Background of Centaur Technology Part 1: 1979–1993

This chapter covers the evolution of my idea of developing a *processor* that could execute both the x86 ISA and another ISA, which was one of the founding concepts of Centaur Technology. It overlaps several of my IBM periods long before Centaur started, and I include random descriptions of related things happening during those periods.

This chapter covers the origin of the founding idea of Centaur Technology: that one could develop a processor that would execute x86 instructions on top of a different instruction set. My personal interest and motivation for a "multiple ISA" processor had started in 1980 when I was at IBM, 15 years before Centaur started. It became more real in 1982–1985 when I managed the development of the first shipping IBM RISC processor, effectively "competing" with the IBM PC developments. It became even more real in 1985–1988 when I was an IBM Fellow and had people working on such a dual-ISA project (internally called the "L86"). Of course, the idea is much older than me: a large part of the success of the IBM System/360 family (announced 1964) was due to its emulation of older IBM processors as well as introducing the brand-new S/360 ISA. Other IBM products (notably the 5100) used emulation of "another" ISA as well. Some mainframe competitors also emulated the System/360 ISA.

Note that there are two general approaches for emulation: (1) an add-in device (e.g., card) that actually contains the "alien" processor, and (2) internal extensions *within the main processor* that emulate or assist emulation of the alien ISA. My focus has always been on #2 in order to reduce the large cost increment of #1. More on this when we get to the "IBM RT Personal Computer Workstation" (RT-PC).

My Apple II & IBM PC

In December of 1979, I was the manager of about 600 people doing the system architecture, microcode development, and all system software development for the new IBM System/38. I was doing the hardest work I have ever done trying to get the IBM System/38 to ship. I had been working 6 days a week for months, and things were starting to look good. But, given that we had missed our publicly announced ship date—a *major* sin in IBM—I was surprised when the Lab Manager, Hal Martin, called me in and gave me another $5000 outstanding innovation award.[1]


IBM presents to
G. Glenn Henry
an Outstanding Innovation
Award
in appreciation for:
System/38 Machine Interface

[1] Note that the award is for the most controversial feature (*during development*) of the System/38 (a "high-level machine interface"). During development, many fights occurred over this "top-priority" feature but I was the main inventor and champion of a high-level machine interface, and since I was the manager of the architecture and all software, the high-level interface remained. Later in 1983 I also got a large "Corporate Award" for this same feature. This concept and the unique S/38 architecture are covered more in my full memoir, as well, of course, in IBM publications such as "IBM System/38 Technical Developments" ISBN 0-933186-00-2.

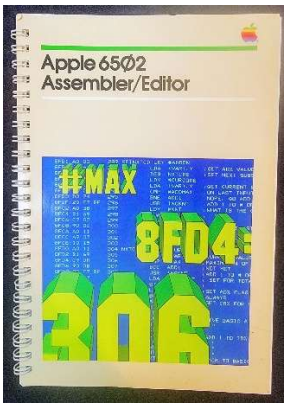
Copyright © 2026 G. Glenn Henry

I already knew exactly how I wanted to spend the unanticipated arrival of money: on a personal computer. I had been eagerly following the emerging PC technology since I first read about the Altair kit in 1975. IBM had also introduced a desktop computer—the 5100—in 1975, but it was targeted toward engineering professionals and cost over $10,000. Interestingly, the 5100 emulated the IBM System/3 ISA so it could run the BASIC code developed for the System/3 Mod 6. Unlike their pervasiveness today, personal computers were quite rare at that time and I knew of no one who had one. I did a lot of reading of computer magazines and then drove to Minneapolis to spend hours visiting computer stores to see what they had (PCs were so rare that I couldn't find a vendor in Rochester).

There were many contenders available that year: Radio Shack's TRS-80 Model I, the Atari 400/800, the Commodore PET, the Apple II, a new NEC PC, a new TI PC, and others I can't remember. I choose the Apple II+ (the "+" was new version shipped earlier that year) with the Apple diskette drive and monitor. My choice of the Apple was based on its "looks" and the number and type of applications that were already available for it. By the start of 1980, the Apple was installed in our spare bedroom.

Initially, I just wrote simple Basic programs on it but also started to collect interesting commercial games and applications. The "killer" app those days was the newly released Visicalc, generally considered the first spreadsheet app. Apple had its own word processor and, of course, there were games! I don't remember now exactly what applications I had in those early days but there were definitely "shoot-'em-up" type games. Kevin, now almost seven, was very interested.

