Oral History of William Michael “Mike” Johnson

Interviewed by:
Kevin Krewell

Recorded: May 9, 2014
Mountain View, California

CHM Reference number: X7168.2014

Kevin Krewell: Hi, I'd like to welcome Mike Johnson to the Computer History Museum. My name's Kevin Krewell. I'll be doing an interview discussing Mike's life history and how he got involved in IBM and AMD's processor business over his career. Mike, could you start by introducing yourself? Your name, title, and what are you doing these days?

Mike Johnson: OK, I'm Mike Johnson. I'm actually CEO of a small company called Mireplica Technology. And I can't say much about what we're doing, but I think it's kind of the computer architecture equivalent of a mind-blowing substance. So we'll see.

Krewell: Well, we could take that in a lot of different directions. But obviously that's--

Johnson: Yeah, take it in good directions.

Krewell: Good directions, oh yeah. That's good. So can you give us a little background on where you're from and what it was like growing up?

Johnson: OK, so I'm originally from Charleston, West Virginia. And it's not really known as a technology center, as I said before. But my father was in the Navy, so I was always kind of around technology and pilots. He was a flight surgeon. And we moved up and down the East Coast pretty much and ended up in Arlington, Virginia, when I was going into high school.

And we were there for three or four years before I went to college. And he was at Bethesda Hospital, so we were there. We got established a little bit. And the Arlington schools had some very nice infrastructure for getting hands-on experience with computers, using the old Teletype 33 machines, I think is what they were called with punch tape. And you could program in BASIC and use time-share machines. So I became fascinated with computers and electronics in general as part of that.

Krewell: So high school was when you first started to become interested in computer programming and developing, working with computers?

Johnson: Yeah, I had always been interested in building models and things like that. But when I discovered electronics and computers, it just fascinated me.

Krewell: So when you went to college, did you decide computers were your thing, or did that just come later on? How did that-- when you were in college?

Johnson: Well, it was really electrical engineering in general or electronics in general. In fact, the way I picked my major is I got the course catalog and flipped through to the first thing that said electrical and signed up for that major. And I could have been an electronic technician except it came after electrical engineering alphabetically. I'm serious. That's how I picked to become an electrical engineer.

Krewell: And which school was that?

Johnson: Arizona State.

Krewell: OK. Oh, that was a bit of a jump from DC to Arizona.

Johnson: Yeah. I wanted to get out and see the world. And I was on an Air Force ROTC scholarship. So I had a bit of flexibility picking where I went. And they had a lot of Air Force infrastructure in the Phoenix area at the time. I think they still do have one Air Force base. So I was interested in flying too because of the background that I had with the Navy.

So I had kind of a foot in two worlds. And the original plan was I was a pilot. I was pilot qualified. So I was supposed to spend four years getting my bachelor's, then a year in pilot training, and then pay back a year for every year that I had spent in training. So that would have been one year in pilot training and another five years being an Air Force pilot.

But I graduated in 1975, and they had filled the pipeline with pilots for the Vietnam War. And so in '75, it was pretty much completely over. And I probably have one of the few commissions signed by Gerald Ford. So they told me, basically, I can go into the Air Force at some point in time, but they were cutting back.

And I was fairly high on the qualification list, so they weren't going to just cut me loose, so they made me stick around. And I couldn't get a real job because I had the Air Force commitment, but the Air Force wouldn't let me in. So what I effectively did or essentially did was go to graduate school and get a master's. And while I was in graduate school, the Air Force sent me a letter effectively resigning themselves to the fact that they didn't have enough people who were voluntarily being let go.

So they gave me what's called a Palace Option, which the idea is you go to active duty for three months, and then you spend eight years on inactive reserve. And so I went to graduate school for a year, was stationed in San Antonio for three months during the summer. It was kind of like a summer vacation. I got paid pretty well.

Krewell: Oh, paid vacation.

Johnson: And I went back to Arizona State in the fall and graduated in December.

Krewell: OK, with a master's in electrical engineering.

Johnson: With a master's. And the master's research was in hybrid control systems and flight controls and that sort of thing.

Krewell: So mixing the loves of flying and electronics together.

Johnson: Yes.

Krewell: So what happened after that?

Johnson: Well, when I was in San Antonio I had originally had this I guess stereotypical view of Texas as flat – tumbleweeds – and nothing really interesting. It turns out that the area around San Antonio and Austin is in some ways reminds me a lot of where I grew up in West Virginia and Virginia. The climate's a little different, but the terrain is similar. There's lots of water, springs in that area.

So when I went back to graduate school, I mean to finish graduate school, I started looking at-- my first preference would have been to go to Beaverton, Oregon, and work for Tektronix. But I noticed some recruiting flyers in the recruiting office at Arizona State. And they had Texas Instruments and IBM, and they were in Austin.

And I said, well, I kind of like Austin. So I signed up to interview with TI and IBM. And I never heard from Tektronix. Or actually I heard after I had already accepted and moved, so they were a little bit slow on the uptake.

I interviewed a few other companies, but, well, the major one was Sperry Flight Systems in Phoenix, and by that time I was kind of burned out on the hot weather and just the climate. So I got to go up in some of their corporate jets, so they were really trying to-- I was under basically a fellowship from Sperry's Flight System. So they tried their best, but I just wasn't interested in the area anymore.

Krewell: Yeah. So you took a job in Austin, Texas.

Johnson: Right.

Krewell: With IBM or--

Johnson: I took it with IBM. I actually liked the job description of what Texas Instruments offered me because it was working on something that-- I think they called it the TI 9900. It's been so long. And I was really impressed because they had multiple registers, stored in memory, by the way, which is not where you want registers.

Krewell: Right.

Johnson: But I thought that was really a cool idea.

Krewell: Fast task switching was possible because you could just use a pointer.

Johnson: In theory, but the registers were in memory.

Krewell: So it's slow.

Johnson: So it's slow. And my interview at IBM had to do with power transistors, and how you activate solenoids, and what happens. Basically when you release a solenoid, you get the fly back voltage, and how do you handle that? Because at the time they did Mag Card typewriters and Selectrics [Selectric typewriters]. It was an Office Products Division.

But they offered me a lot more money than Texas Instruments. And at the time, I didn't have a-- it was go with the flow. So I joined IBM. And when I moved, I had this old Nissan truck. At the time it was Datsun.

And it had this problem with the oil filter. Basically the gasket would blow. And you got to the point where you could hear the sound, and you knew it was happening. So you turned off the motor and reset the oil filter.

Krewell: It would just loosen up?

Johnson: The gasket would actually blow out, and it would start squirting oil. So there's a reason I told you that, because in San Antonio, I stayed with some friends. And I went up to Austin with another friend

in another car and went to go apartment hunting. And I wasn't supposed to join IBM for like a month or two. I was going to take a breather and just hang out in San Antonio.

But I lent my truck to one of my friends there who wanted to pick up a mattress. And so he didn't know about this oil filter problem. And he was cruising down Loop 410 in San Antonio at about 60 miles an hour and drained the oil out of the truck, and the engine seized up. So here I am in between jobs with no transportation. And I called up IBM, and I said, hey. I've got this problem.

I need to buy a car. Can I start working there, like, tomorrow? And they said sure. But they changed whatever job they had in mind for me, and they put me in the 801 group, or what became the ROMP group, the 801 microprocessor group [ROMP stood for Research Office-Products-Division (OPD) Mini-Processor (later documents called it “micro-processor” but that term wasn’t used at IBM at the time)].

So I was the first member in that group because they shuffled people around, and they had this opening that they created for me so I could join early. And I had no idea what I was going to end up working on, but that was how I got into processors.

Krewell: Wow, so that's real serendipitous.

Johnson: Yeah, well, there's going to be a lot of that to come.

Krewell: So you were technically the first member of the 801 team.

Johnson: Yes.

Krewell: Wow.

Johnson: OK, so let me be specific. There was an existing 801 project in IBM Research, in TJ Watson Research. But they were doing, I think it was ECL-based, basically a mini computer. So it was mainframe sized, lots of large motherboard types of circuitry.

And what I was the first was what became the Office Products Division Mini-Processor, which was supposed to go into things like the Displaywriter, early office processing equipment. So the 801 existed as a mini computer. We were planning to spin it as an NMOS-based microprocessor.

Krewell: OK, so it was going to be, instead of discrete ECL was going to be integrated into one chip.

Johnson: One NMOS chip.

Krewell: NMOS chip. And 801 referred to--

Johnson: Building 801.

Krewell: In-- is that the one in New York, or is that in Austin?

Johnson: It's in New York [Yorktown Heights].

Krewell: So that's the original New York building.

Johnson: Yeah.

Krewell: OK.

Johnson: Yeah, their main research building there is building 801.

Krewell: So then the team started forming. And you were the first member of that team doing the NMOS chip.

Johnson: Right.

Krewell: And then so how did that develop?

Johnson: Well, we spent a year. It's actually kind of interesting looking back at it because once IBM decided they were going to do this chip-- I was the first member of the design team, but there were other architecture teams at the time. So there were quite a few people giving input to the design and at the time maybe two or three designers. Like Phil Hester is a name that's a reference.

So I had this kind of funny experience as a new grad in my first year with a lot of, I didn't know at the time, heavyweight IBM architect types coming to Austin because they're going to help architect this chip. So before I even knew who he was, I got to meet John Cocke [John Cocke was a well-known computing researcher who, among other things, developed the concept of RISC (reduced-instruction-set computer) that was the basis of the 801]. In fact, my manager called me into his office-- my manager's office-- once.

