Oral History of David Resnick

Interviewed by:
Dag Spicer

Recorded August 9, 2018
Mountain View, CA

CHM Reference number: X8751.2019

Spicer: Yeah. There we go, don't have to worry about it now. Okay, so today is August 8, 2018, and we are here with...

Resnick: Dave Resnick.

Spicer: ...Dave Resnick. Thank you for joining us today and Dave, you're going to talk about a bit about your career and most of your career, I think, was spent in high-performance computing.

Resnick: All of it.

Spicer: Is that fair to say? All of it, okay.

Spicer: Great. Well, why don't we start at the beginning of your career? Maybe tell us a little bit about, say, where you went to school...

Resnick: Sure, happy to do that.

Spicer: ...that kind of early training.

Resnick: Basically, I decided to become an electrical engineer, and this was in about 1960 that I went to the University of Arizona first. Did not do particularly well, went into the Army because I was about to get put on probation because of grades and I signed up for electrical maintenance, and ended up getting really good training from the Army doing, understanding both tube and some of the first transistor systems. I actually ended up working on the first transistor implementations, which were point-contact transistors, which is a helluva lot of water under the bridge since that, but in any case...

Spicer: ...just to take a little diversion. Point-contact transistors ... can you tell us just a bit about their characteristics...

Resnick: Oh, certainly.

Spicer: ...their reliability, that kind of thing?

Resnick: Well, the ones we worked on were called 2N110s and, basically, point-contact transistors were done before deposition technology was developed and so a transistor was basically a doped piece of germanium, I believe, with two sharpened pieces of metal that formed the emitter and collector while the germanium was the base. I forget exactly what metal that was. The stress of the metal going into the crystal created the inversion layer needed for transistor functionality. (The metal donated some of its electrons to the base, and this reversed the local doping.) The two metals, placed very close together, made a transistor. The gotcha is that the varying atomic distances between the points and some things like that meant that the things like gains were not very ….

Spicer: Predictable.

Resnick: ...not very predictable and consistent, but principally, one of the things that was a problem was that the gate current was high. In other words, the transistor gain was not very high. But in any case, they went into production and they actually became a part of the thing that I was trained for in the Army which was called the Fire Unit Integration facility, FUIF (pronounced 'foof.') ...

There was a central point of control for the Army called "Missile Master" that got radar information from the Air Force and which was responsible for doing all the radar stuff in the United States, in Alaska, and other places. But if the Russians sent over their bombers, then that radar information came in from the Air Force and was made available to the Army's Missile Masters and was made available to a Missile Master, if a particular city was a possible destination of the Soviet bombing run. Each Missile Master was associated with a large city that was anticipated could be a target for the Soviet bombers ...

Spicer: Sorry. What is a Missile [Master] ... is that a building or a thing or...?

Resnick: No, Missile Master was the name of a building that was put up in order to be the center of control for Army management of the group of Viking batteries put to protect a particular city. okay? In other words, it had some local radar capability, but it was basically a place where you had multiple screens where the Army personnel, and the Army officer in charge, would assign a particular group of incoming bombers to a particular screen and the officer at that console and who would then assign a particular air-to-air missile group to send ground-to-air missiles against the incoming bombers. And so the Fire Unit Integration Facility was the entity on the Viking missile side that received those commands and information on planes which became targets presented on the radar screen of the officer actually releasing the missiles and firing them within each local area. And so I was responsible for doing the maintenance of that interface between the Missile Master and the actual place where the missiles were, and as I said, part of that facility was [vacuum] tubes, but part of it was also logic done with point-contact transistors.

Spicer: Now, is Missile Master the Army version of SAGE?

Resnick: Missile Master took its direction from the Air Force SAGE and then did the local control. In other words, Each SAGE installation had computers which, I believe, still are the physically the largest computers ever built. Each computer was a whole floor. Those computers ran the displays and did things like target analysis. The Army got the remote radar information from SAGE, and as a result, the Missile Master was associated with [large cities like] Los Angeles, Seattle, New York or Chicago. I don't remember where they all were, but there were not many. They were large population areas that would be obvious targets and so I was responsible for for residing in a Missile Master in LA, and when we got a trouble call, I was to go fix it. And so that's what I did in the Army.

The result of that was I ended up knowing a fair amount of electronics, in particular repair. I then decided I really wanted to go back to university and graduate. The reason was that at the end of my Army career, I got hired, by the company responsible for maintenance work that was more intense than the local Army people were trained to do, of Missile Master, and because of the skills I had developed, ended up seeing some of the SAGE maintenance people from IBM who came to work in suits and white jackets and their job required them to have graduated, and so that became my goal. I went back in then as a late sophomore, early junior, to the University of Arizona and at that time the University of Arizona bought a CDC 6400 from Control Data. Given that I was interested in computers, I immediately went down to the bookstore and it turned out that they had all the publications from Control Data that described the computer and computer system in great detail. And at the time, the CDC 6400 was just coming on to the market and the CDC 6600 had been shipping for several months.

