DESIGNING A MAINFRAME ON A CHIP:

INTERVIEWS WITH THE i486™ MICROPROCESSOR DESIGN TEAM

BY ALLAN CHEN

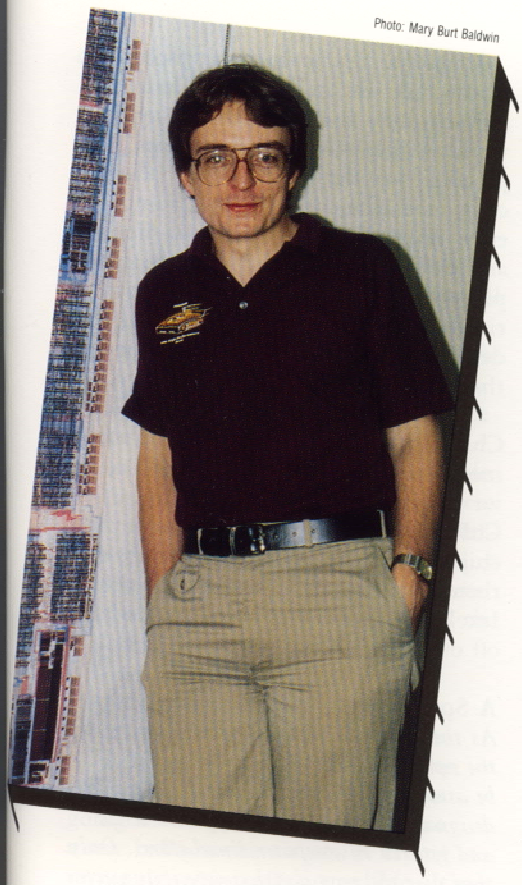
Photo: Ron Tussy

In the beginning, the i486™ microprocessor's design team had a single member. By the time it was over more than three years later, some four dozen engineers had labored around the clock to complete the chip. The end-product was a rectangle of silicon half as long as a paper clip, crammed with nearly 1.2 million transistors, and capable of delivering mainframe-level performance.

The road from idea to silicon required more than hard work. To build a device with a substantial performance improvement over the 386™ microprocessor, but which remained completely compatible with it, required that the team pioneer new techniques to raise processor speed. To increase the chip's integration, the designers brought such devices as cache memory and a math coprocessor on board. That led to a chip with more than four times as many transistors as the 386 CPU, a number so large that the project would never have been completed so soon without superior CAD (computer-aided design) tools. To finish the design on schedule also required that several phases of the design take place simultaneously. The man in charge of keeping all of this in motion was project manager Pat Gelsinger:

Certainly, the most challenging part of doing the i486 CPU was realizing a significant performance increase over the 386 processor. Even before talking to customers, you sat down at the whiteboard, trying to get some basic feeling for what you thought you could achieve. Then you went to customers and said, "This is what we think we can do, what do *you* think about it?" In 1985, shortly after we finished the 386 microprocessor, I started defining what the next chip would be like. I worked on it in isolation for a couple of months, and then put together a customer presentation. Then John Crawford [chief 386 CPU architect] and I started flying around to customers, presenting some of our basic ideas.

Photo: Mary Burt Baldwin



PAT GELSINGER

Q: Once you decided what was going into the chip, what happened next?

Gelsinger: We decomposed the chip into eight separate blocks: instruction decode, control, segmentation, paging, math coprocessor, integer unit, cache and bus controller. Then logic designers started trying to refine the individual block diagrams. They coded, designed and erased diagrams, iterating again and again until they were ready to integrate the different pieces of the logic model. Then, in early 1987, they started to run some real but basic programs against the chip in computer simulations. For the next year, it was a process of adding and refining functionality.

Meanwhile, we also had this major testing effort going on: we were running test programs through the i486 logic model for a whole year. We were working very intensely, running the 386 processor's test suite against it.

Another activity that was going on was the writing of the microcode, the instructions stored in the chip's read-only memory that execute certain instructions. The microcoders worked with logic designers to improve hardware and speed up certain operations.

When the first version of the logic model was ready, we started doing circuit design. The circuit designer designs every transistor of the chip in great and gory detail. The other important activity was to start doing the floor-planning of the chip—determining where the blocks go, and how they hook up.