But, being a low-level computer designer and programmer, I soon became interested in the processor and writing assembly code for it. The Apple manuals were excellent, having a description of the hardware design and the ROM code. There was an assembler available and commands to run the assembled program. I was particularly interested in graphic algorithms and code for the Apple. I had dreams of writing my own graphics-oriented game (tentatively called "SpaceBall"), but that really never happened. But I did write a fair amount of sample assembler code (mainly graphics routines) for its processor—the MOS Technology 6502.



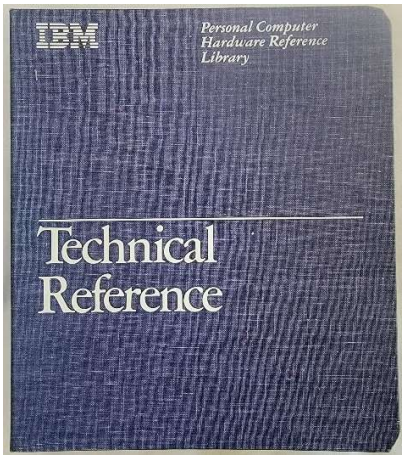
Six months after I got the Apple, the System/38 finally really shipped and Peggy, Kevin, and I escaped Rochester. We moved to Austin in mid-1980 and the IBM PC was still not available, even within IBM. My Apple II+ continued to be used. One interesting story was that at that time Lee Reiswig worked for me, and I remember him visiting and playing games on my Apple. Lee, of course, later became the IBM General Manager responsible for OS/2 and associated software.

Finally, the IBM PC arrived in the Fall of 1981 and because of my position I could get one almost on the announcement date (no price discount however). I distinctly remember picking it up on a Saturday morning while I was taking Kevin to a basketball practice. I remember sneaking out of the gym and sitting on the steps reading the Technical Reference manual instead of watching the practice.

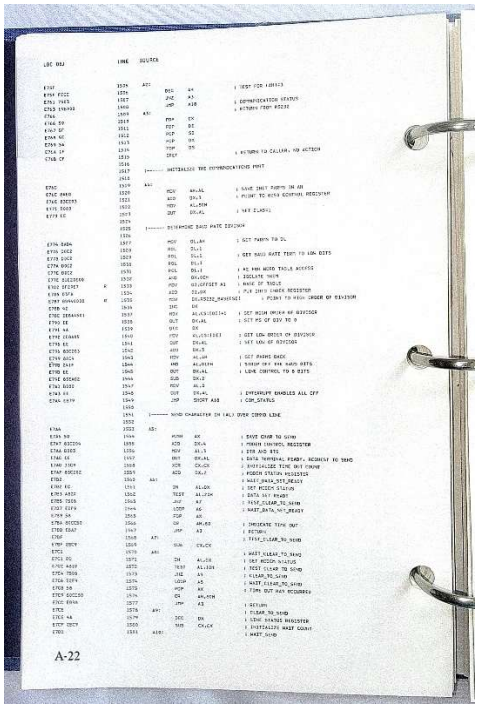
My excuse for this questionable parenting is that the IBM PC Technical Reference manual was one of the most important books in the history of computing. The Technical Reference manual described in detail almost everything one needed to know in order to make a personal computer that could execute software written for the IBM PC! Including, in the first release, a listing of the entire IBM BIOS!

Copyright © 2026 G. Glenn Henry

Below is a random page from Technical Reference manual showing part of the listing of the complete BIOS (showing part of the communication-port code). This was a gold mine of information for a potential clone maker and also for a low-level programmer such as myself. I spent many hours devouring the BIOS listing.



I immediately started to write sample 8088 assembler code. I noticed that the Apple's "8-bit" 6502 ISA was almost a perfect *functional subset* of the "16-bit" 8088. The 8088 had more registers, both had similar instruction functions and same memory address size, and so forth. So, could I somehow reuse my 6502 code on the 8088?



The obvious approaches were (1) to write a static "translator" ("compiler") of my 6502 assembly code into 8088 assembler code, (2) write an execution-time emulator of the 6502 running on the 8088, or (3) manually rewrite my code in 8088 instructions.

I went down the static translator and dynamic emulator holes for a while but ran into problems whose solutions would slow execution a lot. The 6502 "problems" I remember are handling setting the status register after loads, the "zero page", and the fixed address stack ("page 1"). The address cases deal with the fact that the 6502 has special addressing modes to reduce instruction size.

