LSI Logic Offers New Approach to Computer Designer," by L. Boysel and J. Murphy, pp. 141–146). However, the AL1 was never available as a separate product. Four Phase Systems, based in Silicon Valley, disappeared after being swallowed by Motorola.

Those quibbles aside, there's no doubt that the 4004 marked the beginning of the microprocessor industry. It's a direct ancestor of Intel's 8008 and 8080 microprocessors, which, in turn, are direct ancestors of Intel's x86—the dominant CPU architecture in today's PCs and servers. The 4004 provoked a top-to-bottom transformation of the computer industry that even the most visionary engineers didn't anticipate in 1971. (See the sidebar, "How Microprocessors Upset the Computer Industry.")

Moreover, the 4004 is a surprisingly modern design that differs from today's processors mainly by its relatively humble specifications, not by the broad concepts of its architecture or microarchitecture. Present-day engineers will find the inner workings of the 4004 comfortably familiar. (See the sidebar, "Analyzing the Intel 4004.")

Original 4004 Designers Speak Out

Four engineers played key roles in designing the 4004, and two of them made a rare joint appearance last month at the Computer History Museum in Mountain View, California. At a special 35th anniversary event sponsored by Intel, former Intel engineers Marcian E. (Ted) Hoff and Frederico Faggin described their work on the 4004. A third former Intel engineer, Stanley Mazor, was unable to attend the event but was represented by a video clip the museum had recorded

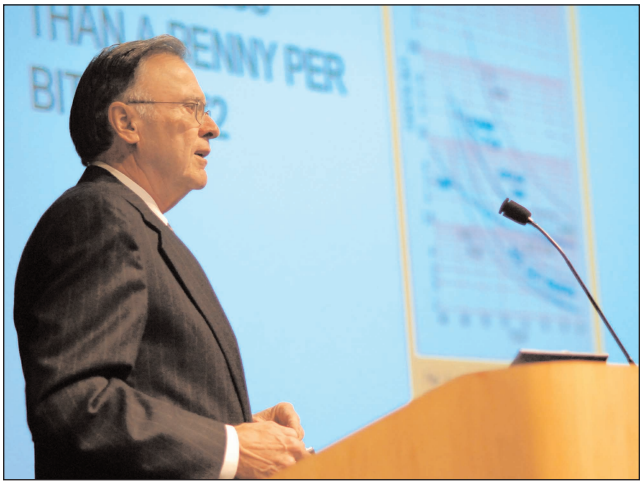


Figure 1. Ted Hoff, coinventor of the Intel 4004, speaks at the 35th anniversary event at the Computer History Museum in Silicon Valley. (Photo by *MPR*.)

earlier. The fourth engineer who played a role in the 4004 project was also unable to attend but sent a congratulatory message. He is Masatoshi Shima, formerly of Busicom, the Japanese company that motivated development of the 4004.

In 1970, Busicom asked Intel to design a chipset for a new desktop calculator. With Shima's input, Hoff and Faggin responded by designing not only a chipset, but also a general-purpose programmable microprocessor that later proved suitable for many other applications. *Microprocessor Report* recorded Hoff and Faggin at the Computer History Museum; below is our slightly edited transcript. Dave House, a former 22-year veteran of Intel, moderated the event.

Hoff: Back in 1968, I was at Stanford University...I got a call from Bob Noyce, saying that he was starting a new company, and would I be interested in possibly interviewing for a position there? This was before Intel had a place to meet, so I interviewed at his home and ended up becoming employee number 12 at Intel [as manager of applications research].

Intel was set up to do semiconductor memory. Up until that time, the major memory technology used in computers was magnetic cores. These were, typically, little donut-shaped pieces of ferrite strung on matrices of wires, typically by hand. The goal, we felt, would be to try to compete with them. They were several cents a bit, and they were making use of integrated technology to reduce the cost. But in about early 1970, the director of marketing and I put out an article in which we projected that we would get the cost of semiconductor memory down to less than a penny a bit by 1972. We figured at that point we would take over from magnetic core. Might point out we did overshoot that target in recent years—now we're down to about a millionth [of] a penny a bit, and still dropping.

Now, it was felt that it was going to take a while for the computer industry to accept this new concept of semiconductor memory, and what do you do for revenue in the meantime? And one of the things that semiconductor companies did was undertake custom work. And so we were contacted by a Japanese calculator company that sold calculators under the name Busicom. They had a fairly complete design for a family of LSI [large-scale integration] chips that they were going to use to actually make a whole series of different calculator models. In April of 1969, we signed a copy of an agreement—if anyone wants to see it, I have a copy of it—and in June of '69 three engineers came over from Japan to transfer their design. I got the job of being their liaison, which basically meant, [if] they got a problem, I'm supposed to find out who at Intel they should go to solve the problem.

Well, I was curious about their design, and I had been at those meetings in April, and I had an idea of what the target costs were for the set. Even though I was new to the semiconductor industry, the people there who were more familiar with things like packaging and chip design and so on were teaching me about the cost structures, and it began to look like it was going to be hard to meet the cost targets for this set, as it was presented to us by the Japanese.

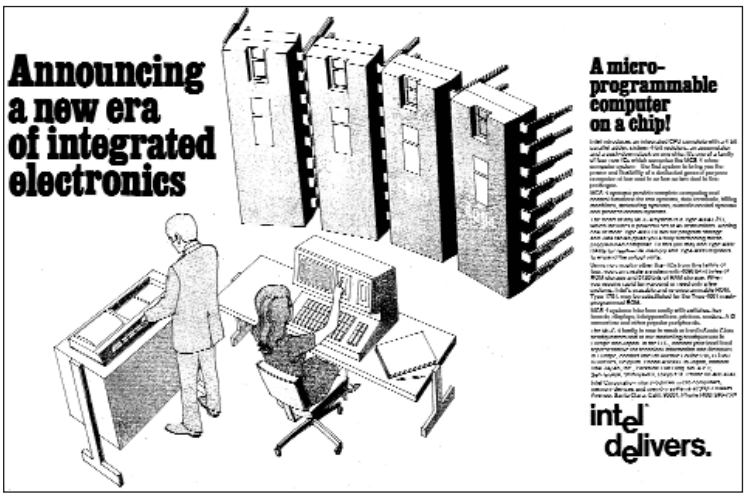


Figure 2. Intel placed this advertisement in trade publications in late 1971 to announce availability of the first standard-part microprocessor, the 4004, and its companion chips. (Source: Intel)

Intel Proposes an Alternative Design

Hoff: Encouraged by Bob Noyce, I started looking at another design, and that had a couple of changes. One, it was more general-purpose, more like a general-purpose computer than the original calculator set. And the original set was based on what's called shift-register memory, which uses six transistors per bit. We were working with DRAM and seeing that three transistors per bit—which is what our technology of the time used—we could possibly do a better job and reduce the costs somewhat. We made a formal proposal in September of '69, and we got the go-ahead from the Japanese calculator company to go ahead with

the Intel approach as of October. The contract was actually signed around February of 1970, and I believe I have a copy of that also here tonight, if anyone wants to see it.