And here was this old guy holding a cigarette with about an inch and a half of ash, and he was trying to keep it from falling on the floor. And he starts asking me questions about how I would go about doing certain things. And I figured, at IBM, there was always somebody coming by to poke at what you're doing. So I gave him kind of a flippant answer.

I vaguely remember what it was about, but it had to do with some kind of bus error signal and how you might implement it. So I only later found out who I was kind of treating flippantly. But we used to have these meetings with 20 or 30 people sitting around a conference room, half of them smoking, all of them yelling at each other.

And I mean, that's a bizarre experience to begin with. And the other thing that I always thought was funny was any time I raised an argument, they would take me seriously. And they would argue with me, even though I at the time basically had no idea what I was talking about.

Krewell: But it was a new area.

Johnson: Yeah, it was a good experience getting exposed to that sort of personality I guess, even though you don't see much of that chain smoking around a conference room and screaming anymore.

Krewell: No, chain smoking is pretty much gone. You have to go outside to do your chain smoking.

Johnson: And even cigars, it was cigars, pipes, cigarettes. It's kind of funny thinking back on it.

Krewell: Different era. So how did that program develop? And your involvement in it, how long did you last?

Johnson: Well, pretty much, I would say starting the second year until I left eight years total into it. Well, IBM had this I guess they consider it a rigorous culture. So you develop the architecture. And then we built what's called a functional equivalent model. So it was implemented in TTL, and it was functionally equivalent. So it was these boards about like this, and it was six of them.

And it wasn't equivalent to the chip design, but it met the architectural spec. So we built a couple of those I think. And I ended up with the job of somebody wants to do the ALU. Somebody wants to do the bus. And then there's all this other stuff to make it all work together.

So I ended up being kind of the glue to make sure that this unit talked to that unit talked to this unit talked to that unit. And I wrote all the microcode for the thing. It was a RISC machine, but it was still microcoded. And it was mostly one instruction per one microcode instruction.

Krewell: Right.

Johnson: But it also had like the interrupt handling routines were written in microcode.

Krewell: Oh, OK.

Johnson: Exception handling, because IBM was very big into this what they called RAS, which was Reliability, Availability and Serviceability. So it wasn't just execute instructions. It was all this other infrastructure to detect errors and try to figure out what was causing them.

Krewell: So they had to check your architecture and all that stuff.

Johnson: Yeah. For example, one of the things that it did was when it powered up, it would send a signal on each pin and feed back through the pin into the receiver and make sure that the buses had integrity, for example, and report an error if there was a problem.

Krewell: That all ran on microcode?

Johnson: That was all run on microcode, yeah.

Krewell: So not state machines. You used microcodes to--

Johnson: Yeah. And it actually found-- when I first was required to do it, I thought it was kind of a silly thing to do. But it actually did find errors. And when it found them, you knew what they were instead of running for who knows how long.

Krewell: Good. Deterministic error checking. So you said you spent, what, roughly eight years at IBM?

Johnson: Yep.

Krewell: And followed the 801 project through to--

Johnson: Yeah, well I just mentioned the functional model.

Krewell: Yeah.

Johnson: The next thing we had to do was a nodal equivalent model. So that means that every gate in the final chip had a representation in TTL. So we would develop the netlist for the chip and then transform that netlist into these quad NAND/NOR/flop implementations. So this thing was 18 Augat boards with 384 modules. I mean, these things were giant.

Krewell: Augats are big boards with wire-wrapped pins in the back.

Johnson: Wire-wrapped, boy, do I remember the wire.

Krewell: Did you have to learn to wire wrap?

Johnson: Yeah, and I had to deal with all the-- it's a long story, but basically one of the engineers on the team thought IBM technology was second rate to things that were available with third parties. So he didn't want us to use-- he was kind of a senior guy that joined later.

He didn't want us to use the IBM wire-wrap wire, which was Teflon coated. It was actually really nice stuff. We used third-party stuff. And when it went around a pin, it would nick and short. And I don't know how many hours I spent. It wouldn't necessarily short. It would just kind of short.

Krewell: Oh, yeah.

Johnson: But anyway, I ended up being chained to that thing for probably a year, year and a half. Because since I knew how everything worked, guess who gets to debug it? Well, the ALU adds, and the bus sends and receives things.

Krewell: Well, it's a good learning experience.

Johnson: Oh yeah. Well, the nice thing about it is I had already had the NMOS experience and the TTL experience. I had to do some bipolar gate arrays in this process [of designing the nodal model] and all the clock distribution. And you can imagine clock skew across 18 boards is fairly interesting to manage.

Krewell: Sure.

Johnson: So I did all the clock distribution in ECL. So I got exposed to ECL design. So by the time I left, I had designed in just about every technology except CMOS, which of course became the thing that you designed in.

Krewell: Yeah, but in that era, CMOS was still considered slow.

Johnson: Oh, it was very slow. It was basically just coming on. Actually, NMOS at the time was significantly faster than CMOS, which is involved in how I left. But that will come.

Krewell: Oh. So tell us more.

Johnson: Well, so, the last few years, we had done this processor. And I had done a lot of the verification. It actually eventually did go into the PC-RT. But I got kind of frustrated with the slowness of IBM. And we were doing what became the PC-RT design, and the people designing the memory controller wanted to implement it in CMOS.

OK, so we had a 6 megaHertz processor. It was a 6 megaHertz, 32-bit processor, which may sound slow, but at the time-- this was 1983 or so-- was actually significant. Because at the time, the state of the art was, I don't remember the frequency, but 8-bit, 8088 type stuff from Intel. So the group doing the memory design or the memory interface design wanted to use a CMOS ASIC effectively.

And it would only run at 4 megaHertz. So I know 4 and 6 sound small, but 6 is 50% faster than 4. So through some interesting politics, I commandeered the design and used a combination of bipolar gate arrays and some TTL clocking and got the design up to 6 megaHertz. But the memory interface group kept control of the memory card spec. At the time, it was card on board type of packaging.

So I was doing the processor, memory controller, and other people were doing the MMU. And the interface of the memory card came out on my stuff, and the specs started changing. So I had committed up to the lab director that I would meet 6 megaHertz, and all of a sudden, the RAS-to-CAS delays and those sorts of things.

I don't remember all the terminology, but I met it, then it changed. And I met it, and it changed. And I met it, and it changed. And I finally said, I want to go somewhere else, so I went to IBM Research. And in the process of that, I was basically sitting in my living room one late Sunday morning, and I happened to pull down one of my DSP textbooks. And I hadn't looked at DSP in eight years.

And I looked at this book, and I said, gee, I need to do something else besides what I'm doing. Because I'm really working hard, but it's just you're fighting the politics. And it turned out it was going to be another two or three years before the system actually shipped. And everybody else was starting to do work stations based on 6800s and 68000s.

And we were supposed to go into the original IBM PC, but they picked the 8088 and of course basically enabled Intel. And so all of that, just as I was looking at this book, I said, I don't remember anything about DSP. I need to go back and get my PhD so I can learn more. And so I asked to be transferred up to the IBM Research Center just to get exposure to basically the academic community.

Krewell: And the research center up in New York?

Johnson: Yes.

Krewell: In Hudson Valley.

Johnson: Building 801 is actually where I worked. So there I got academic contacts to MIT, and Berkeley, and Stanford, and a few other places. And at the time, IBM had this graduate fellows program, which was really a great deal. They would move you to the area, pay your salary, pay you to go to school, effectively, almost regardless of how long it took. As I said, a really sweet deal, so that's what I was originally planning to do.

But to make a long story short, while I was there, one of the people that I dealt with on a day-to-day basis was a Stanford grad who had gone to Stanford with Paul Chu. And who was at AMD, so Paul Chu called him to ask if he knew anybody who had a RISC processor background. I'll get into why when we get into 29K.

Krewell: Sure.

Johnson: So I come back. I moved back to Austin because the deal with IBM Research was temporary. And basically was in a mood to leave. And in fact, some of the people I had talked to at IBM Research had counseled me to leave. And at the time, nobody left IBM. You joined, and you retired, and you got a pension.

Krewell: Job for life.

Johnson: So just basically the frustration built up. The visibility of what I was working on grew and grew, but my sort of visibility as part of it was getting less and less. So IBM did me a favor and installed an answering machine in my office. And so Paul Chu called and left a message on my answering machine.

It's kind of a long story, but I came back to the office very frustrated because of a situation. They were having a big demo for a bunch of managers and executives and told me I couldn't participate, but I should wear a suit in case something went wrong. So I came back spitting nickels-- spitting nails.

Krewell: You couldn't participate, but if something broke, they wanted you to go in there and fix it.

Johnson: Yeah, you can't stand by it and explain how it works, but it might break while our golden boy is explaining. So I came back, and I turned on the answering machine. This is Paul Chu. And I sort of knew right away, I'm leaving IBM and going to AMD.

Krewell: So it's like that moment it just struck you, this is it. I got to make a change.

Johnson: Yeah. So I still had my applications out to MIT, and Berkeley, and Stanford. And I left IBM, which is kind of an interesting experience. They won't let you-- when they are convinced you're leaving, you'll get the about an hour or two meeting a succession of managers who will give you, if you want more pay, we'll give you more pay types of stuff.

And after you've turned them all down, your manager comes and walks you out the loading dock, because they don't want anybody to see you being led out the front door, so they take you out the back door. And I couldn't go back in the building after that.