Since normal 8088 programs (esp. BIOS) use the real page 0 and 1, the 6502 memory address map can't be mapped directly on top of the 8088 memory map. Of course, I could rewrite my 6502 code to avoid using page 0 and 1, but the point was not to write any more 6502 code. The fact that 6502 loads set condition codes is troubling from a performance aspect: all translated 6502 loads must be followed by a test instruction on the loaded data. And, there likely are other complications now forgotten.

So, I gave up on directly reusing my actual 6502 graphics code and just rewrote it in 8088 assembler.

But… the idea stuck in my head that if one were designing a new hardware version of the 8088 it wouldn't take much extra hardware to be able to have it boot and execute in 6502 mode. The complications that I mention above won't be a problem if the processor always executed in one mode or the other with a memory-address mapping, and a little extra hardware could be added for fetching/decoding the 6502 forms of instructions. In addition, a small amount of native code could handle trapped little-used protect-mode curiosities. Today we know that the 8088 used about 29,000 transistors and the 6502 used about 3,500, many of which were ALU functions that are almost exactly the

Copyright © 2026 G. Glenn Henry

same as contained within the 8088! So, the hardware cost of dual-ISA execution would seem to be trivial.

But why would someone want such a dual mode processor? The answer, of course, is to *reuse software!* At the time of the IBM PC announcement, the Apple OS and especially utilities and applications were much better than those on the PC. So, a dual mode processor could be booted in Apple mode to play our favorite game, and then rebooted in IBM mode to run PC applications. And so forth. Plus, such a processor would likely be a lot faster in 6502 mode than the 6502 chip itself was. Even at that time it was obvious that the cost to develop or migrate software from one processor ISA to a different type would be huge. Today, the cost for software to switch ISAs is prohibitive.

And, of course, clever design would remove the need to reboot and just apply "Apple mode" to a particular task: that is, an integrated emulator.

Such dual-ISA processors were not new to IBM. Much of the early success of the S/360 family was attributed to the fact that they all did hardware emulation of popular previous generation IBM processors ISAs. For example, the Model 50 could emulate the 1401, 1410, 1440, 1460, 7070, 7072, and 7074 processors. Initially, the emulation was standalone but later emulation could be controlled by the OS and individual threads used emulation or not.

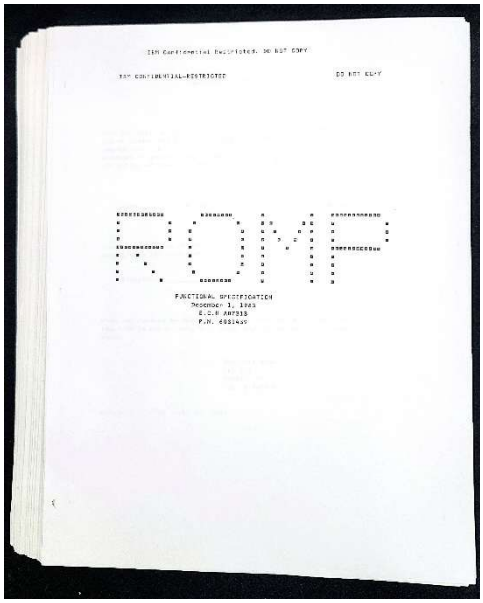
Of course, emulating one's previous processors in a new processor was sort of an obvious business approach. But, to me, even more compelling is emulating *other* manufacturers in order to intercept the business that is driven by their software. As an example, several other computer manufacturers emulated the IBM S/360 ISA. [2]

But, I wasn't in the business of designing processors, so I moved on at that time.

[2] Occasionally when I bring this topic up someone asks something like "can't instructions be copyrighted or patented or something to protect them from other people using them?" Having dealt with this topic a lot in my time at Centaur, the short and simple answer is that it is possible to protect *patentable* features of how an instruction is implemented, but it's hard and costly, and the ISA format and function of an ISA instruction, per se, can't be (easily) protected.

Copyright © 2026 G. Glenn Henry

IBM ROMP vs. Intel 286



When I arrived in IBM Austin in 1980, the organization and projects were somewhat in disarray. I actually had no meaningful job for about two years until a reorganization occurred. But sometime about the start of 1982 I became the Lab's "Advanced Technology" manager and inherited a small group (maybe 10) that was developing a new RISC processor. It was called "ROMP" which stood for "Research Opd MicroProcessor" (OPD was the name of the Austin division within IBM at the time the project had started). This was a follow-on to the original 801 RISC processor project from IBM Research and it was tightly connected with the Research division.