Well, at that point, it was a concept—and I believe Dr. Faggin will tell you a little bit about how it went from concept to working parts. We had working parts early in 1971, and we were able to show the Japanese calculator company that they were working. It took a while, but we actually persuaded Intel's management that this should become a regular product for Intel. And considering that Intel had been set up to be a memory house, there was a lot of opposition. It was felt that if we go into the computer business, we are going to look like we're competing with our computer customers, and they may stop buying memory from us and go to our competition. So there was a lot of discussion, but finally the decision was made to announce the products. In November of '71 this ad appeared [see Figure 2]. People have asked us, did we know what we had? I don't consider this a very modest statement. It was pretty bold. *[Audience laughter]*

The 4004 came out then [November 15, 1971]. I have copies of a price list that dates back to September of 1972, so this is about a year after it came out. And by the way—if you can see the fine print—but the 1103, which was 1,024-bit dynamic RAM, was selling for well under $10 if you bought a quantity of 25 or higher. So we had made the penny-a-bit target in 1972.

The 4004 in single quantities was selling for $60, and in 100-piece quantities for $30—relatively inexpensive for a computer when you compare it to what we had dealt with maybe a decade before. *[Editor's note: Adjusted for inflation, $60 in 1971 is about $308 in current U.S. dollars, and $30 is about $154.]* In 1961, while at Stanford, I had the opportunity to work with an IBM 1620 computer...The performance of the 1620 was about 2,000 five-digit decimal additions per second. That's about the same performance that a 4004 system would offer, in the same order of magnitude. Now the 4004 was, you know, just a CPU, and you had to add other things to it, but consider that you got the CPU for $30 if you were buying a number of them, or $60, buying just one. For the IBM 1620 back in 1961, a decade earlier, we paid $2,500 a month to lease that machine. And that was at Stanford—we were getting a 60% educational discount. So other people would have paid two and half times that.

In addition to the processor, we also offered design aids. And we called these the SIM series of boards, and one of them was a SIM-402. It was a single-board computer, and you could plug in up to 16 of the 4002s, which gave you 1,284-bit quantities. They could be thought of as either binary or binary-coded decimal digits. You could plug in 16 EPROMs for up to 4Kbytes. So that cost—fully loaded in small-quantity purchases—would have cost you about $2,400, one-shot cost. To put it in perspective, compared to the [IBM] 1620, it had maybe only a fifth of the amount of memory. But here you paid for it once, not every month. Performance was similar.

Now, people say, "Well, what about the personal computer?" We were not ready for the personal computer at that time. In other words, there were a lot of other things that were needed, and memory was still a major factor. When you consider the major difference between the cost of that board, loaded, and the CPU, it's the cost of the memory that went onto it. So it took a while for other parts of the system, for memory, to come down farther, before the personal computer [was practical]. Our market we saw was what we now call embedded control. I might point out we have a SIM-402 system sitting on the table over here [see Figure 3].

In 1972, we tried to interest a different kind of manufacturer in the possibility of using microprocessors. I believe the company was Magnavox, which made an analog videogame [system]. And we thought there ought to be a business for microprocessors in videogames. We talked with them about it, and they were not interested. The microprocessors were way too expensive. It wasn't long after that, there was quite a market there. If you want to see the demo afterwards, we have a demo of a working SIM 402 with an original 4004 in it, and this was a replica of a demonstration that we did back in 1972.

So that's where we were in '72. Of course, a lot has happened since then. There were also many steps that it took to make the processor happen, and Dr. Faggin will tell you about some of those.

Frederico Faggin Builds the 4004

Faggin: Thank you. It is really a pleasure to reminisce about the microprocessor. Let me start with the MOS technology in 1968. Basically, in production, there was a high-threshold metal-gate MOS technology. It was pretty unreliable, relatively slow technology. To make complex random-logic chips we had to use four-phase designs, and really there was not much that could be done still with the MOS technology in '68.

However, there were two key technologies in development at that time. One was low-threshold metal-gate MOS with ion implantation—that's the technology that MOS Tech developed that was really the mainstream technology for a number of years—and self-aligned silicon-gate MOS technology that was developed at Fairchild. I was in charge of that project, and that project was completed in 1968. A paper was given at the International Electronic Device meeting. This paper described the technology and also an integrated circuit,

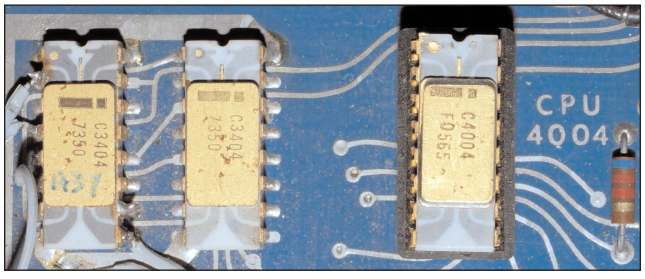


Figure 3. This is an original Intel 4004 microprocessor on a SIM-402 single-board development system that Intel sold to customers in the 1970s. Notice the gold pins. (Photo by *MPR*.)

a commercial integrated circuit. That was the Fairchild 3708. This technology was then adopted by Intel and became the basis for semiconductor memories in the microprocessor.

Now in April 1970, I joined Intel. The day that I joined Intel, Stan Mazor showed me a block diagram and the basic specifications of a four-chip set that was to be the Busicom project. There was the four-bit CPU and the basic architecture. And the instruction set was pretty much completed. There were a few loose ends, which I completed later on, but basically the specification was done.

The design of the chips was supposed to have started in October when the agreement between Busicom and Intel occurred, but no work was done since October. So I had the honor of being six months late the day I started on my project. On top of that, the following day, Masatoshi Shima, the Busicom engineer, was arriving from Japan to check the designs and verify the logic design of the 4004, which was supposed to be in layout. In July 1970 was when the entire chipset was supposed to be completed. *[Editor's note: Faggin was the first in a long line of microprocessor designers who got an impossible schedule dropped on his head by management and marketing.]*

Shima arrived, and said, "I'm here to check. Where is logic?" And I said, "Uh...oh...mmm," and I gave him what I was given, the block diagram and this stuff. And he said, "No good! I had this! This is only idea. I want logic!" And I said, "I don't have any logic." [Shima said] "You bad! You bad!" And I said, "I'm just arrived here! I just was hired yesterday!" [Shima said] "You late!" *[Audience laughter]*

And so I was baffled. Of course, at that point there were a number of very agitated phone calls between Shima—he was in my office; Stan Mazor, myself, and Shima were in the same office—and Japan. I could tell by the tone of the conversation they were not very gentle phone calls. And basically, in the meantime, I tried very hard to figure out a time schedule that would really minimize the delay that was incurred by Intel.