Krewell: Oh really? They walked you out, and that was it, and you were done.

Johnson: Yeah. So we were supposed to move. This was right after Christmas. We were supposed to move. I left I think January 3, and my wife was all antsy about moving to California and so forth. So we had a long discussion, and I said, well, I'll go out by myself and figure out how things are, and we'll move later. And right after we decided that, the moving van pulled up. And we had actually canceled it, but somehow the message didn't get to the van company.

So we said, OK, we'll move. What the heck. So we moved out and as we were relocating stayed in quarters here in the Bay Area. And I still remember one day I got a letter from Berkeley and a letter from Stanford. And I opened the one from Berkeley. Oh no, we regret to inform you. And I said, gee, if I didn't get into Berkeley, I'm for sure not going to get into Stanford.

And I opened the one from Stanford as, congratulations. And it was copied to the IBM fellowship coordinator or whatever. And I said, well, this is interesting. I've been accepted to Stanford. I have a new job. I have kids one and three.

I don't have a house. I still have my house in Texas. But sure, I'll go to graduate school. So that's how I ended up going. I didn't actually join-- this was January. I didn't actually start Stanford until the fall.

Krewell: But you had accepted a job at AMD.

Johnson: Yes.

Krewell: And they were agreeable to you going to Stanford to get your PhD.

Johnson: Yeah, and in fact they paid the tuition.

Krewell: OK. So that, you had already negotiated that with Paul Chu and AMD by that point in time.

Johnson: After the fact, yeah. But they were pretty open to it.

Krewell: OK.

Johnson: Somebody-- I think I actually wrote this in the preface to the book or something. Somebody said, never, ever have a family, go to graduate school, and work full time. But that was what I was doing plus carrying two houses in the end because I couldn't sell the one in Texas, and the one here cost about eight times what the one in Texas was worth.

Krewell: Wow.

Johnson: But the good thing is, yes, I was working full time. I was actually travelling, doing marketing pitches, or with marketing doing pitches. But the Stanford experience was very synergistic with what I was doing at AMD. So it was in a sense two full-time jobs, but there was a lot of overlap. So I was able to use knowledge, actually knowledge and getting to know people and things like that.

Krewell: So who was your adviser at Stanford?

Johnson: Mark Horowitz.

Krewell: OK. So I assume then your research at Stanford revolved around RISC processor design and--

Johnson: Well, I had already had a lot of experience with processor design. I wanted to learn more about the systems aspects. So operating systems, compilers, more basic exposure to semiconductor technology and that kind of thing. I really didn't need the design.

I had done so much design at IBM that I was pretty comfortable with my knowledge there. And it was more the related fields, like I said circuits, compilers, operating systems, programming languages, those sorts of things.

Krewell: OK, so kind of the Gestalt of the whole thing.

Johnson: Yeah.

Krewell: So, but meanwhile you had accepted a job at AMD. So let's talk a little bit about that experience. You talked to Paul Chu and whoever else. And was that the 29000 program, or was that before?

Johnson: Yes. It's what became the 29000. At the time it was called SIP, which stands for Streamlined Instruction Processor. And the objective was-- I know there's a lot of confusions, and you mentioned earlier about the 29K about what we were trying to accomplish. But what I was given to believe we were trying to accomplish and what we were focused on was to continue a progression of AMD's what originated as bit slice products, which are little four-bit slices of ALUs, and registers, and that kind of thing.

Krewell: Building blocks.

Johnson: Building blocks. But those building blocks had been replaced by what they called functional slices, which are 32-bit data paths.

Krewell: Right, the 29C300 family-- the seamless version was a bipolar version.

Johnson: Right. Yeah, that's another funny story. At product planning at AMD at the time, you get up and present a product concept. And there were two groups, the bipolar group and the CMOS group. And the

first question was, is this bipolar, or is this CMOS? And it's like, there were other questions besides what technology it's implemented in.

Krewell: But that was actually-- because AMD was a manufacturing company.

Johnson: Yeah.

Krewell: That was actually important. That was an important thing, whether it was bipolar or CMOS. AMD went for a period of time when bipolar was their primary solution to a lot of problems and then switched to CMOS eventually, and NMOS in one way too.

Johnson: Right, it was just interesting to me that that was the aspect that was most important to the product.

Krewell: Yeah, well the bit slice products were started in bipolar.

Johnson: Oh yeah. And the early functional slice were bipolar also. So these were nice parts. I'd actually used bit slice parts in this functional equivalent model that I mentioned at IBM. I had been exposed to AMD. Kind of another funny story, when AMD first moved to Austin, I was still at IBM, and they used to advertise on the radio about great job opportunities at AMD.

And I used to think, gee, I'd really like to work for AMD because I'm really impressed with these bit slice products, but they'd never hire somebody like me. It was just the concept of working in a company like that for some reason, I thought they were up here, and I was just basically designing in TTL type stuff.

But anyway, back to the objective of the 29K, they wanted the flexibility that you got from the functional slice and the bit slice, but they wanted it to be a target for a C compiler. Because the big problem people had was that it had to be effectively micro-programmed, which is what we had done at IBM. But it's not a very efficient way to develop and maintain programs.

So our job was to define an efficient target for a C compiler, which is why most of the features in the 29K were there. It was the RISC philosophy in terms of making it an efficient target, which is why we had so many registers and that kind of thing.

Krewell: But there were other models. There's MIPS, and there was SPARC that were still being developed at that period of time or were available as models.

Johnson: Right.

Krewell: Did you look at those architectures as well?

Johnson: Well, MIPS was actually very similar to what I had done at IBM. In fact, some claim that-- maybe this is a little out of school. But some people claim that the MIPS idea was basically a derivative of some interaction with IBM Research.

Krewell: OK.

Johnson: Which is actually one of the things IBM Research did at the time was kind of get universities involved in things that they thought universities ought to be involved in.

Krewell: So the concepts of the 29K was-- but somewhat similar, how did that develop?

Johnson: Yeah, I knew I had something I wanted to circle back to, and I was trying to remember. We did look at those architectures. But the reason that the-- OK, so we knew about SPARC. We knew about MIPS. But the register file for the 29K actually has this property that it reduces the frequency of external loads and stores by a significant amount, which, for the embedded market we were thinking that most of the interface would be to things like DRAM.

So both SPARC and MIPS assumed external caches, and the load latency wasn't a big deal. The 29K stack cache is kind of-- let's say it's a very rudimentary cache. It actually caches a fair amount of the activity that you would see to the L1 [level 1] caches. So it wasn't just something we did to do it. There was actually a reasoning behind it.

And it had advantages against SPARC because the windows weren't fixed. They were actually variable. And this is something that a lot of people miss that you only allocated the amount that you needed. And you could basically manage it purely like a stack. And with respect to MIPS, MIPS was basically spilling and filling a fixed number of registers because of the procedure linkage calling convention.

Krewell: But you had a total of I think 192 registers if I remember correctly.

Johnson: Yeah.

Krewell: Which was a very large amount of registers for that time, full 32-bit registers.

Johnson: Yeah.

Krewell: So that had to take up a fair amount of die area. So that had to be an interesting trade-off when you said, how much die area are you going to allocate to register file compared to other functions in the part?

Johnson: Yeah, it was actually-- the die was about a quarter registers, about a quarter functional units, about a quarter the branch target cache, which is something we haven't talked about yet, and about a quarter MMU. And all of those were designed-- the register file may have been let's say the most robust with respect to what other people were doing. But in the other areas like the branch target cache, it had the same philosophy.

It was caching as much as you could on chip and being as effective as you could with that on-chip cache without requiring the expense of an off-chip cache. Because we were expecting effectively a DRAM design for the external memory. And the memory management unit was software reloaded, which MIPS had a similar type of concept mainly because from a flexibility standpoint you didn't want to force customers to have a specific page table architecture.

At the time, there were a lot of different alternatives to the page table architecture. And also, MMUs, in a lot of people's experience-- and I'm not sure why this is-- the Motorola 68000 had real issues with managing the die growth of the MMU and the complexity and verification. We had a lot of 68000 people on the design team. Effectively the only way we get them to implement an MMU is to say, it's not nearly as complex as what you're used to seeing. But it was still-- I think it was effective.

Krewell: Well, an embedded MMU may not be as important because at that time, a lot of code was not paged or--

Johnson: Right, the main issue was memory protection, not demand paging. In fact, I doubt anybody ever did a demand page structure. But it was definitely used in the embedded applications for memory protection.

Krewell: Yeah, if you wanted to run Unix on top of it, you would have been demand paging.

Johnson: Right.

Krewell: So it's interesting because there was a disconnect with certain upper management. And I think specifically I can remember a poster, and it talked about 29K the next platform. So there was some

disconnect between the engineering group and your vision, and that clearly was the way things were going was this is an embedded processor. But there were some people in upper management at AMD thought this was the next Unix platform that AMD was going to develop.

Johnson: I think it's maybe somewhat more complicated than that. I only discovered this after the fact, to be honest. At the time, AMD was having a lot of issues with Intel. When IBM picked Intel, they forced Intel to pick a second source. First of all, they gave Intel a lot of capital as kind of insurance, and they made them enable another party to be able to build 8088s to start.

And I suspect that Intel picked AMD thinking they would never cause them any trouble, effectively. Because as you said, they were a bipolar company. At the time, they were doing Zilog processors and other licensed types of products and would not be an obvious competitor to Intel. But AMD actually was very aggressive and did a pretty good job beating Intel to market in some cases.