But, I had no interest in making a processor just for the Research division's usage, so I proposed the obvious: we should make a new IBM *RISC-based UNIX-based engineering and scientific workstation*.

The problem was that mainstream IBM didn't know what UNIX was, or what an engineering and scientific workstation was. UNIX had been around for just a few years, but mainstream IBM ignored it. Workstation products were just then appearing in the marketplace: the first workstation product from Sun (the SUN-1) shipped in early 1982. So, I had funding for a RISC *processor* development but nothing that could support my idea of a UNIX-based workstation.

But, I developed a plan, obtained funding by escalating all the way up to the #2 IBM executive, hired UNIX talent, developed a relationship with a UNIX supplier[3], and we started design on the workstation itself and its supporting software, and modifying the purchased UNIX to become IBM's AIX OS. Obviously, a lot of work, but I'll skip the details here.

The evolution and development of the IBM RT PC and the new IBM AIX OS was long and very frustrating to me. All of the flaws of IBM were displayed. I'll skip that horror story for now to focus on instruction emulation thoughts, which returned consistently to me during the development. During development, there would be upper management reviews of major projects every month or so. Austin and Boca Raton were now in the same division in IBM so these reviews always addressed both the RT PC project and the various new PC projects like the PC AT. A review usually went something like this: The PC representative would present charts saying how fast the new PC was, what its development cost was, what its product cost was, and how many applications were available for it. Then I would have to get up and explain that our CPU performance (using questionable benchmarks like "MIPS") was maybe 100% faster, our development costs were 300% larger and our product costs were 100% more than those of the new PC, and we had *no* applications (other than built-in UNIX functions)![4] To the unpleasant comments and questions that greeted this, I tried to explain that our target workstation users didn't use off-the-shelf applications like in the PC world, and our high cost was because we were using the conventional IBM process and product costs vs. the PC using its "special" process

[3] "Interactive System Corporation" they offered UNIX versions based on AT&T's System V.

[4] I obviously don't remember the actual data, but these numbers are pretty good approximations of the facts as I remember them.

Copyright © 2026 G. Glenn Henry

and vendor costs, and mumbled about bad performance benchmarks and how good a RISC ISA was for extendibility, IBM leadership, etc.

When you combine these comparisons with the fact that IBM constantly tried to turn the RT PC into a "business" system (ultimately costing us about a year in schedule [5]), it's a wonder that we ever shipped. A lot of the credit has to go to my new boss—Frank King [6]—and my great main lieutenants: Charlie Sauer, Clay Cipione, and Larry Loucks.[7] The corporate influence of IBM Research, who was pushing the new RISC architecture, probably also helped, but they also delayed us in some areas.

In the meantime, the IBM PC sales were growing rapidly, and most importantly, the software available was increasing rapidly. My family followed the PC trend, getting a PC XT (hard disk) when it came out in 1983. We even got a PCjr later that year for Kevin to use, and most importantly a PC AT with the Intel 286 when it was available in mid-1984.

At work, I knew the AT was coming with its new 286 processor, followed by the 32-bit i386 with virtual memory. It was obvious that the increased performance, increased memory size, and protect-mode features of these new processors would cause an explosion of applications and the usefulness of the PC. So, I was constantly thinking about modifying the ROMP processor to be able to emulate the Intel 286. I figured it wouldn't be hard, and wouldn't add a lot of cost, and wouldn't hurt RISC mode performance. But, it would give us instant access to mass quantity of PC applications, albeit running in PC mode. It seemed to me to be an obvious "intercept" strategy that would buy time until the value of a RISC/UNIX system became more obvious. But, given that RISC was the pet project of Research, and that in general they opposed the PC, and most definitely hated the Intel ISA, they wouldn't support this idea of an integrated emulator and it was dismissed out of hand.



But… the rapid growth of the PC market finally forced consideration of PC emulation into the RT PC; but it was done via this very costly and very complicated add-in card containing a i286 chip (lower left on the card) and a lot of other circuitry to deal with providing bus, interrupt, and video trapping and relocation. This is not my idea of how to do emulation! Most of this logic could be done within the main processor with far less product cost and much greater AT-mode performance.