So I basically put a schedule together that gave December 1970 as the date at which I could give all four chips, samples for all four chips. This was really working very hard, because nine months for four chips, of which three were state-of-the-art chips, was really pushing it. Of course, I had to go do all this logic, the circuit design, layout, ruby cutting, all this stuff.

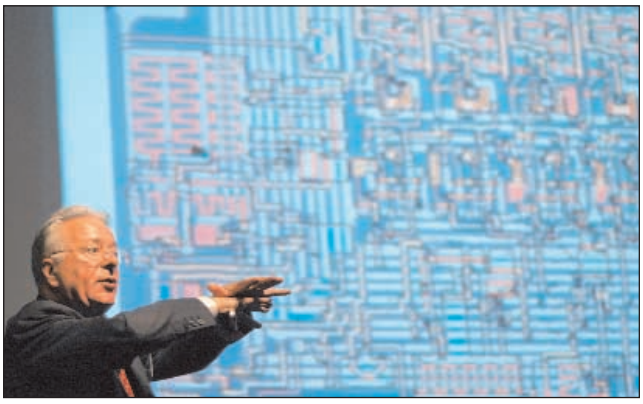


Figure 4. Frederico Faggin describes the 4004 layout during the 35th anniversary event at the Computer History Museum. (Photo by *MPR*.)

All 4000-Family Chips Designed at Once

Faggin: I decided to stagger the chip development, starting with the 4001 [a 256-byte ROM]. While the layout of the 4001 was going on, the design of the 4002 [a 320-bit DRAM] would occur, and so on, so that by December we would have all four chips [including the 4003 I/O chip and 4004 CPU]. But to do that, I needed an engineer to help me, and also adequate layout and technician support. I was by myself. I had no engineers, no layout people—that's it, I had to do it by myself. So I asked Shima to convince his management to let him stay for a while until I could find an engineer. If he wanted to [meet the deadline of December] 1970 to get the chips, that's what was required. And management decided to let him stay, so he stayed until October 1970.

In 1970, Intel had no expertise in random logic design. There was no methodology and no infrastructure for random logic design. You know, companies like Fairchild or TI, they had specialized cells, they had circuit designs, they had computer programs to help with logic simulation and chip design and so on. None of that was available at Intel. On top of that, the random-logic design with silicon gate required a different style than metal gate, so basically I had to start with developing the basic methodology. I did that by deciding that I was going to use a two-phase design, but that required bootstrap loads. In those days, bootstrap loads were considered unworkable with silicon gate, for a variety of reasons I don't have time to go through, but I made it work.

Then I decided to use buried contact. That was an invention that I made at Fairchild a couple years before, which allowed [you] to make direct contact between polysilicon and junctions, so that you could have effectively two layers of interconnections, making very high, very dense circuits. I also devised a graphic design that would allow you to size transistors very rapidly, because I didn't have time to do any simulations.

I designed all the basic building blocks that we were going to use for the family, so that we had that as sort of building blocks. And then I combined the logic design and the circuit design into a single document. In 1969, what people would do, they would do the logic design, then transfer that into a circuit design, then from there into the layout. Of course, every time you transfer from one set of documents to another, you can make mistakes. So I decided, to avoid mistakes, I would combine logic design and circuit design, and do that in a single sheet in a single design, but also taking into account the layout, so the transistors would be put as close as possible to their physical location in the layout. Then, of course, a design for testability, which was clearly very important, and I tried to do that by minimizing computer simulation at the time, because, as my boss told me, "Don't use computers! Very expensive!"

So the result of that is in October [1970], on schedule, the 4001 was designed and came out and worked first time. This is a 1Kbit, metal-mask programmable [chip]. This part is the...specialized I/O [chip]...Here is the four-bit bus drivers for the data bus. They need to drive 150 picofarads, so they had to be pretty hefty transistors, and the rest of it is pretty much a ROM. This is the 4003. This is the simplest of the four chips. This has a 10-bit shift register; this was done in our spare time. And this is the 4002; 320 bits of dynamic RAM, three-transistor dynamic RAM, self-refreshing; the refresh counter is here. All the I/O port and control and specialized timing is here. The rest of it is similar to a typical memory.

Faggin Describes the 4004 Layout

Faggin [referring to a projected slide of the 4004 layout, seen in Figure 4]: And, finally, the 4004. Here I am showing an example of the type of document that contained the circuit design and the logic design that I mentioned earlier. Here, for example, this is the instruction register; this is the instruction decoder; the encoder; this is the control logic for the index register. Notice that all the transistors have a little number you can actually not read but that is the size of the transistor. That's when I used the graphic design to size all those transistors without having to do any simulation.

Here, by the way, that's the four-bit bus that goes inside the chip and controls the rest of it. Here's an example. This is just a corner of the chip. You can see here, for example, those are the big data-bus drivers. These little pink squares or rectangles, those are the capacitors of the bootstrap loads that were absolutely essential to get the performance out of the chips. Here, for example, you see every contact, when you see metal running over the logic, so that, in fact, you have logic underneath and metal connecting on the other layer, therefore achieving about twice the density of metal-gate technology and four to five times the speed of conventional metal-gate technology.

And finally here is the entire chip. [See Figure 5.] Here's the instruction register, the instruction decoder, the control logic of the two memory blocks. This is the index register, 64 bits [sixteen 4-bit registers]. This is the stack pointer, the program counter, the incrementer-decrementer is here, this is the ALU, the control logic of the ALU, this is the timing, this is the control logic for the external memories, and this piece also. This chip was 3×4mm. And those are my initials, by the way: "FF."

So basically I was pretty much on time, probably a couple weeks late, but late December 1970 I received the first wafers for the 4004. I load them up and nothing happened. I went to another die, and nothing happened. And I was thinking, "Oh, my God. How could I have screwed up so bad?" You know, the other three chips worked the first time, except the 4002 had a couple very simple mistakes that were immediately cleaned up.

Basically, I found out that [for] one of the masking layers, the buried contact was left out of the processing, so a good 30% of the gates were floating. That's why nothing was happening. So finally I got the new silicon about a month later, and I received the wafers toward the end of the day. I load them up and started testing, and everything seems to be working. I kept working and working and working. Eventually, by four o'clock, I was really tired, but I was also very happy because everything I tested was working. So I went home, and my wife was waiting for me, and I said, "It works!" And that was the night that the 4004 was born.