So Intel was starting to push them away. And I think AMD management saw the 29K as something they could position against-- let's say might replace x86 at some point. I was never part of those discussions. And as I said, I only found out through the grapevine, so to speak, after the fact. And if that's what we were trying to do, we weren't designing the right part.

Krewell: So that thought, that platform architecture, never influenced the design on the 29000.

Johnson: No. Well, for a period of time we did an off-chip cache part that I never really quite understood why we were doing it. And it wasn't instigated by me. I did it. I worked with some other engineers to define it, but it never quite understood what we were doing, why we were doing it. And based on what I found out later on, I suspect that it had to do with wanting to position against Intel.

But the thing is, there were a lot of people doing computer type products or platform type products. Engineering workstations were the big thing. Personal computers were everywhere. Why would you be one of five or six people doing yet another computer when the embedded market had some significant performance requirements, but they couldn't afford all the cost of a computing platform?

Krewell: I think that worked out really well because the 29000 became a successor to the 2900 bit slice and the 29300 Family of horizontal slices. And a lot of that customer base went to the 29000 eventually. And it became a good embedded product. And platform-wise, it would have been swallowed up trying to compete with MIPS, SPARC, Power, and the others.

So tell me a little about the early 29000 team. Who was on it?

Johnson: Oh, Brian Case, Smeeta Gupta, Tim Olson, Phil Freidin, Dave Sorensen. I hope I haven't left anybody out, but--

Krewell: Well, you can always add later. [Gigy Baror was also on this architecture team.]

Johnson: --I think that's about it.

Krewell: I didn't realize Sorensen was on the team, because he's a field application engineer.

Johnson: I'm talking about the very initial early team. He did-- oh, Rod Fleck was another.

Krewell: Rod Fleck, OK.

Johnson: Both Dave and Rod left. Did I mention Brian Case?

Krewell: Yes, you did.

Johnson: OK, so they all left pretty much after the spec was done and went to do other things.

Krewell: Yeah. And Tim Olson didn't stay too late either. Did he--

Johnson: Oh no, he stayed.

Krewell: Oh he did stay through the whole thing?

Johnson: He moved to Austin when the team moved to Austin in '89. But he was there I would guess maybe '94, '95.

Krewell: Oh, OK. So he did stay quite a long time. So this project started developing. You had some lead architecture, and then you started going to implementation. Most of that work was in Austin, Texas, correct? The whole team?

Johnson: All of the implementation was in Austin, Texas. And the architecture, product planning-- what we called product planning-- was in Sunnyvale.

Krewell: And then how did you map out the architecture instruction set, finalize the size of the number of registers? Was that basically limited by die area? The 192's sort of an unusual number, so it wasn't 128. You had some fixed registers, plus--

Johnson: Yeah, we wanted 256, and so we hacked where it was more efficient to hack. They were organized as global registers and local registers, somewhat mapping to global variables and local variables in a stack. And the global registers weren't quite as useful as the local ones. So when it came time to whack, we whacked the global registers.

Krewell: And you whacked them, just was it for die reason?

Johnson: For die reason.

Krewell: Yeah. Did you have a target die size, and then you had to fit everything into it? Or is it just kind of grew and then you figured, well, that's good enough?

Johnson: There was a target, but at the time, it was very hard to predict where you were actually going to end up. So it was accepted that things would grow somewhat. I mean, it probably still is. There's so much going on that you can't really-- there's so many variables that go into die size you can't really necessarily predict where you're going to end up.

Krewell: One of the truly unique parts of aspects of it was the branch target cache.

Johnson: Right.

Krewell: That was a very cool idea. Can you explain a little bit behind it?

Johnson: Yeah. It was originally an instruction cache. And a cache of sufficient size really wouldn't fit on the chip. And we were arguing among ourselves in one product planning meeting, and I made the offhand comment that it's kind of a shame that we have so much memory there, because mostly what it does is cover the latency to fetch branch targets.

And that's as far as I took the idea. And Phil Freidin the next day said, well, if that's the only thing it's good for, why don't we just cache branch targets in the cache? And I said, oh yeah, I never thought of that. So once you see it, it's a very cool and obvious idea. But a lot of cool ideas are obvious only after you see them.

Krewell: Yeah. So functionally, if you got a branch, rather than waiting, stalling, waiting for the DRAM to access, you have a short four bytes was it, or four words?

Johnson: Four instructions.

Krewell: Four instructions that you could access from the beginning of the branch that you were taking while you wait for the warm up the DRAM to get access to the rest of it.

Johnson: Yeah.

Krewell: Yeah, that was quite cool. So it gives you the low latency of a cache but without the penalty of die area.

Johnson: With a much smaller die overhead, yeah.

Krewell: So OK, so as the 29000 program was coming together, the instruction set, was there anything unique about putting the instruction set together? It was two source, one destination-- two source, one destination architecture. Everything was register-based, pretty classic RISC architecture.

Johnson: Yeah. At the time, all RISC instruction sets were 32 bits, except for what we had done at IBM. That was actually a precursor to what came later in things like ARM with Thumb. At IBM, one of the big changes we made from the 801-- the 801 was actually a 24-bit architecture, of all things. So when we moved to 32 bits, the Office Products Group was very concerned about what they called byte efficiency, which is how much you can encode in a certain set of instructions.

So they insisted we have 16 and 32-bit instructions. But the rest of the world was doing only 32-bit instructions. And that's what we did with 29K. And when you have 32-bit instructions, 32 is actually more bits than you really need. So where most of these RISCs differentiated at the time was what they did with all the bits that they didn't need. But I can't think of anything in particular that was unique about the instruction set.

Krewell: It was fairly straightforward, standard RISC instruction set. But the unique aspects were the large register files. So in theory, you had eight bits per register file address.

Johnson: Right, well that's where we used the bits that you don't-- other people were wasting them on other things like predication and whatever. We wasted them, if you want to call it wasting them, on register identifiers.

Krewell: Was there any microcode in the 29K? Was it all state machine?

Johnson: No, not at all. All state machine.

Krewell: And then so how did the chip development take place? You had the architecture put down. How did the chip itself progress?

Johnson: Well, it started in '85. And it took I would say about three years and maybe four revs till it was functional. I don't think there was anything-- it wasn't unusually long or unusually short for the time frame. It had some issues when it went into production. Basically the branch-- it's a very long story, but I guess I might as well tell it, because it's kind of a funny story.

Krewell: OK.

Johnson: Basically the branch target cache, there were two sets. And if neither sets hit, it would tri-state both cache outputs. And the instruction buffer would go metastable as a result. Metastability means the latch basically chases its own tail. And the one symptom of this is it would flip a bit in the stack pointer into the stack cache.

OK, so what would happen is we found this, we called it the Prem bug because there was a guy who had joined the team called Prem Sobel. He wrote this Towers of Hanoi program, and we ran it. And we never could get it to work. It would always crash. And the reason it would crash is that it's a very recursive program.

And because of this metastability, you'd get to a point, and the stack would jump up by 4k locations and keep growing. And when it came back, it would find basically a bunch of junk on the stack and crash. And around the time I was moving back to Austin, the whole team was tearing their hair out with that. We eventually actually did a spin of the part without the stack cache [actually, without the branch target cache, not the stack cache] called the 29005.

Krewell: Oh, wow, OK.

Johnson: But my recollection is it took us about nine months to figure out what was going on there. But other than that, everything was fine.

Krewell: Yeah, aside from that one minor problem. What other unique things do you remember about the 29000 program?

Johnson: Traveling all over the country talking to teams, which is something I'd never done at IBM. I was actually surprised that AMD would let me out of the house, so to speak. Because at IBM, engineers never, ever talk to customers, at least not any that I knew. And at AMD, pretty much you had to go talk to customers, which is much better, in my opinion.

But it was something that was just really struck me that you could go talk to Digital Equipment and Data General, and at the time Wang, and all these companies that were famous. And you could talk to famous people and effectively get to learn the industry just by being an engineer and not a salesperson or marketer.

Krewell: Well, it was part of the-- because Jerry Sanders was such a sales-oriented executive and the company itself had a very sales-oriented culture, it drove everybody to be more outward and more customer facing. But that was an interesting experience. I didn't realize that had an impact on you.

Johnson: Oh yeah.

Krewell: So the 29000 comes out. It's pretty successful. There's a lot of design work going on. And we had started working on the next generation part. How did that develop?

Johnson: Well, we did, I would estimate, maybe four or five follow-on products. So--

Krewell: Before the K5 days.

Johnson: This was before the K5 days. The original 29000 bus, because we had been poking at Intel-- this is my read of the situation-- Intel will always position something against their competitors. And so they won't let anybody, especially AMD, have a product that they don't have some kind of response to. So in the case of the 29000, Intel picked the i960, which-- we don't need to go into the background-- but it wasn't meant for what they were positioning it as. But they had something to position.

Krewell: Right. And in fact, they also had 860. And they actually swapped their positioning.

Johnson: And a 432. Intel has all kinds of things in their legacy. So Intel launched their marketing campaign against the 29000 bus because it had three buses, which is something we haven't talked about. But it was an interesting-- at the time, other products had Harvard architecture, instruction address, data address, instruction memory, data memory.