[5] A horror story that I'll cover in the main memoir.

[6] W. Frank King III was a PhD from Research division who had made major contributions to IBM "System R" project which had pioneered the relational database concepts. He was good technically as well as being a polished manager with good relationships with Research.

[7] After the long-term value of AIX to IBM was seen, in June 1988, Charlie, Larry, and I got a large Corporate Award for AIX.

Copyright © 2026 G. Glenn Henry

Another area of disagreement between Research and myself was on structuring the software immediately above the processor ISA. They were interested in seeing their good "invention" (RISC processor) become "real" and get shipped with best possible performance. I also wanted that, but I also wanted an "architected high-level machine interface" between the processor and the operating system (shades of System/38). And, as the manager, since I had to deal with the budgets, people management issues, IBM politics, relationship to other projects, and other distractions, when there were differences of opinion or priorities, we did it my way.

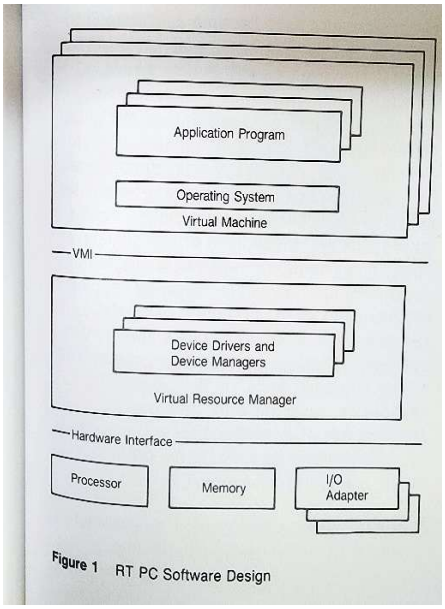


Figure 1 RT PC Software Design

So, there was a "high-level machine interface" between the processor and the operating system called the VMI (Virtual Machine Interface) and the functions below this interface were the VRM (Virtual Resource Manager). This structure is conceptually similar to the high-level machine concept I mentioned on the System/38.

Research people complained that the VMI slowed down the system performance a few percent, but this interface was very important for compatibility, customization, and simplification of software development. This is just one example of the differences between our product-oriented group and the Research division.[8]

Another issue with Research was how great the "RISC" concept really was (or is, today). While the ROMP ISA was infinitely cleaner and more efficient than the i286 or i386 ISAs, I always questioned how important this would be in the future when transistors became "free" and the ISA per se wasn't the bottleneck on performance (as opposed to memory access, for example). It is interesting to note that the two dominant general-purpose ISAs today on which all business runs (x86 and ARM) are both *very* complicated.

Finally, I should mention someone who worked for me on the RT PC project: Tom G Whiteside, who was an engineer working on the ROMP subsystem. He went on to greater things, and in early 1994 when I was looking for a home for my "Centaur" idea he was the president of MIPS. He hired me and the three other Centaur founders (Terry Parks, Darius Gaskins, Al Sato) as consultants while we developed our ideas about a MIPS processor that emulated an x86 and looked for someone to fund us support. Thanks a lot, Tom!

My "L86" Fellow Project

Finishing the development of the RT PC and AIX was complicated, long and very difficult!. By the Spring of 1985, I was managing both the software and hardware development of the RT/PC and AIX products, but we were almost done with the first release. One day, I was in a meeting when I was called out to take a call from Don Estridge, who was our division President at that time. He said that I had been elected as an IBM Fellow, one of four selected that year!

[8] This picture is from a great IBM publication book if you want to know more about the RT PC: "IBM RT Personal Computer Technology", SA23-1057.

Copyright © 2026 G. Glenn Henry

This was a total surprise to me. In those days there were only about four or five Fellows per year selected and previous IBM Fellows included some major technical people such as John Cocke, Harlan Mills, Ken Iverson, Frances Allen, Benoit Mandelbrot, John Backus, Edgar Codd, George Radin, Gene Amdahl, and so forth. There were Nobel prize winners and Turing award winners among them. The three elected that year were Hans Pfeiffer, Grant Wilson, and Jerry Woodall, all PhDs with major accomplishments. The following year two of the new Fellows had received the Nobel prize that year. Also, almost all Fellows came from the Research division, and I had very poor relationships with Research, so I had no expectations, and I had never even indicated that I wanted to be a Fellow.