Busicom Builds the Calculator

Faggin: A few weeks later, I found out there were a few minor mistakes, which were corrected. Eventually, in March 1971, we had fully working 4004s. Now, in the meantime, Shima went back to Japan in October [1970] and worked from October to January to develop the firmware of the calculator. He gave us ROM patterns. Masks were supposed to be made for the 4001s in February, so that by March, all the chips ended up in Japan, and the first prototype of the calculator ended up working completely.

This is the prototype of the calculator. [See Figure 6.] The PC board is underneath this printer here—this is a Seiko printer. There are about ten 4000-family chips underneath.

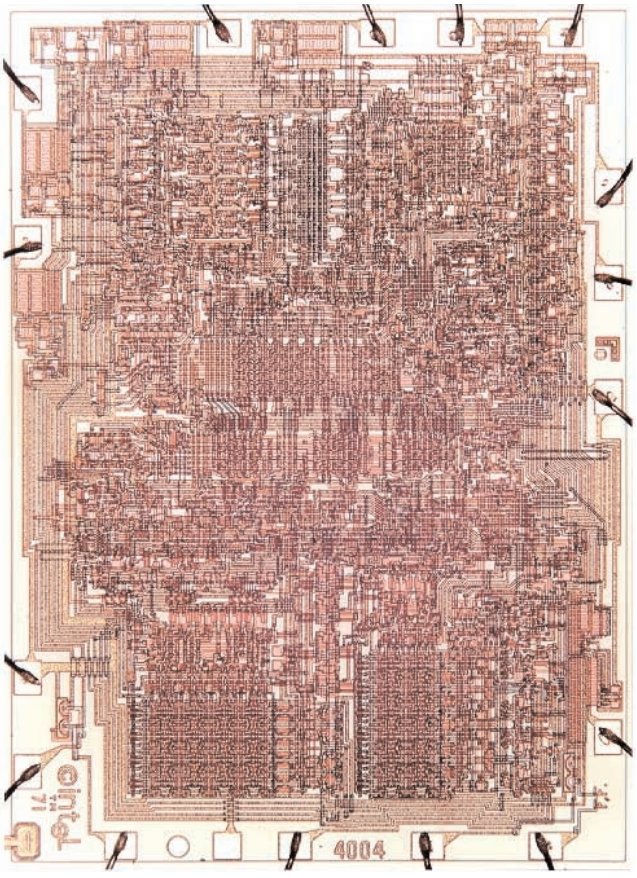


Figure 5. Intel 4004 die photo. Frederico Faggin's etched initials are visible in the lower-right corner of this first-run sample die. In actual production versions of the 4004, the initials are located near the center of the die. (Source: Intel)

This was a personal gift to me by the president of Busicom, Mr. [Yoshio] Kojima, because basically I worked 80 hours a week to make up the delay of six months that was incurred in the project. And by the way, in the middle of 1971, Texas Instruments announced a CPU-only chip that was actually the TI version of the 8008, and that was announced only a few months after the 4004 worked. So timely execution was the key to get the first microprocessor at Intel. By the way, this [prototype Busicom calculator] is now my gift to the Computer History Museum. *[Audience applause]*

So from the end of March until June we completed the characterization. The transfer to production was a lot of work to produce the chips. Then in early '71, I decided to design and build a tester to test wafers of the 4004. I decided to use the 4004 as the controller—as the microcontroller, you would say today—of the tester, for three reasons. Number one, I thought it could do the job and make my life easy. Number two, I could show Intel management that you could actually do an application that was a control application using the 4004 instead of calculators, which was the original aim of the family. And number three, I wanted to show an outside engineer, like a customer, what it could do to develop a system using the 4000 family.

So with that, I used the 1702 EPROM chips that were a new invention of Intel, the programmable electrical memories. With that project working, that gave impetus for all of us, including Ted [Hoff], to convince management to really commercialize the 4000 family. But we couldn't do that because Busicom had exclusivity for the 4000 family. So we had to have some kind of recision of that exclusivity. I was happy to find that Busicom was in difficulty, by talking to Shima, and I proposed to Noyce to actually give a price concession to Busicom in exchange for the exclusivity. *[Editor's note: Intel reportedly bought back the rights to the 4004 in 1971 for $60,000, or about $308,000 in current U.S. dollars.]*

Figure 6. This is the desktop calculator that Busicom built with the Intel 4004 microprocessor and the other 4000-family chips. Frederico Faggin has donated an original prototype to the Computer History Museum. (Photo: Intel.)

Intel Introduces the 4004

The bottom line is that by the middle of the year [1971], the decision was made by management to release the 4000 family under the name MCS-4, and the product marketing position was created to handle the product. Hank Smith was the manager for the product marketing organization. Finally, the announcement was made in November. You've seen this before [Figure 2]. This is one of the few ads where the hoopla here was actually truth in advertisement.

There were a couple of papers published soon after the announcement...*Electronics* [magazine] in April [1972] published another article by myself and Ted. So this was the beginning of the marketing of the product.

Now, I want to acknowledge many other contributors to the MCS-4 in addition, of course, to Ted Hoff and Stan Mazor. Masatoshi Shima, we talked about his people. Alfina Yun Feng* helped me in the [characterization] and test programs development toward the end of the year and the early part of '71. Paul Metrovich* and Charlie Korbin* helped with the testers implementation, the characterization work, and so on, starting in the October/November timeframe.

Rod Sayer* was the first draftsman that I hired. He had never designed a chip—he was actually a mechanical draftsman from Lockheed, and I had to teach him how to design chips. Barbara Mannis* and Julie Hendrix,* however, had done designs before, so they were quite helpful—not that Rod wasn't, but he actually took a few months to learn. And Hank Smith, because he was the first product marketing manager that made a tremendous contribution to the success of the microprocessor.

The rest is history. Thank you! *[Audience applause]*

Audience Question-and-Answer Period

Question: I was wondering if you could tell us about the work they had to do with the process engineering team at the same time that you were doing all this design.

Faggin: The process development was already done at Intel, so basically I used an existing process. There was no additional work that was required. In fact, the idea of the buried contact was already basically lifted from Fairchild and used in the 1103 design. So I used that same technology that I had developed a couple years later at Fairchild.

Question: I'm Karen Tucker with the Computer History Museum, and this question is for Tim [McNerney]. Tim, can you tell us something about the new exhibit at the Intel museum? *[Editor's note: McNerney conceived and curated the Intel 4004 35th-anniversary exhibit at the Intel Museum in Santa Clara, California. See the "For More Information" box.]*

McNerney: Yes, as I mentioned, this is the core of the exhibit. It's a fully functioning replica [of the 4004]. There's also an interaction area where people can see what it was like to use the calculator software and also see what was inside, how things were going on inside, to see the registers, to see how the program takes through a flowchart, etc. And I welcome everybody to go visit this.

Question: I have two questions. First, has anyone tried to write the 4004 in Verilog and resynthesize it to compare the quality of design?