And we had one address bus for both. And then it was kind of a unified address and then split instruction and data. So you get the effect of a Harvard architecture with fewer pins, effectively. And Intel other processors at the time just had multiplexed or demultiplexed address, and they were all [both] instructions or data. Intel would go out and just hammer us with customers.

And of course that would come up back through the sales force, and they would hammer on my management and say, man, you really screwed up, didn't you? And at the time, the 29K was actually the highest instructions per second per square millimeter of silicon. It was absolutely the most efficient design at the time.

But the reason for that was the bus architecture, and the stack cache, and the branch target cache. And so everybody started just whining, effectively, that we had a crappy bus architecture, if I can use that word.

Krewell: Well, it did require more complexity in the board design.

Johnson: Right.

Krewell: Because you had a multiplexed address bus. You had to demultiplex.

Johnson: The issue is what do you get for that complexity.

Krewell: Right. But for many designs, typically in that day, you had separate instruction and data buses and memories.

Johnson: Right. So anyway, the point is we ended up doing a product that grew the caches only because we had a more advanced process technology. So initially we put on a real instruction cache, if you will. And that allowed us to remove the split instruction data bus. And we cleaned up a few protocol things. And that became the 29030.

We also had started to become fairly successful in the laser printer market. So we did a series of chips that were I would say initial, what would have become SOCs because we had the processors plus peripheral logic where the peripheral logic was dedicated to laser printers. And we had three different versions of that, kind of a low, mid-range, and high end.

We tried to do a superscalar product, but that never really-- that eventually got I would say de-emphasized in favor of an x86 design using the same concepts.

Krewell: But you missed the 29050, which had the floating point unit.

Johnson: Oh, right, yeah. That was actually done very close to the 29000. And I didn't do that. I was finishing my PhD at Stanford.

Krewell: OK, so after the 29000 was done, there was a different development that you didn't-- because they took the 29C327 floating-point core as I understood it.

Johnson: Right, and the lead architect for that was Bob Perlman. And he actually led the 29050. I didn't mean to omit that. I thought you were asking about things that I worked on, and I don't consider that to be-- it wasn't a part that I would take direct responsibility for. It was a fine part. In fact, it's in still in lots of 777s as far as I understand, in the avionics. But that wasn't why I didn't mention it. It's just that I was off trying to finish my research.

Krewell: Well, but that's a clarification on the 29000. I didn't realize that you had no involvement on the 050. That was different.

Johnson: Very little. I would spend maybe two days a week at AMD arguing with people about what we should be doing. But other than that, it was really more an advisor.

Krewell: OK. So you finished up your PhD at Stanford, and where does the blue book come into play? This is post your PhD or while you were doing your PhD?

Johnson: I wrote the book afterwards.

Krewell: Yeah.

Johnson: But it's based on the research that I did for the PhD.

Krewell: The title of the book was--

Johnson: Superscalar [Micro]Processor Design.

Krewell: Wanted to get it out there. And interestingly enough, at the very tail end of that book, you posited the idea that you could do an x86 with a RISC core.

Johnson: Right.

Krewell: And that actually, just sort of like a throwaway at the end of the book. You talk about all this stuff, and then it's just sort of in there.

Johnson: It was a throwaway. Basically that wasn't part of the original plan. It was put in there for let's call it marketing reasons. Basically more people would read the book if it had some mention of the x86.

Krewell: Really.

Johnson: Yeah, really. So I had to actually dig down to get the manuals to try to figure out how the x86 worked. And that was the first time I'd ever actually paid any attention to it, to be honest.

Krewell: Interesting. So, actually we'll jump to the K5 in a little bit, but is there anything we should wrap up on the 29000 program or the 030?

Johnson: I think we've covered most of it.

Krewell: So this little throwaway, now, how did that then become the K5? Or was it completely unrelated, the throwaway and then the K5 idea was completely separate?

Johnson: I would say it's completely separate.

Krewell: OK, because it seems like they were related, that you were talking about it in the book, and then the K5 program takes off based on the similar concept.

Johnson: They had the same topic, but they had nothing to do with each other.

Krewell: OK, so how did the K5 program, how did you get involved in it? And the K5 is the first clean room x86 processor AMD ever did. Well, not clean room, because it was a bottoms-up AMD-designed x86 processor.

Johnson: Right. Well, at the time, OK, so Intel's [actually, AMD’s] doing reverse engineered x86 processors. After the 286, the 386 and the 486, the agreement with Intel was that AMD would effectively get the designs and would be unable to manufacture them. So they'd get the mask sets and run them

through their process fabrication facilities. [“Unable” to manufacture refers to the fact that the mask sets were not of the correct design revision, so the products were not usable.]

But they were doing so much better than Intel [before the 286] that Intel got into the mode of just not cooperating by giving them the 386 and of course not the 486. So AMD developed the strategy of reverse engineering, which was to have two design teams taking pictures of layouts and separately reverse engineering the circuits, creating the schematics, and then comparing the schematics to make sure that they had inferred the circuit structure correctly.

And of course they realized-- it doesn't take much to realize-- that you might do that with a 386, and they did. And you might do it with a 486, and they did, although there were a lot of issues related to it.

Krewell: Well, there was a big issue with the microcode, because as much as you get all the circuitry done, the microcode has a lot of the special sauce built into it. [It’s also covered by copyright, which is a different legal issue than a patent license.]

Johnson: Right. I know. I managed the microcode clean room. That's maybe not a well-known fact. It wasn't supposed to be known, but I guess the statute of limitations has run out.

Krewell: Yeah. I think AMD had a clean room version of the microcode in development in case they couldn't get rights to the Intel microcode.

Johnson: Right, and I was responsible for staffing that, and managing it, and keeping them clean, so to speak, making sure nobody talked to them that shouldn't talk to them except for me. So we started this discussion by the situation that AMD was in was they knew they needed to get out of this reverse engineering business. But at the time, PowerPC had started. Alpha was claiming they were going to take over the world. And MIPS was claiming they were going to take over the world. And SPARC was claiming they were going to take over the world. So there were all these companies doing RISC processors, and we're going to beat Intel. There's no way they can keep up, blah, blah, blah.

And so AMD got into this indecisive mode where do we continue on the x86 path somehow, or do we license an Alpha or license a MIPS, or license a PowerPC, or even just do an original implementation? Because some of those RISC processors don't really have a lot of IP protection to the instruction sets. So that's where I started becoming involved.

It wasn't, go do an x86. It was, help us answer this question. We know you like RISC, but we really want you to be honest about answering the question of whether RISC can beat x86. And at the time, of course, I had spent my whole career basically doing RISC processors.

Krewell: Right.

Johnson: But the more I looked at the question, the more I realized that there's nothing fundamental to x86 that says it can't keep up to some level of performance with the other processors. And at the same time, Intel had lots of capital to spend in process technology and circuit design. So architecturally, there was nothing they could do or not much they could do.

But in terms of raw megaHertz types of performance, there was plenty they could do. And it never seemed to me that there was going to be more than about a 30% advantage to RISC. And these days I'm not sure there's any at all. But at the time, the standard was you had to have twice the performance to break Intel's monopoly.

And it was sort of funny, because I was expected to be a RISC proponent, and I ended up actually arguing with AMD management that we shouldn't pursue the RISC path. If your problem is personal computing, you ought to find a way to stick with x86. The problem is that my research was finished in 1989, and because of this analysis paralysis, if you will, we actually didn't decide to do anything until '92.

So in retrospect, we lost three years analyzing this problem. And at the time, Intel was taking about five or six years per generation up until the fifth generation. And then starting at the fifth generation, they were reacting to the PowerPC, which was IBM, Motorola, and--

Krewell: Apple.

Johnson: Apple. And they were really concerned about PowerPC. They weren't that concerned about what Alpha might be or what MIPS might be. And their response, as I said, was to invest loads of capital into shortening the development cycles and increasing the frequency growth curve that their processors were on. Any meanwhile, AMD was, do we do an original processor or not?

So once we did start, we had three years compared to what Intel's timelines were going to be, not that we knew it. And AMD's process technology was-- AMD was always short for capital, so we were partnering at that time with Hewlett Packard. So our process technology was three levels of metal, and no trench isolation, and really a very basic process technology for the problem that we were about to be faced with, except that we didn't know we were going to be faced with it.

We used to argue with marketing guys that the design shouldn't be more than 66 megaHertz. It ought to have a 486 bus. It ought to have a unified cache. And if you know anything about a four-issue superscalar x86, you don't want to have a unified cache. Trust me.

Krewell: No, definitely not.

Johnson: But the 486 had it, so these are the levels of product planning discussions we were having. So 486 bus, we actually started with a 486 bus. We didn't have a unified cache. I actually won that argument. But at the time, because AMD had been reverse engineering x86 products-- a lot of the circuits that Intel had implemented were dynamic and had lots of hazard conditions.

And to avoid any circuit reverse engineering problems, AMD had made all of their circuitry static. So it ended up having the advantage that with AMD products you could stop the clock because they were a static design. Intel, you stop the clock and they'd blow up or something, because you end up with a crowbar current and those kinds of things.

So AMD actually became pretty successful in the early mobile market because you could stop the clocks on AMD products. So we got the K5, and the emphasis was going to be mobile. So it had to be 66 megaHertz, and they actually liked the fact that the K5 had a very high IPC [instructions per clock] rate, even though it was a low clock. And it was--

Krewell: That should be more efficient.