All of these operations were going on in earnest throughout 1988: finalizing logic and microcode functions, and the huge effort of testing the chip. We pretty much completed the testing activity by the end of the year.

As mentioned, logic *designers were the first to work on the chip. They developed an abstract, computer model of the logic operations and functions the chip would perform, called an RTL model, for Register Transfer Level. This type of model is a computer language-based description, rather than a circuit schematic. With a first draft of the model ready,* circuit *designers began to translate it into circuits, arrays of transistors that perform the operations specified in the model. The i486 CPU's circuit design group was led by Ben Roberts.*

Q: How does a circuit designer translate RTL logic into a circuit?

Roberts: It's pretty much done by hand. They have to hand-size the transistors, integrate the logic and run the design several times through computer-based checking tools.

Q: In a sense, you're generalists. You have to be able to design circuits for any section of the chip?

Roberts: Yes. Other designers in the team handle the logic. We were the ones that did the electrical design of programmable logic arrays, arithmetic logic units, read-only memories, registers and other circuits. We also set up performance targets—how fast it would take signals traveling along different paths to go from one place to another on the chip.

Running in Parallel

While circuits were being designed, other groups were also busy: architectural validation, design automation and microcode. Deepak Verma, who managed the microcode group, explains its function:

Verma: Many instructions require several clocks [the basic chip-level unit of time] to execute. By implementing all multiple-clock sequences in the chip's microcode—in structured memory elements rather than as devices on the chip—we reduced the complexity and increased the flexibility of the chip's control unit. Microprograms, sequences of microcode instructions, have basic advantages during the design process. For example, it's easier to change a memory element's content than to redesign circuits.

Q: What kinds of improvements in speed are implemented on the chip?

Verma: The most important is that certain common instructions that took a minimum of two clocks to execute on the 386 processor, have been

BEN ROBERTS

Photo: Mary Burt Baldwin

RAJESH GUPTA, JIM BRAYTON, KEN SHOEMAKER

Photo: Katie Woodruff

reduced to one clock on the i486 processor. We did this by having hardware and microcode control the execution of these instructions. Another improvement was using cache memory to make memory access faster. A third step was to write shorter microcode sequences for a given function.

Another aspect of the design process is called performance verification. Circuit designer Rajesh Gupta coordinated this effort even as he was finishing the design of the chip's math coprocessor:
Gupta: We were trying to verify what we designed, to make sure there were no major flaws. We were pulling together a big chip, and so many different design styles were being thrown together—we wanted to make sure there were no incompatibilities or interface mismatches. I made a list of some 30 different configuration violations that I wanted to check out. I ran queries and did configuration checks myself and then each circuit designer went through those outputs individually. We found some discrepancies and fixed them. We were the first to do performance verification *during* the design process—it has only recently become recognized in the industry as an important part of chip design. In most cases, it was done after the chip was completed.

Design team members relied heavily on CAD tools, both new and established, to help them. Some of these tools were at the cutting edge of technology—so new that they were being perfected as design engineers used them. Jim Brayton, the youngest member of the design group at 24, was in the design automation group, where he worked on several of these tools:

I was one of the design engineers whose task was to define CAD tool methodology and to make the tools work on very advanced processors. One problem was the large amount of time consumed when it came to pulling everything together—chip planning, and the full-chip hookup and assembly process seemed to take up too much time. Standard layout tools weren't designed to provide all the facilities that were needed.

To address this, a tool called ChPPR, for chip planning and placement routing, was developed while we were designing the i486 CPU. ChPPR places and routes units of the chip and the interconnections between them automatically. In the end, the Intel tool shaved a couple of months off our schedule.

A Sprint to Tapeout

As the design process neared completion, the separate blocks of the chip needed to be assembled into one unit and the final design checked and rechecked, and tested and retested in computer simulations. Only then was the vast database specifying every transistor of the i486 processor design ready to tape out—i.e., be transmitted to Intel fabrication plants. Masks are created from this database, and the masks are used to manufacture chips. Avtar Saini, manager of the tapeout process, describes the final frenetic weeks:
Saini: We would work around 24 hours, go to sleep for five hours, and work again for 22 to 24 hours. In