McNerney: I have to take exception to "compare quality of design," but Fred Huettig did take the schematics and derive a Verilog simulation to do a design verification on that. Yes. Does that answer your whole question?

Question: So what was the number of transistors?

Huettig: Actually, there were two different Verilog implementations that I made. One was straight from the schematics, converted at the transistor level. I found 2,304 components, and I think 1,843 were FETs that were used for switching and not loads, or caps for the bootstrap loads. But more recently I did a Verilog simulation at just the RTL level, and I'd have to look to see how many gates that took. It was about 10% of an Altera Cyclone II-8, so that makes about 800 gates. That includes all the memories and the ROMs. *[Editor's note: After reviewing this transcript, Fred Huettig clarified that he meant to say 800* logic cells, *not 800* gates. *An Altera programmable-logic cell may perform the functions of about 1 to 12 ASIC-equivalent gates, depending on the efficiency of the synthesis. His simulation includes the memory in the 4001 and 4002 chips, but his 800-cell estimate does not. Huettig used an Altera Cyclone II-EP2C8, an FPGA with 18KB of RAM and 8,256 cells. Altera's timing-closure tools indicated that, without optimizations, the simulated 4004 would run at 100MHz.]*

Question: Is there at least one instruction that survived, or the instruction opcode that survived until today?

Faggin: Of the 4004? Opcode? I don't think so.

McNerney: NOP. *[Audience laughter]*

Faggin: Different opcode! But a different opcode for NOP, perhaps.

Question: One of the other big problems with the 4004 as a user and implementer was it had no interrupt...

Faggin: Yep.

Question: Hi, I'm Bob Zeidman with Zeidman Consulting. So I have two questions: The first one is, I think Dr. Faggin, I think you pointed out a section that you said was the automatic refresh circuitry for the DRAM. And if I understand that correctly, if it automatically refreshed the DRAM, why was that never incorporated in standard DRAMs after that? And the second question is, I would like to know what you think is the worst part of the 4004 architecture that you wish you had done differently?

Faggin: Uh, that last question is for Ted. *[Audience laughter]*

Zeidman: My career would have gone a lot smoother if the DRAMs had automatic refresh on them, because that was the biggest problem I ever had to do, designing DRAMs, figuring out how to get the refresh signals in there.

Hoff: Well, in the case of the DRAM, first of all, we took advantage of the fact that there was a time when we knew the DRAM was not being used, when it [the 4004] was fetching instructions. That's not the case in the normal, non-Harvard architecture computer. So that made it more difficult to do the same kind of thing in a DRAM for regular applications. Although, I believe around 1975, another semiconductor company did file for patents on on-chip refresh with DRAM, and in the file history you'll find that there was a reference to the 4002. But the statements that the applicants made was that they had no idea what the 4002 was! So, in fact, they got the patent issued, even though the 4002 was a DRAM with on-chip refresh. I'm sorry, I missed the second...

Zeidman: In the architecture, which part of the architecture were you least proud of, that you wish you had done differently?

Hoff: Least proud of? Well, probably one is the addressing in there. Initially, we weren't setting out to make a truly general-purpose computer. The job was to simplify the calculator architecture, and so the calculator architecture dictated the number. There were a few things that really kind of stood out. They were talking about having displays with up to 16 digits. We had, you know, four bits for binary-coded decimal arithmetic. Well, if you are making a binary machine instead of a BCD machine, four bits will select one out of 16 items. So let's take a look at a binary machine. Well, you've got a binary machine and you add two binary-coded decimal digits, you end up with a valid binary result, but it's not a valid binary-coded decimal result. So we put in the decimal-adjust accumulator instruction, and now you had a machine that could either be decimal or binary, and it could actually go between the two as needed, so that was a nice feature.

For More Information

Intel's website has a great deal of information about the 4004 microprocessor, including downloadable images of the chip, layout masks, a data sheet, and a user's manual. See:

- www.intel.com/museum/archives/4004.htm
- www.intel.com/museum/archives/4004ip.htm

The following website, not affiliated with Intel, has been compiled by Tim McNerney. McNerney conceived and curated the Intel 4004 35th-anniversary exhibit at the Intel Museum in Santa Clara, California. He led a team of software and hardware engineers that built a fully functional 130x replica of the 4004. This team wrote Verilog models of the 4004, created a verified set of schematics, and even wrote an animated 4004 simulator in Java:

- www.4004.com

Another non-Intel website has detailed information about the 4004 and one of its designers, Frederico Faggin:

- www.intel4004.com

Additional information about the 4004 35th anniversary and many other historical events is available at the Computer History Museum and its website:

- www.computerhistory.org

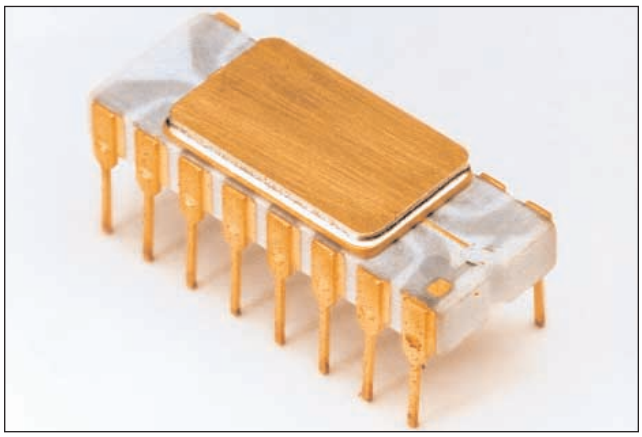


Figure 7. Intel's most widely reproduced photograph of the 4004 makes it appear the package has a cap made of wood, but it's actually a ceramic-and-gold package. The resemblance to wood is an artifact of the color correction. (Source: Intel.)

Hoff: The other was the lack of interrupts. We just weren't ready to put the cost of that in. It was quite a while before we even did an estimate of how many transistors [the 4004 would require], and our first estimate was just under 2,000, somewhere around 1,900. I'm amazed that it came in as close as it did, when you consider how rough our first estimates were. Because I was not an MOS designer, and, as Dr. Faggin has pointed out, there weren't many people there [at Intel] who were MOS logic designers. They were all looking at memory circuits, and you have maybe two or three circuits—a decoder and a memory cell—and the whole circuit is made out of, you know, those two elements, where you need things like flip-flops and timing and all the rest that goes into a logic circuit. So I think we were pretty lucky.

Hoff: We did put in the remnants of an interrupt into the 8008. In fact it came about when one time Stan Mazor and I were sitting around saying, what could we do to make the 8008 interesting, and we raised the issue of interrupt. We didn't want to pay the cost on the chip, but we asked the question, "What would you put on the chip to be able to add it as an afterthought?" And we came up with a solution, which was confusing initially. In fact, it was confusing to some of our competitors, who filed for patents on the design and got it wrong because they didn't understand it.