Johnson: It was more efficient, but sort of God help us, it was a four-issue superscalar with a five-stage pipeline. And that thing wasn't going to run much faster than 66 megaHertz. But the trouble is, by the time it actually was approaching production, first of all, we had eventually punted on the 486 bus and adopted a real Pentium bus. We made all kinds of changes to the cache architecture.

Krewell: And the Pentium bus was much more robust. It was much wider than the 486 bus, so you had more bandwidth.

Johnson: Yeah. So I don't think there's an appreciation for the fact that, OK, we had three years. We started with zero development infrastructure. There was no spec for the part. We effectively had to reverse engineer the spec. There was no verification the way you do verification of original designs. The verification was two people taking photographs and comparing the results.

And in those three years, we must have restarted the spec in a major way at least twice. And we pretty much made the schedule or came very close to it. But our product was 66 megaHertz. And I'll never forget this, because the same people arguing 66 megaHertz, by the time the product was literally coming out of the pipeline, the development pipeline, oh, it's got to be 130.

Mike, just go back and figure out. You see there's this huge disconnect. I can't remember how many times I heard, well, look, the IPC is high. It's just the clock is low. If you could make the clock high at that IPC, it would solve all of the world's problems, which it would. But at that time, by that point, it's locked. You can't make that kind of a change.

Krewell: Right, and then coming from your background on the embedded side, and using the 30% performance boost per clock over the Pentium at the time was based largely on SPECint. I remember you did a lot of benchmarking on using what was [a] traditionally common benchmark

Johnson: Right. [With compiler optimizations, the advantage was 50% over Pentium on specInt: 30% was using the results of standard x86 compilers. SpecInt permitted compiler advantages to be included in the benchmark number, similar to Dhrystone, and the underlying RISC implementation of K5 exposed more optimization opportunities].

Krewell: But the PC market had adopted these PC benchmarks, which were completely different.

Johnson: It had adopted them right towards the tail end of our development. So the workstation world was using SPECint. All of the benchmarking that everybody did at the time was based on SPECint.

Krewell: Yeah, across the platform.

Johnson: We did fine on SPECint, but WinBench was based on Windows code. And I'll never forget some of these traces people started bringing me because Windows is built abstraction upon abstraction upon abstraction upon abstraction. And you'd see these traces basically push, call, push, call, push, call, push, call, push, call, push, call, add, return, pop, return, pop, return, pop, return, pop...

And I'd go, you're kidding me. All you're doing in that kind of a code is your instruction cache is just banging against itself because these procedure addresses have no relationship to each other. So there's no locality, and our cache design wasn't up to that style of code. So we had one more fix. Actually, after we were in production with a de-rated part, let's call it.

The SSA 75 I think is what it was called. So we made what was called the big Icache fix. And I don't remember the contents, but I still remember this massive email that I wrote because everybody was saying, man, we don't know if this change is going to work or if it's going to meet the performance requirements. So I had to lay out in this long argument why it was going to be a very robust change.

And it would take us I think six to nine months. But we could be guaranteed that we'd meet this WinBench target. But the frequency was always an issue. And there's really not much you could do about that.

Krewell: Yeah. It's interesting that the initial positioning was that it could be a mobile part. Because as I remember, its power is one up actually because I think it was trying to push hard to get more frequency out of it. So that push process wound up making it a little leakier and a little higher power.

Johnson: Yeah, It was supposed to be a fairly low-frequency design. I think it was in 24 stages per clock, low frequency but reasonable IPC at that low frequency and static design, of course.

Krewell: Yeah, but bottom line though was unfortunately it didn't meet marketing requirements compared to the Intel's Pentium, which was very aptly but also kind of derivatively described at one session as two 486s bolted together.

Johnson: Which is what it is, yeah.

Krewell: Yeah, essentially. And it wasn't a regular pipeline for each. It was an asymmetric pipeline between the two. But it was two CISC processor cores with a two-issue instruction dispatcher was basically what it was. And it had the white bus, Pentium bus, so it could take multiple instructions at a time.

Johnson: But it had the clock.

Krewell: Yeah, but yeah, it had a deeper pipeline.

Johnson: That's why our marketing folks were reacting so strongly. Because Intel, 130 is better than 66, right? And that's the extent of it. And one thing that amazed me is we had a product that ran at 90 megaHertz, and it was only something like 3% slower than the Pentium at 130 [on Winbench]. And the difference in price between theirs and ours was $320 some, for 3% performance because it had 130 and ours had 90 on it. There's a lot of money in a three-digit number, I guess.

Krewell: Yeah, well, but from a die point of view, I think for transistor count, K5 still had a higher transistor count. It was more complex.

Johnson: Oh yeah. I'm sure it did.

Krewell: But did you know that Intel was simultaneously doing their own internal RISC processor that became the Pentium Pro?

Johnson: No.

Krewell: So you were completely unaware that the Pentium Pro program, which basically was the same idea-- it was a RISC implementation internal but with an x86 front end-- was being developed. And in fact, I had heard from a couple people at Intel that the blue book was found in Oregon in a few prominent desks.

Johnson: Oh, I've heard that Intel engineers were required to read it. One of the reasons I wrote it is to light fires under the right managers at AMD. And unfortunately, it kind of backfired. I never expected Intel to pay much attention to it.

Krewell: Yeah, well they did. So you had no idea that there was a Pentium Pro program going on that was basically RISC internal core.

Johnson: Not at that time, no.

Krewell: Which then eventually became the Pentium II.

Johnson: Mm-hmm.

Krewell: So the K5 unfortunately had these problems competing in the PC marketplace because of lower clock speed, even though it had good IPC and all that. So that was hurting AMD's business prospects. So they had to build that fab in Austin, Fab 25. And the amount of K5 and the old 486 volumes weren't filling it.

So Jerry decided to buy NexGen. Did you have any input on that, or did you know the NexGen team at all? Had you talked to them, because they were also doing a RISC internal design as well.

Johnson: Right, I knew the team professionally but not-- I was kept out of that loop because of IP issues, effectively.

Krewell: Data isolates--

Johnson: Yeah, I read it in the paper--

Krewell: Really

Johnson: --on a Saturday morning. And at the time I was on K7, which I'm sure we're going to talk about.

Krewell: Yeah.

Johnson: And so I read in the newspaper, AMD's acquired NexGen, and Vin Dham who had been at Intel and was at NexGen is now taking over the K7. So of course my phone's ringing all morning with, OK, Mike. What's going on? You're in charge of K7. I said, beats me. I'm reading it in the paper just the way you are.

Krewell: Well, what was the AMD version of a K6? Because you had K5, and then there should have been in theory a K6. Or was your K7 really K6 at the time?

Johnson: Our K7 was what was coming after K5, not that we named it that.

Krewell: So they skipped the K6 for some reason?

Johnson: Well, they didn't have those names.

Krewell: These were all retroactive names.

Johnson: The K5 was the only thing that was named. We acquired NexGen, called it the K6, and called what we were working in Austin K7. So K6 and K7 were retroactive, effectively, or based on the fact that we already had a K5.

Krewell: So what was the K7 you were working on, what was the code name for that back then?

Johnson: Argon.

Krewell: Was it Argon?

Johnson: Or Krypton. It was either Krypton-- OK, Krypton was K5, and Argon was K7. And Jerry Sanders wanted to name the K5 Kryptonite, but Marvel Comics or whoever has the copyright, wouldn't let him use Kryptonite. So we had been calling it Taurus in Austin because all the reverse engineered products were named after cattle.

So Taurus was going to be the cosmic bull. So all of our documentation was named Taurus. And Jerry said, call it Kryptonite. So we changed it to Kryptonite. And then they wouldn't let us use Kryptonite, so we changed it to Krypton. And K7, because of noble gases, we used Argon.

Krewell: Did Jerry realize that calling it Kryptonite was you're taking out Batman? I mean you're taking out Superman. So that makes Intel the good guy. And I don't think he didn't quite catch that.

Johnson: Well, the good guy, but still Superman.

Krewell: Yeah. So when did you start on what was the K7, Argon? When did that start?

Johnson: As we were finishing K5, AMD insisted that I go on to the next generation. And K5 was transferred to another manager, which also kind of impacted K5, because you have the true believers who that was their baby, and then you have the people who are responsible for getting these other folks’ design to market.

So it's hard to make a transition like that from between tape-out and production, which is what we did. But it did free us up to-- now that we knew that frequency was a target, we were in a position to completely respin the architecture. And the K7 architecture’s not anything like either the K6 or the K5.

Krewell: So how much of the K-- actually I should step back for a second. How much of the 29000 team wound up on the K5?

Johnson: Oh, I would say maybe 25% or so.

Krewell: Now, once you pulled a lot of the team out of the 29K program, after the 29030 and then the derivative of 035, the superscalar part I think was Jaguar?

Johnson: It was called Jaguar. It never really went much of anywhere.

Krewell: Yeah, it never got developed. I remember arguing with Jean Anne Booth one time because it always seemed to be three years out. And it was like every time we talked to her, err another three years out. But did that sort of pretty much kill the future on the 29K program because you pulled a fair amount of the core team out to go on the K5 and then K7. It seemed like it took a lot of the wind out of the sails of the 29K.

Johnson: Right. But it was really the only decision that AMD could have made because x86 parts paid for fabs, and the 29K wouldn't. And vice versa, if you don't have the fabs paid for by x86, the design team isn't the big issue. The big issue is where are you going to build the things?

Krewell: Yeah, but it was the era when Jerry was still very much into having and owning your own fab. If it had been a fabless semiconductor manufacturer, it might have been different. But that's--

Johnson: Yeah, it might have been.