In many ways an IBM Fellow was the perfect job. A Fellow got to work on whatever he/she wanted with some TBD funding for the projects. There were few budget constraints and almost no management interactions. Another perk was yearly attendance—with spouse—to the corporate recognition event. Peggy loved going to that event.

I had originally assumed that I would continue managing my group to do the follow-on to the RT/PC, but now Andy Heller, an earlier IBM Fellow, took that position. That group went on to evolve the ROMP processor into the IBM Power processor and the RS/6000 family of systems using new versions of AIX.

In the meantime, I wanted to focus my energies on the PC world since I saw that was where computer processing was headed. I remember going to see a high-level executive in New York and explaining how the natural evolution of PCs and technology would significantly change the computer business. He laughed at the thought and said that maybe the PC could be used as a terminal to a mainframe, but otherwise it wasn't important. This was the common attitude of the IBM executives. But, as a Fellow, I could ignore such short-sightedness… so I started working on developing a "Audio-Visual PC". This was a PC that could record and play back sound and video. That seems obvious today, but the PC of 1995 couldn't do either with commonly available or reasonably priced add-in devices. Our thesis was that such audio-visual I/O was fundamental to a PC and should be easily available. I was able to find and recruit a few very talented IBMers, and we started research and actual design of some components.

It turns out that my old friend Brad Beitel had gone back to California from Boca and was a manager of an AV project at IBM's Menlo Park scientific center. In particular, his group had just produced a great DOS app for the PC called IBM Storyboard.[9] This was a PowerPoint-type application available several years before Microsoft's PowerPoint! Brad's group was short of funding, so I told him to come work for me. Nobody every raised funding questions. So, Brad and his team worked for me for a little over one year as we developed the Audio-Visual PC proposal. They focused on software support while the Austin team focused on hardware.

We assembled a prototype built out of several discrete components supported by some new Storyboard features and took it to Armonk and showed it to executives, including the new CEO, John Akers. We got no interest from them, and none from the main PC development group in Boca. At this point the PC team were busy defining and starting their attempt to wrest control of the PC market from others by introducing proprietary IBM technology into the new PS/2 product line (like the new "micro channel" bus and the OS/2 software).

[9] https://en.wikipedia.org/wiki/IBM_Storyboard_Plus. I'll give a rare pointer here because this is such a good product for 1985. It (and Brad Beitel) deserves to be remembered as pioneers of the PC revolution.

Copyright © 2026 G. Glenn Henry

I was an out-spoken opponent of both of these attempts to capture market share by use of proprietary-only technologies as opposed to beating the clones directly with better products. In particular, I'll mention that one time I was asked to give a talk in Boca on the future of processor technology. I had an audience of hundreds and at some point, I said something like "anyone who builds a new OS today without supporting virtual memory is crazy and doomed to fail in the marketplace." Apparently, this incensed the OS/2 managers since the original OS/2 did not support VM. I actually got a chastising mail from some Boca executive. Years later at Dell I gave talks at technical conferences showing that the new "IBM Micro Channel" was not really better than the EISA PC bus.

I was also asked to be part of a corporate-level task force that was investigating why the IBM PC group "missed" the first introduction of a PC that used Intel's new 386 processor, which, of course, implemented a super important feature: virtual memory. Compaq had introduced a PC using the 386 before IBM and got significant business and credibility with it. The answer, of course, was that the responsible executives didn't understand that IBM no longer controlled the PC marketplace and couldn't dictate or control the use of new hardware and software.

Anyway, following the lack of interest in the audio-video PC project, I decided that we should apply the "processor intercept" strategy I had developed over the years: start with a product that provided unique IBM benefits (like the new Power processors and AIX), but add a *low-cost* capability to *also* run a standard PC OS and PC software as a "task".