Hoff: We decided we could do a call instruction and force it [the interrupt] instead of letting the processor fetch an instruction from memory. We would actually force a call to a subroutine. And then it occurred to us, if you do that, and the call will put the program counter on the stack, but unfortunately you've bumped the program counter ahead as it fetches the call instruction, and what you push on the stack means that you've missed some of your original code. Solution: stop advancing the program counter when you recognize an interrupt. We added essentially one, you might say, one gate. When you acknowledge an interrupt, you don't advance the program counter. And that was the hook we put in to allow interrupt later on. There were other things we should have done, because it was not easy to service an interrupt in an 8008, even with what you could do later on with it.

Faggin: I want to add one thing, because as you probably know, all the chips were packaged in 16 pins, 16-pin packages. We threw away about three times the performance by going to a 16-pin package instead of, say, a 40-pin package, which in those days was also quite possible. That was not Ted's fault, actually; that was Intel's management that had a phobia about, you know, big-pin-count packages. We were forced to use 16 pins, when in fact, as I said, we threw away three times the performance. In other words, we had 10.8 microseconds per instruction, instead of, say, three microseconds per instruction, in those days. Now what saved us was the fact that silicon gate technology was so much faster than metal gate technology. Even with that, we were competitive with metal gate chips that came out later on using four-phase designs, like the Rockwell EPS-4, for example. So if I had to say what was a major flaw in this family? Using a 16-pin package for the CPU.

McNerney: I'd like to sort of play devil's advocate here. One of the things that we really liked about this design was that you can run this four-bit bus through all of the chips and it makes the PC board layout very easy. This is a very elegant chipset. When you look at later chipsets, which required decoders and bus multiplexing logic, this is just wonderfully elegant, even at that level.

Faggin: Yes, it was elegant, but it was not efficient. *[Audience laughter]*

McNerney: OK, I can't argue with that.

Hoff: I beg to differ. It was necessary to get it through Intel. *[Laughter]*

McNerney: Oh, I agree with that!

Question: Hi, I'm Tom Halfhill with *Microprocessor Report*... The most commonly reproduced photo of the 4004, which is on the Intel website, makes it appear that the package is made of *wood.* So what's that all about? [See Figure 7.]

Faggin: ...The package is made of ceramic. It may look like wood to you, but it actually is ceramic packaging. That's the typical package in those days.

Hoff: There were concerns in the early days about silicon gate MOS. I believe there were certain steps that were done to make the oxide layers smoother—the high doping of phosphorus in the glass—and the aluminum over that could be etched by phosphoric acid if moisture got in the package. There was concern whether plastic packages would be moisture-resistant enough to be able to have reliable silicon gate circuitry, and so one of the reasons why the initial packages were ceramic. Later on, it was found they could deal with that.

Faggin: Another thing was the power dissipation. It was 700mW, and there was a concern that 700mW was too much for plastic.

**MPR has been unable to confirm the spellings of these names.*

Analyzing the Intel 4004

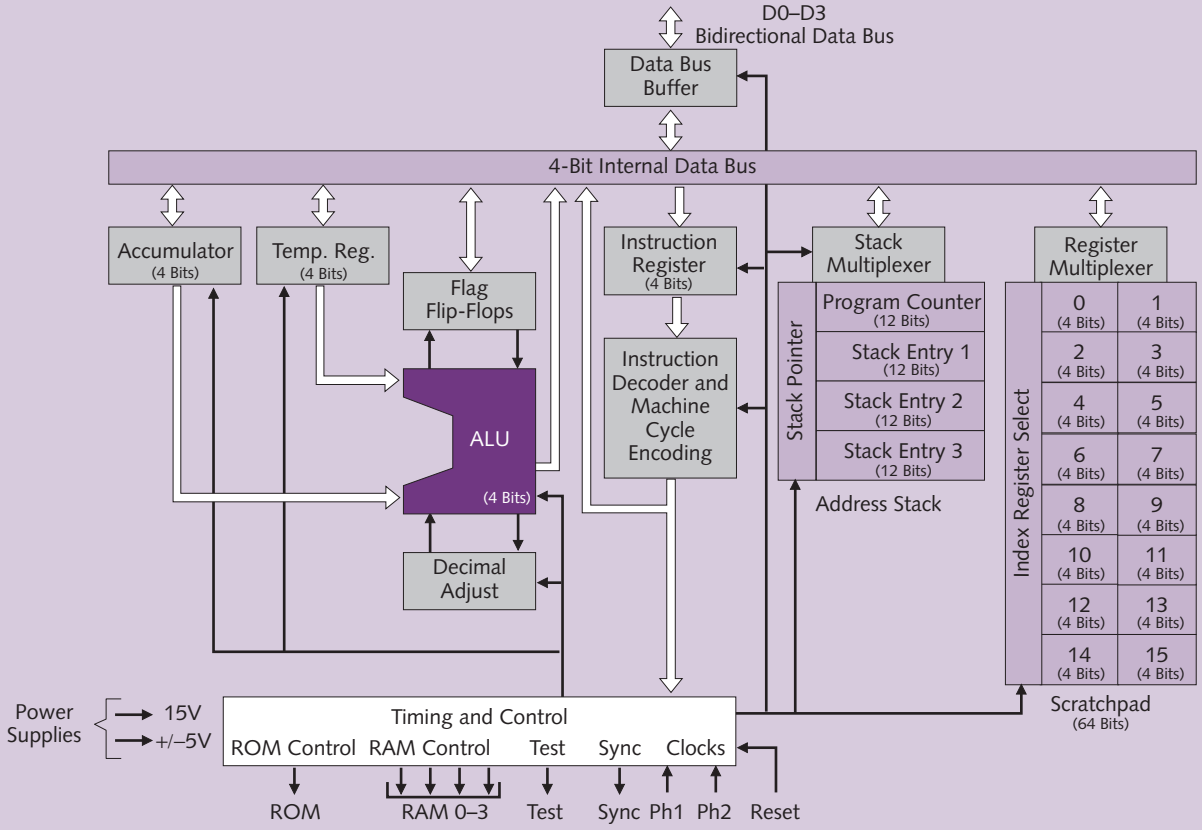
Despite its age and simplicity, the Intel 4004 microprocessor is not starkly primitive. Its basic architecture and microarchitecture are remarkably similar to today's microprocessors. The 4004 will seem familiar to anyone who has worked with small embedded processors, and it isn't radically different from the powerful 32- and 64-bit x86 processors in the latest PCs and servers.