Krewell: So did you follow the 29K as it evolved, or--

Johnson: I was still involved. I was heavily on K7 at the time. And was still heavily involved in the product planning and basically architecture and design of the 29K parts.

Krewell: OK, and then who was on the K5 team?

Johnson: Oh man.

Krewell: Well, who do you remember off the top of your head?

Johnson: Well, the names, sort of the prominent ones, are Dave Christie, Mike Goddard, Scott White, Dave Witt. And I'm leaving out lots of people just because it was actually a fairly sizable team in the end.

Krewell: OK.

Johnson: And quite a few of them came from the x86 side of things, and I didn't quite have all the background with them.

Krewell: OK. And then, so as the K7 team developed, who of the K5 team went off to do the early K7?

Johnson: Let's see. This is kind of fuzzy in my mind because initially it was Dave Witt and myself almost exclusively. I think Dave Christie. The others that I mentioned largely stayed on K5 to help finish that up. So we were doing the initial microarchitecture and the concepts and starting to ramp up the RTL team. And AMD went through this other-- once the K7 officially launched, we had this massive hiring program.

Krewell: Once the K5 launched, you mean?

Johnson: No, once we--

Krewell: Launched the K7 program.

Johnson: Once we decided to do K7, we recruited just massive numbers of new engineers, which is also kind of-- I really wanted to use the K5 team because you develop a track record with them. They went off and were working on K5 issues. And so we hired a new team to do K7. And right in the middle of that, NexGen gets dropped in our laps. And let's say it was a chaotic descent, to say the least.

Krewell: Yeah, that's got to be very disruptive.

Johnson: Yeah.

Krewell: So you said you found out about the K6 in the newspaper.

Johnson: Yeah.

Krewell: And how did that impact your work on the K7?

Johnson: Well, they had made Atiq Raza, who had been the CEO of NexGen the CTO of AMD. And he really wanted me to establish or form an organization that would, let's say, reflect Intel's Architecture Lab. So AMD didn't have an architecture lab. Intel did. He thought we ought to structure ourselves like Intel.

Krewell: So this was Atiq, not Vin Dham?

Johnson: This was mostly Atiq.

Krewell: Because Atiq was CTO. And then Vin Dham was more business. He ran the business unit.

Johnson: Right. Yeah, the issue with Vin is I had hired somebody from Digital Equipment, Dirk Meyer, to, as we're ramping up K7, he was one of the senior technical people that we hired at the time into my group. And my interaction with Vin, let's say, was to make sure that Dirk Meyer took my place on K7. Because you really needed somebody with that level of insight to what we were doing.

It can't just be let's say any architect. It has to be somebody who really understands how these moving parts fit together. And so there was a period of time where I was on K7. Dirk worked for me. I was in some sense promoted. I was still a director, and Dirk was promoted to a director working for me. And then I was promoted to a VP under Atiq and started the Advanced Architecture Lab.

Krewell: Right. But did that pull you away from the sort of day-to-day K7 program?

Johnson: Oh yeah, but I had no issue trusting Dirk to do that, which is why I didn't fight it so bad. I didn't want to have a researchy type of role, but I really didn't want to see K7 not work out effectively because it had the wrong people in the leadership roles.

Krewell: Yeah, and I think the proof was that Dirk did develop a pretty decent product there.

Johnson: Oh yeah.

Krewell: The K7 was really a sweet product. But then the K6 was rolling out, and so did you have much interaction with the K6 team at that point, or was your groups there separate, your research group?

Johnson: Well, they consulted with us on architectural issues with K6. It was a friendly relationship, but we didn't really have-- there wasn't a lot of influence either way, us on them or them on what we were doing. Because we ended up in the Advanced Architecture Lab, yes, we did some processor research. More of it was platform-based; starting to look at the question of 64-bit x86 architectures.

But other than that, it was collecting groups within AMD who had worked in other product areas, particularly networking and communications. So from a professional standpoint, it was good for me because I got exposed to technology that I had early on in my education-- I had had experience with DSP, and communications, and so forth. But I had been working on processors for at that time maybe 20 years.

And so I got to go back into the wired and wireless communication, and telecommunications, and DSL, and home phone line networking, and just lots of interesting technologies, some of which did pretty well for a period of time. But nothing that really resulted in a new product line, let's say.

Krewell: So did you feel that was somewhat disappointing that the stuff you worked on in the labs never quite made it into an AMD product?

Johnson: Sure because some of the things that we developed, well, one of them was a two-chip 802.11b chipset with CMOS RF. And it was something that nobody in the world had. But at the time, AMD was distracted with x86. But we put together a-- I was able to attract marketing people. And we had really great engineers. Most of the actual RF work was done in Dresden, Germany.

So we started a fab. AMD started a fab in Dresden. And part of the agreement with the government there was that we would establish a research group, which I was responsible for. And they were really, really, really good, methodical RF design type of engineers. So we literally got about two weeks away from shipping that wireless LAN product. We actually had the boxes, and all the packaging, and all the boards, and everything.

And AMD decided just-- even though at the time Intel was making a heavy push for their mobile platforms. You see the Centrino kiosks in all the airports. That technology was nowhere near as quote, "technically good" as what we were doing. But AMD just didn't have the wherewithal to-- Intel was spending hundreds of millions of dollars on that marketing program.

And we had great products but nowhere near the marketing input. So that sort of continued for two or three years to the point where I just said, I got to go do something besides x86 processors because that's all that AMD is going to be doing.

Krewell: Yeah, that's a shame. They actually did have some great early wireless technology in the early stages but couldn't really commercialize. At that point in time, the market was largely verticals, and then it was just starting to go broader in terms of-- and then like the Centrino program, which kind of made it more pervasive to have wireless and them Wi-Fi as the brand came out.

Johnson: Yeah, when I joined AMD, they had lots of product lines. RAM, programmable logic, and x86, and bit slice, and networking, and communications products. You know SLIC, SLAC. I'm sure you know those basically line card products. There was a lot going on. It was really interesting. And around the time that I left, everything else had been burned off. And what was left was just this one product line, which is where they made all the money.

And they couldn't really afford the attention, because their [business] cycles are always going up and down at the whims of Intel. They really couldn't sustain the three years at a minimum that you need to establish any kind of new products. And to me, just doing that for the rest of my career would have been really boring. I'm sure could have done it from a practical standpoint, but I didn't want to do.

Krewell: And then so eventually you decided to leave AMD.

Johnson: Effectively, yeah.

Krewell: When you effectively-- the research group, did that sort of wind down in terms of size?

Johnson: Well, the research group-- let's see, this was late I think 2004. I may be getting the time frame wrong. It became another target of, we need to cut people. So we had a fairly massive layoff, and my group was hit pretty hard. And it was one of these things where you developed a relationship with lots of people, and you just have to have these massive cuts. And at its peak, the group was 300, and it was cut back to about 90 I think.

And then two weeks later, I was told to cut all but about five. I mean literally two weeks. And to me, you don't do that. You don't do the kind of thing where you send a message that it's over and then come back and do it again. And I think within-- if memory serves. I may be mixing up the timing a bit-- I think within six months I was gone.

Krewell: Did you just quit? What did--

Johnson: I quit and did nothing for a while.

Krewell: Oh really?

Johnson: Yeah. And I had a lot of ex-AMD people that I knew at Texas Instruments. So at the time I just wanted to go do hands-on engineering because I'm an engineer effectively. So I got an emeritus title at TI and a job where I could get introductions to groups and go around and get exposed to all the DSP products and the wireless work that they were doing. And that was a lot of fun for about five years, and then OMAP started to get into trouble.

Well, Nokia and TI both in some sense missed this transition from let's say a baseband-microcontroller/DSP that happened to do applications to a handheld computer that just happened to have a wireless connection. And even though at some level these are all processing platforms, from the standpoint of just the mindset of people, the mobile phone chipset and let's say the iPhone type of design are just culturally, they couldn't be more different.

And it's really hard to explain to either side of that cultural divide. You can't explain what the other's mindsets are. And I'd been exposed to both, and that sort of didn't matter. If you sympathize with both,

you're not the friend on either side of the equation. So TI and Nokia essentially started their decline because they just didn't make that transition from modem platform to computing platform.

Krewell: Yeah, and when they tried to make the transition, they came late to the party.

Johnson: Yeah, my perception is they weren't that focused on what you have to do to make that transition.

Krewell: Yeah. And it turned out to be a very competitive marketplace because you have a tremendous amount of vendors who jumped into it. And then of course you have the big player, Qualcomm, which wound up purchasing AMD graphics chip after the AMD ATI merger. They took the mobile graphics core became their Adreno.

Johnson: Right, and you have Google, and Microsoft, and Apple, which are computer companies, building phones. But they're not phones. They have phone functions and wireless connections, but they're computers. And it's too much detail, but the underlying platform design is completely different between the two.

Krewell: Yeah, it's a different mentality. And so you spent some time at TI doing some work there, and then TI sort of ramped down their modem platform. What did you do next?

Johnson: What I'm doing now.

Krewell: OK, so can you at least give us a little inspiration, a little bit of an idea behind the inspiration of your new company or what inspired you to do this project?

Johnson: Well, one of the--

Krewell: Without revealing anything.

Johnson: At TI, I got to work on lots of different things for lots of different products. And largely OMAP 4 is where we had the most impact. And TI was kind of interesting to me because they had really great engineering teams, really, really good people. But they tended to be siloed. And I never really quite figured out why.