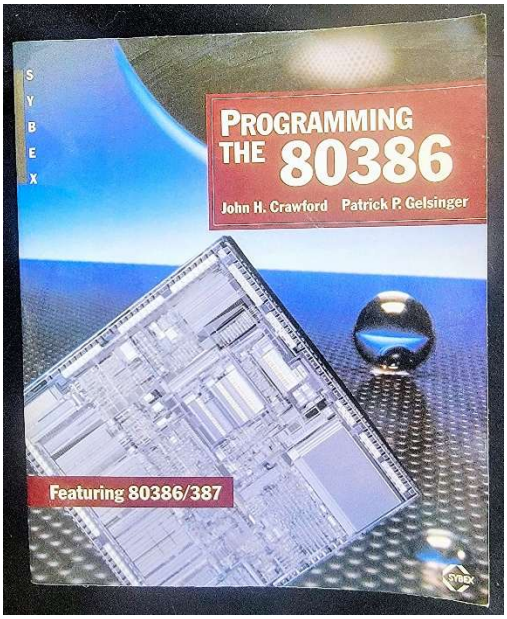
This was my "L86" Fellows project. The "L" stood for "last". The idea was to "intercept" the new x86-based software that was coming out for the newly shipped 386 architecture by being able to directly run that 386 software under control of an IBM OS that also ran "native" applications that took advantage of a level of performance and function that the Intel base couldn't match. That is, an integrated task-level 386 hardware emulator capability.

And, I knew where to find the base processor and software… right across the hallway where the new IBM Power processor and next AIX were being developed by people who used to work for me. I and my team would design the x86 emulation features. Note that we didn't plan to emulate the complete 386; we would trap to native-mode for some rare x86 features that were not used or not critical to performance; the 386's tasking instructions being a good example.

But first, we needed to become more expert on the details and implementation of the 386 than just reading manuals. Two major things really helped here. First, I found that IBM Burlington fab had made a CMOS clone of Intel's 386SX under license from Intel. It was called the 386SLC or "Blue Lightening." Second, I was appointed to be the technical representative of IBM to Intel on x86 architecture topics.

I somehow contacted the right group in Burlington to see if I could find a x86 processor designer. I finally talked to Dan Millett, a manager who said he had a great processor designer: Terry Parks. I got Terry to come to Austin for an interview and immediately hired him. That turned out to be the best hire I ever made, as I can confidently say—having worked closely with him for the next 35 years—that he's the greatest processor engineer I've ever met. Terry followed me to Dell, he was one of the four founders of Centaur, he was the chief architect of Centaur, later an Intel Fellow, now an AMD Fellow, and so forth. Later, when Centaur started, Dan Millett was also one of our first-year hires.

Copyright © 2026 G. Glenn Henry



Sometime around this time (1987), one of the great books of processor architecture was published, shown here. Since I was the IBM technical representative to Intel on architecture, Terry and I had many calls with the authors. A typical call would be like this: we would say "the books says X but we don't believe it", and they would answer "yes, the truth is really Y", or whatever the facts were.[10]

So, my team continued to grow as many people were fans of the PC architecture and questioned IBM's uniqueness strategy. We worked on proposing changes to the new Power ISA architecture. We had lots of interactions with key Power designers where they would come and we would propose a change or addition to their component, and we would work out the technology together. The only area where I remember things getting complicated was floating point (Intel's ugly x87 floating point had a lot of detailed changes from the pure Power FP).

It immediately became very clear that many IBM executives were not thrilled with the L86 "intercept" concept. In the early 1980's, IBM had actually owned 20% of Intel stock. Furthermore, the top technical executive, Jack Kuehler (Vice Chairman) had gone to school at Santa Clara Univ. and knew the Intel executives personally. So, even though I was an IBM Fellow and supposedly immune from management interference, we started to get push-back from executives and technical audits. An audit was a standard IBM process of sending top technical experts to review a project and make recommendations. I had participated as an auditor, and my projects (especially the S/38) had been subjected to many. Thus, I was pretty good at being able to deflect/stall/invalidate audits. The problem here was that the issue wasn't technical, it was strategy re the IBM approach to the PC market.

There were many arguments and issues re strategic issues, but our technical work to define such an integrated 386 emulation capability was going well. I had maybe 30 people, mostly hardware types working on it and I was constantly collecting more, including new hires[11]. But, I needed more PC software knowledge and capability. Amazingly, IBM offered to let me hire a few people in Boca that were closely associated with the PC software. I found about five there to join our project. The key Boca person was Dave Bradley, who was the main person writing the original IBM BIOS!

Anyway, L86 progress was happening but I was spending most of my time on selling the concept and "management" stuff instead of what I wanted to do (technical).

And… And…

So, I quit and went to work for Michael Dell! But, that's another story.

[10] Many many years later when I went to work for Intel I talked with Pat Gelsinger (then the CEO) and we reminisced about those discussions.

[11] The last new college grad that I hired, Tim Elliott, later became one of the first-year hires at Centaur.

Copyright © 2026 G. Glenn Henry