True, the 4004's specifications are humble. It's a four-bit machine with one four-bit ALU and 46 instructions—most of them eight bits long, plus a few 16-bit operations. But it's relatively generous with registers, even compared with eight-bit processors introduced years later. As the block diagram shows, there are 16 general-purpose four-bit registers, an eight-bit instruction register, an eight-bit temporary register, a four-bit accumulator, a 12-bit program counter, and a memory-address stack with three 12-bit entries. Surprisingly for a pre-RISC design, the 4004 has a Harvard memory architecture, segregating program code and data in separate memories. It can directly address 4KB of instruction memory in ROM, but only 640 bytes of data RAM per bank.

Off-chip communications are necessarily sparse in a processor with only 16 pins. The bidirectional I/O bus uses four pins. Four control pins select the RAM data banks, and a fifth control pin sends the select signals to program memory in ROM. Two pins receive the two-phase clock inputs, and another pin transmits a synchronization signal to ROM or RAM at the beginning of each instruction cycle. The remaining four pins are for test, reset, and power (15V, ±5%).

As codesigner Frederico Faggin has noted, the 16-pin dual-inline package is the 4004's greatest limitation. In 1970–71, when Intel designed the 4004, the company's management was averse to relatively costly packages with more than 16 pins. Consequently, the 4004 must use the same four-bit I/O bus to send all 12-bit addresses to external memory, fetch all instructions from external ROM, load all data from external RAM, and write all results to external RAM. (Of course, the 4004 has no caches—but then, one of today's most popular embedded processors, the ARM7, has no caches, either.)

To a large degree, the 4004's multiplexed bus defeats the advantages of the Harvard architecture, because the



Intel 4004 block diagram. *MPR* has redrawn this diagram from a reproduction of a vintage (and not very legible) Intel 4004 data sheet, making only minor modifications.

processor can't simultaneously fetch instructions and data from their separate stores. The 4004 requires five clock cycles to fetch an eight-bit instruction: three cycles to send the 12-bit address to memory, then two cycles to actually load the instruction. Fetching a 16-bit instruction requires ten clock cycles: six cycles to send two 12-bit addresses to memory, then four cycles to load the instruction. And, of course, none of these operations is pipelined.

All together, the 4004 requires eight clock cycles to fetch and execute an eight-bit instruction, and 16 clock cycles to fetch and execute a 16-bit instruction. Original 4004 documentation refers to the time required to execute an eight-bit instruction as an "instruction cycle" and says the duration is 10.8 microseconds. However, the documentation doesn't specify the processor's actual clock frequency. This omission has caused much confusion. For years, Intel's website has stated that the 4004 ran at 108kHz. *MPR* believes the actual clock speed was about 740kHz.

We base our calculation on a vintage 4004 data sheet and user's manual that clearly define an "instruction cycle" as eight "clock periods." (That's for an eight-bit instruction; the same documents say that a 16-bit instruction requires two instruction cycles consisting of 16 clock periods.) In other words, in the context of the 4004 documentation, an instruction cycle isn't the same as a clock cycle. The same vintage documents specify a clock period as 1.35 microseconds, which is consistent with a 10.8-microsecond instruction cycle divided by eight clock periods. Therefore, we calculate the 4004's actual clock frequency as 740.74kHz (1,000 divided by 1.35)—quite a bit faster than Intel's official 108kHz.

The 4004's instruction set is a little quirky, but not alien. Using original documentation, *MPR* created the instruction-set table accompanying this article. We count 46 instructions (unique mnemonics) representing 239 assigned opcodes. The only unassigned opcodes are hexadecimal 01–0F plus FE and FF. Other sources cite slightly different numbers of instructions and opcodes, probably because of small errors we discovered in the original documentation. (One table omits CMA, and another omits NOP.) Although 46 instructions is a tiny set by the standards of a modern high-end processor, it's about average for a small embedded processor, even today.

During the recent 35th anniversary event at the Computer History Museum, a questioner asked if any 4004

Instruction*	Description
	Arithmetic and Logic Instructions
NOP	No operation
INC	Increment register RRRR
ISZ*	Increment register RRRR and go to ROM address if result <> 0
ADD	Add register RRRR to accumulator
SUB	Subtract register RRRR from accumulator
CLB	Clear both (accumulator and carry)
CLC	Clear carry
IAC	Increment accumulator
CMC	Complement carry
CMA	Complement accumulator
RAL	Rotate left (accumulator and carry)
RAR	Rotate right (accumulator and carry)
TCC	Transmit carry to accumulator and clear
DAC	Decrement accumulator
TCS	Transfer carry subtract and clear
STC	Set carry
DAA	Decimal-adjust accumulator
KBP	Keyboard process (convert 1-of-4 code to binary code)
DCL	Designate command line (select from multiple RAM banks)
	Branch and Jump Instructions
BBL	Branch back (down one stack entry) and load 0000 into accumulator
JCN*	Jump to ROM addr if condition code true
JIN	Jump indirect to register address
JUN*	Jump unconditional to ROM address
JMS*	Jump to subroutine ROM address

Instruction	Description
	Data-Transfer Instructions
FIM*	Fetch immediate from ROM data to index registers RRR
FIN	Fetch indirect from ROM via index registers RRR
LD	Load accumulator from register RRRR
XCH	Exchange register RRRR with accumulator
LDM	Load data 0000 to accumulator
	I/O and Memory Instructions for 4001, 4002, 4008, 4009, and 4289 Companion Chips
SRC	Send address in registers RRR to ROM and RAM
WRM	Write accumulator to previous RAM location
WMP	Write accumulator to previous RAM output ports
WRR	Write accumulator to previous ROM output ports
WPM	Write accum to previous nybble of program memory
WR0	Write accumulator to previous RAM status char 0
WR1	Write accumulator to previous RAM status char 1
WR2	Write accumulator to previous RAM status char 2
WR3	Write accumulator to previous RAM status char 3
SBM	Subtract previous RAM char from accumulator
RDM	Read previous RAM char into accumulator
RDR	Read previous ROM input port into accumulator
ADM	Add previous RAM char to accumulator
RD0	Read previous RAM status char 0 into accumulator
RD1	Read previous RAM status char 1 into accumulator
RD2	Read previous RAM status char 2 into accumulator
RD3	Read previous RAM status char 3 into accumulator

MPR reconstructed this table by studying vintage 4004 documentation, including a data sheet and a user's manual provided by Intel. Most instructions are eight bits long, but a few (denoted by an asterisk) are 16 bits long. Many instructions will be familiar to present-day assembly language programmers, especially those who work with embedded processors.

instructions survive in its descendants. The joke answer was yes: NOP. Actually, several 4004 instructions are common in today's instruction sets, although certainly the opcode assignments have changed. Most of the arithmetic and logical instructions will be familiar to any assembly language programmer.

Three unusual instructions stand out: DAA (decimal-adjust accumulator), KBP (keyboard process), and DCL (designate command line). In the main article with this sidebar, codesigner Ted Hoff explains that DAA allows the 4004 to manipulate either binary or binary-coded decimal (BCD) numbers—a feature specifically included for the 4004's primary target application, the Busicom desktop calculator. Essentially, the DAA instruction converts a binary value in the accumulator into a BCD value.