They knew each other personally, but I got into this role of, well, you're doing this, and you're doing that, and you're doing that. Why don't you guys put this stuff together, and we'll help you. So we took things that were already existing and put them together in unique ways. And quite a few of the subsystems in OMAP 4, nobody talks about it because they're embedded in the imaging and that kind of thing.

There was a significant impact on the architecture of the SOC platform. Because the application processor’s largely what's licensed from ARM. And so where you can do mostly innovation is in the media subsystems. And it's my involvement in the medium subsystems that got me doing what I'm doing now.

Krewell: OK, yeah, that's where a big differentiator is. Because you can do a custom ARM design, and in fact AMD just announced they are doing an internal custom ARM design, and Qualcomm has, and Apple has. But it's a pretty well bounded problem. But in media processing, it's a fairly unbounded problem, a lot of stuff you can do.

Johnson: Right, particularly in image processing and in vision types of applications, there's this huge gap between the algorithms. Because the researchers can develop algorithms and prove their quality in however long it takes to prove the quality. That's very different than running these very, very complex applications in real time.

And my opinion now is the only thing interesting that I can see from my perspective in computer architecture is this big gap between those types of applications and what current computer architectures are capable of doing. And that's a really, really hard problem, and it's really interesting. So it's kept me off the streets for three years now.

Krewell: Well, it is an interesting segment. There's many different ways of approaches. Even the FPGA vendors are going with a reconfigurable logic. The graphics vendors, the GPU vendors, are going in after it with GPU compute. The DSP guys are going after it with DSP processing. Everybody's trying to crack that nut right now. It's a very interesting problem. Do you think that's one of the more interesting problems right now in computer architecture?

Johnson: It's the only one that interests me. And I'm familiar with all these approaches, and there's really a gap between what they can do and what's required.

Krewell: And then in general, what do you think about the state of the industry today? Because you have a sort of an outsider view of it now that you're not inside a company anymore. You've got your own outside consulting and looking at the big picture. What do you see the-- this is obviously one of the big challenges. But do you think that everything else is kind of rote these days? It's not as exciting to you.

Johnson: In some sense. Computer architecture's been static for a long time. And yeah, multi-core is the world's oldest new idea. Or some version of let's put hundreds or thousands of processors on the chip, and somehow they will achieve a level of performance. And there are fundamental limitations there that can't-- well, people will argue with this-- can't be overcome the way that these things are architected.

Krewell: So think the Amdahl's law problem, is that what you're referring to?

Johnson: Basically. You can't get away, the way these platforms are architected, with, let's say I have two cores. This one wants to read something, and this one's writing it. So the reader says, have you written yet? Have you written yet? Have you written yet? Have you written yet?

And the writer says, yes, I've written. But wait a while until it shows up where you can actually read it. And this one says, OK, but now don't write again until I'm done reading. And are you done writing yet? Yes I am. So there's no way to have that just happen like that. Well, there is, but that's what I'm working on.

Krewell: So let's reflect back on your years of the 801, 29K, K5, and early K7. What do you remember most fondly about those projects?

Johnson: I liked working in the embedded market a heck of a lot more than I liked working in the PC market. Because in the embedded market, you get to solve problems that people really have. And in the computing market, at some level, everybody's doing computing. They're all doing the same thing. They're all bashing each other all the time.

And those computers already exist, and you can make them faster and add the next generation of USB or whatever. But in the embedded market, you can actually find cases where people have problems, and they don't even really know how to solve it because their expertise has to do with whatever the embedded application is. They're not computer experts.

So when you can find opportunities like that where somebody has a problem and you can solve it with computer architecture, to me, that's a lot more interesting than building the next PC platform product.

Krewell: So it's solving real problems and with an engineering approach. So this gets back to being an engineer and approaching an engineering problem.

Johnson: Yeah. I mean, it's using architecture and other techniques to solve a need somebody has.

Krewell: So it keeps you still excited about being an engineer and solving problems.

Johnson: Oh yeah.

Krewell: Anything else looking back on the old programs? Obviously you said-- was anything specifically about the 29K program and K5 program that you sort of reminisce about? Or was it just the design team? Did you see any of the design team members much at all down in Austin?

Johnson: Quite a few of them, at least the ones that I interacted with most. And we have the occasional reunion.

Krewell: There is an open-source version of 29K. Would it be interesting if anybody followed up on that? Or you think the architecture is pretty much dead. There's nothing much interesting to do with it.

Johnson: There are a lot of things like it, I mean a lot of things. If you really dig under the covers, of what companies like TI were doing, and they had special spins of processors all over the place. So at this point, there's nothing unique about the 29K compared to quite a few other things.

Krewell: And then any other kind of an aha moment you had along your career that sticks out in your mind? Branch target cache was one of those. It was sort of a combination of two of you.

Johnson: Oh, I don't know. I think the funniest one is, we kind of touched briefly on me being at Stanford. I went to classes for like two years and got to the point where I was really burned out, as you might imagine, because I was traveling all the time.

And Stanford graduate studies isn't the easiest thing in the world. And I definitely wasn't the smartest person among my classmates. And I was in a completely different situation. They were students. I had a family, and a house, and mortgage, that kind of thing.

Krewell: And a job too.

Johnson: And a job. One summer, I came really close to quitting. So I had been taking classes every semester-- or I think they're on a quarter system-- every quarter, and including the summer. I went through that for two years, and during the summer, I was on the verge of quitting. So I didn't take any classes and kind of just took the summer off.

And I knew that if I went back, I'd have to pick a research topic. And I had no clue what I was going to pick as a research topic. So I had driven into AMD, and I got out of the car once. And I still remember, it

hit me like a bolt of lightning. It's like, why don't you look at multiple instruction execution per clock? I go, yeah, well that's an interesting problem, and it's kind of silly too.

And of course the first thing that-- I mentioned it to a few people, and everybody I mentioned it to laughed. They thought I was joking. It's like, it can't be done. But I said, look, I'm here. I'm going to Stanford. I'm working with a lot of smart people. It seems like an interesting problem to me. So I would say that's the one aha moment that became-- again, I was told by several people that it’s silly, crazy, couldn't be done.

Krewell: Wow.

Johnson: Because it was a complete shift from the way people had-- or if you could do it, it would be too complex and so forth and so on. But now you pick up any PC, and that's the fundamental implementation under the cover.

Krewell: Yeah, it's hard to go back and look at that and say, superscalar design. It's so ubiquitous and so obvious today. But it was still very difficult engineering problem or considered as such back then.

Johnson: Right.

Krewell: And that was still in the in-order superscalar, right? Your book, did you cover out-order?

Johnson: Oh yeah. That was actually--

Krewell: Oh, the more complex part, yeah.

Johnson: That was actually the main-- well, there were effectively in-order superscalar concepts at the time. But for the most part, they were integer and floating point paired instructions, right? So it's somewhat obvious.

Krewell: That was the 860 was paired like that.

Johnson: Right. So the interesting problem is, let's say it's any set of instructions up to a certain number, not necessarily two and not necessarily integer/floating-point type of partition. So that was the, gee, I wonder if there's a solution to that problem that led to all the other things that I did in that area. But it literally was just, oh, that'd be cool. And I've got to have some topic.

Krewell: Yeah. Well you didn't pick a super easy one, so it was still a fairly complex problem to solve. But it made it interesting.

Johnson: Yeah.

Krewell: So have you counted how many patents you hold? Do you have a--

Johnson: I lost count. I lost count at 80 something.

Krewell: You should update that for your resume.

Johnson: Well, I wanted to lowball it. What he's referring to is my resume says, over 50 patents. And I'm sure it's well over 50. But I don't know how many it is, and I don't want to lie.

Krewell: So we're kind of wrapping up here. But how would you summarize your career? How would you describe how you got into this crazy business?

Johnson: Just being really excited about learning new things and being bored if they're not new. And that's really what it comes down to.

Krewell: So you have a low threshold of boredom is basically what you're saying.

Johnson: Yeah. I've told a lot of people I'm edge triggered.

Krewell: So we're here at the Computer History Museum, and I know you haven't had a chance to look through it yet, but what do you think of the importance of capturing the history of the computer business and the microprocessor business?

Johnson: Well, I think it's pretty important because the interesting thing about computers is they've created this infrastructure that's basically very ephemeral. You can read paper documents that go back hundreds of years, and they're still paper. You can still access them.

You go back 10 years, and there's all this information stored on computers that's just totally inaccessible. And I think it's getting even worse. It's getting even worse as standards change and so forth. I just went through the process of upgrading from XP to Windows 8.1.

Krewell: Oh, that's a big job.

Johnson: And just trying to preserve the information you had in XP into Windows 8.1, it's a painful path. And that's in theory a compatible platform. So it's only getting worse as things get more complicated and all the media change and that kind of thing. And so it's a little bit ironic, but it's easy for all this information to get lost unless you're really trying to preserve it.

Krewell: Yeah, we don't want to lose sight of some of the history, and how it was made, and the people who made it. That's kind of what this program is all about. Well Mike, I'd like to thank you very much for spending time with us. And are we off?

Videographer: Can be.

Krewell: Oh, OK. And thank you so much for spending the time with us here at the Computer History Museum. I'd like to thank you and hope to follow on sometime in the future with a 29K program and a K5 panel as well.

Johnson: Sure, thanks.

END OF INTERVIEW