AVTAR SAINI

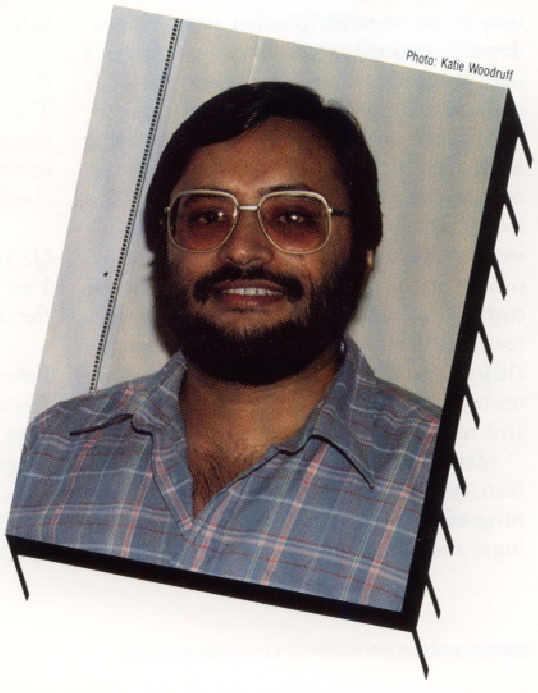


Photo: Katie Woodruff

about four days, somewhere along the line, you lost a day.

Q: Can you describe the tapeout process?

Saini: It's putting the chip together and creating a database to run full-chip verification [a series of tests]. All the unit designers had the responsibility of making sure their units were clean. To make the tapeout process easier, we divided the chip into three clusters, each consisting of two or three of the chip's eight units. People at the cluster level took units in their cluster and made sure they were interfacing correctly. Then we put the clusters together—making sure interfaces between clusters were clean—until the full chip was assembled.

Project manager Gelsinger found that things were speeding up at that point:

When you get to the end of the project, there are so many things going on in parallel. You have so many balls in the air you're trying to juggle—it's really incredible. It's like a great symphony building to a huge crescendo at the end.

Q: All told, how fast did you tape out?

Gelsinger: We finalized the database in February, and then we started doing the final assembly of the chip. In about four weeks, we went from this disassembled database to tapeout. We taped out March 1st at 3:42 p.m. On March 20 at 3:45 p.m., we received the first silicon from fab.

Saini: In less than an hour, we had the first three tests going, so we knew the chip was alive. In less than five hours, we were running more than a million test vectors. One part of our test suite was all done. Well before midnight, we were up to two million test vectors. It was like watching election results. A few days after getting silicon, we were running DOS, Space Invaders, Flight Simulator, all sorts of games. The chip was amazingly clean.

Ken Shoemaker, a logic designer and member of the debug task force, explains a longstanding tradition of the test phase:

"...In about 10 weeks, we went from disassembled chip to shipping multiple samples and running large quantities of software. I'm amazed..."

While I was at Purdue University, we had an 8085 C compiler and a program based on the Space Invaders video game which we used to bring up hardware. I worked on the 80186 after coming to Intel. I put together a system to test the first silicon and got Space Invaders working on it. When the 386 processor came out, we put together a card so that we could plug it into an 80286 system. Space Invaders became the first program to run on the 386 as well. Of course, precedent had to continue, so it came up on the i486 CPU, too. Worked like a champ.

The project manager's associates had a new name for what happened in the days following the receipt of silicon—Gelsinger warp speed:

These things happened so quickly on the back end. We shipped out our first sample to customers four days after receiving the first packaged units. The silicon was extremely clean, so we could sample on the very first thing that came out of fab. In summary, in about 10 weeks, we went from disassembled chip to shipping multiple samples and running large quantities of software. I'm amazed.

Q: Now that the i486 microprocessor is out, what comes next?

Gelsinger: I think that one of the hardest things was making the i486 CPU significantly better than the 386 CPU. Well, now we do it again. The i486 processor uses about 1.8 clocks per instruction, compared to 4.4 clocks per instruction on the 386 processor. In the next generation, we'll have to figure out how to halve that yet again.

This may sound really crazy, but the day after we got the first full-chip plots, I was looking at them on the light table, thinking, "We're going to be able to do so much better next time." Those were my first impressions. My wife thinks I'm crazy, but I guess it's these kinds of demented people that keep this business going.

DEEPAK VERMA

Photo: Mary Burt Baldwin