Similarly, the KBP instruction converts the contents of the accumulator from a one-in-four value (0001, 0010, 0100, or 1000) to a corresponding binary value (0001, 0010, 0011, or 0010). Any other original value (e.g., more than one bit "on") becomes 1111, signaling an error. Apparently, this instruction allowed the 4004 to interpret input from the Busicom calculator's keypad.

The DCL instruction allows the 4004 to access multiple banks of data RAM. If a program accesses memory without using this instruction, the 4004 defaults to bank 0. Preceding a memory access with DCL allows the program to select one of three other banks. An optional external decoder allows the DCL instruction to select one of eight banks.

Although the 4004 has a few jump instructions for calling subroutines, it lacks an explicit return instruction. Instead, programmers used the BBL (branch back) instruction, which branches to the 12-bit memory address stored one entry below the stack pointer in the address stack. The stack has only three entries in addition to the program counter, so there's not much depth for nested subroutine calls. Also notable is the instruction set's lack of Boolean logic operations, which Intel corrected in later microprocessors.

Like almost all early processors, the 4004 has indirect memory addressing. Several instructions access memory via register pointers and transfer values among external memories, registers, and the accumulator. These operations are useful in a small microprocessor with limited memory capacity. Later, as memory latencies lagged behind processor speeds, CPU architects frowned on memory indirection, because each memory access stalled the processor. In addition, the increasing reliance on high-level compilers favors instruction sets that separate memory operations from other operations.

Overall, the Intel 4004 is a sophisticated design that holds up surprisingly well, even 35 years later, especially considering the circumstances: it originally targeted a single application; it was designed in about a year, without EDA tools; the design budget was only about 2,500 transistors; and it was the first microprocessor created by what was then a memory manufacturer. The 4004's only severe handicap is its lack of interrupts, which renders it obsolete for modern microcontroller applications.

Perhaps the most interesting lesson the 4004 offers is that the ridiculously strict design constraints—not the imaginations of the architects—were the major limitations on the project. Given even a little more design freedom (a larger transistor budget, more I/O pins, more time, better design tools), Intel could have created a significantly better microprocessor in 1971. This realization implies that chip-fabrication technology, design methodology, and time-to-market pressures are truly the driving forces in microprocessor design.

How Microprocessors Upset the Computer Industry

By Don Alpert, *MPR* Editorial Board

Intel's first advertisement for the 4004 claimed it was "Announcing a new era of integrated electronics." In 1971, that must have seemed like pure marketing hyperbole to almost everyone who read it. Few, if any, recognized that the new technology would be so disruptive to the computer industry.

At the time Intel introduced the 4004, the computer industry had come a long way since IBM chairman Thomas Watson Sr. remarked in 1943, "I think there is a world market for maybe five computers." By 1971, mainframes were in their heyday. IBM and other companies had their own semiconductor fabs to satisfy their needs for logic components, and semiconductor memory was starting to appear, although most memories were still made from ferrite magnetic cores. In 1968, Robert Lloyd of IBM's Advanced Computing Systems Division asked the question: "What the hell is it [a microprocessor] good for?"

Minicomputers from DEC and other companies had overcome initial skepticism about their ability to establish a market distinct from mainframes. The semiconductor industry supplied small- and medium-scale integrated circuits to build these minicomputers, as well as to build electronic

products in other industries. The minicomputer industry thrived during the 1970s with many new companies, architectures, and applications. These minicomputers used relatively simple, general-purpose integrated-logic circuits, like datapath bit-slices, combined with semiconductor ROMs for microcoded control and DRAM for main memory.

The first microprocessors had limited functionality, low performance, and poor reliability compared with existing computer technology, but they were inexpensive. Moreover, the microprocessor satisfied important needs for both semiconductor manufacturers and embedded-system developers. Semiconductor manufacturers had a new logic product that could take advantage of large-scale integration and could be sold in sufficient volume to overcome increasing development costs. Embedded-system developers could replace specialized logic with a common set of circuits customized for a variety of applications. The industry had created a virtuous circle separate from the computer industry. Investment in semiconductor memory technology (DRAM and ROM) enabled further microprocessor development and new applications, which in turn created more demand for memory.

In 1977, the first microprocessor-based personal computers that didn't require assembly began appearing, but DEC chairman Ken Olson declared, "There is no reason for any individual to have a computer in his home."

By 1980, semiconductor technology had progressed to enable the full functionality of a minicomputer—including 32-bit integer data and addressing, 64-bit floating-point data, and paged virtual memory-in two or three chips. New microprocessor architectures and vendors flourished. RISC technology spurred proliferation by simplifying microprocessor designs. Microprocessor technology enabled new types of computers, such as workstations and small servers, which challenged the old computer industry. By the end of the 1980s, single-chip microprocessors had eclipsed the performance of minicomputers. In this period, Intel sold over 10 million 386-series microprocessors.

In the 1990s, continuing advances in semiconductor technology enabled further improvements. By this time, nearly all computer systems—from small portables to enormous supercomputers—used microprocessors. But the cost of building a state-of-the-art fab had grown to $1 billion or more, and many players in both the semiconductor and computer industries had to fold.

DEC was the most visible victim. Its Alpha microprocessors won the technology race, but low volumes in DEC's fabs made the chips too costly. It has been reported that John Sculley, then chairman of Apple, approached Ken Olson about using Alpha processors in the Macintosh. Olson reportedly declined, missing an important opportunity to make the Alpha financially viable. Instead, Apple joined with Motorola and IBM to create PowerPC.

Hewlett-Packard considered the same challenges of fab costs and limited production of its PA-RISC microprocessors. HP approached Intel to jointly develop a new microprocessor architecture based on research by HP's labs, and the project that developed Itanium was launched. When the designers from both companies first met, it was clear that the economics of semiconductor production had created vastly different perspectives. For Intel's designers, fabs were the company's crown jewels, its most valuable resource and biggest competitive advantage. Intel designers viewed their responsibility as developing advanced microprocessors for volume production in synchrony with Intel's investment in new fabrication technology and manufacturing capacity. In contrast, HP's designers viewed their company's fabs as cost centers that made their technically excellent systems too expensive to be competitive.

Today, microprocessors have become ubiquitous and yet nearly invisible. Multicore processors are becoming commonplace. Multithreading and system-level virtualization are making virtual processors more highly valued than many physical processors. Embedded processors are disappearing in a synthesized sea of gates that forms a complete system on a single chip.

And so the computer industry has been turned on its head. IBM now supplies game-console manufacturers with some of the most advanced microprocessors in the world, just so it can afford to continue investing in the fabs that manufacture microprocessors for its own workstations, servers, and mainframes.