Spicer: What year would this be?

Resnick: ’69.

Spicer: Okay. Thank you.

Resnick: The vast majority of the stuff I was able to get, the documentation, was about the CDC 6600, which was the first machine that became successful for Seymour Cray. I was just blown away because there were features in the machine that were new and obviously better than any other machine that had ever been done, whether it was by IBM (which was becoming the big honcho in computers), or from what became UNIVAC, Burroughs, et cetera, and some of the others. There were a whole bunch of smaller companies.

Spicer: Can you expand a bit on that? Is it the architecture that you liked or...?

Resnick: I think architecture is the whole of it ... things like multiple accumulators. I think IBM had some limited, but Cray did it much better. In particular, he had three different kinds of registers. There were A registers, B registers, and X registers and the A registers were for addresses and B registers were for indexes and the X registers were for floating-point, integers, and any other values; that was unique. He was the first one to actually make fully parallel execution units and so you could issue, in effect, in parallel, and you did not have to keep track of how things were going to come back. So they kept things straight so that both parallel and serial program streams were fairly automatically managed if there was a register conflict, the reservation logic kept that straight, but the result was that Cray had found a way to make parallelism real, whereas pretty much all the other computers were like single accumulators ... so you did something, you waited, you did something else, and you waited. Whereas Cray was set up to ... you could do issue, issue, issue, issue without being directly concerned about execution times and You could have all the functional units busy, and so that was the first attempt, successful attempt at real parallelism. (Yes, you still might well be concerned about execution functions that had relatively long latencies. One good thing about the 7600 was that the small-core memory had a cycle time that was one-tenth of the base clock rate/base instruction issue rate. The result, along with the built-in parallelism, was that cache memory was not really needed (and hadn’t been invented at the time ☺) .

The other thing that was really, really, good, being Seymour, is that the instruction set was simpler than virtually every other instruction set at the time. In other words, there were some really simple things, like from Digital Equipment Corporation, which was successful at the low end. But Cray's was an attempt to be a fully capable science and engineering system and so just the number of things that were brand new with that machine, as I said, blew me away. [Aside: Was a RISC/Reduced Instruction Set Computer before the concept was reinvented and popularized.]

At the time, the CDC 6600s were selling very well and Control Data consequently went out and did an awful lot of interviewing, trying to get people to come in as programmers and customer engineers. When they announced they were coming, I was the first guy on the list at the University of Arizona. Yeah, and consequently I was offered a job and it was to write diagnostic software and I said, “But I don’t know the machine. How in the hell am I going to do a good job?” And they said, “Well, what do you think?” I said, “Well, could I be a customer engineer first and train, then therefore learn the insides in detail and then go into customer engineering?” So that was the basis under which I was hired and I never went back into software. The result was that I got trained on the CDC 7600 and we’ll do the war story that’s part of that, which is where I worked with Cray for just a little bit and basically was training on Serials 2, 3, and 4. Went with to the CDC 7600 Serial [Number] 3 that was delivered to Los Alamos. I think Serials 1 and 2 went to...

Spicer: Livermore maybe?

Resnick: ...Livermore. Thank you. And so the ... I was there. I was in New Mexico then for a year and at the time Control Data was developing the STAR-100 and they had a promise for when they would have a delivery and there’s a whole set of war stories around why that never happened.

Spicer: Yeah, I’d love to hear those, if you feel like it.

Resnick: The gotcha is that there is a whole bunch of people that are still alive, I think, I have to be careful to not insult...

Spicer: I understand.

Resnick: ...whether or not they’re worth insulting.

Spicer: Yeah, I understand. Yeah.

Resnick: Okay. We’ll talk about it later. In any case, the ... my manager at Control Data said, “Dave, Los Alamos has signed up to buy a STAR-100,” and the system that they were going to buy would either be delivered to Los Alamos or it would be delivered to General Motors, depending on how things worked out, and he said, “Well, what we’re going to do is we’re going to take two guys who are currently on site at Los Alamos and send them back for STAR-100 training, and the General Motors people,” the Cray people who supported General Motors, “are going to send two people back with the agreement then that whoever gets the machine gets those guys back and the other two guys move from wherever they were at when they went back for training to the system,” And so those four people would be initial on-site maintenance people. The result of that agreement, however, was different in that the STAR-100 was in a helluva lot of trouble.

Spicer: Technical problems or management problems?

Resnick: Management problems, by far. Well first of all, the STAR100 was a tremendously complex machine. It ended up having one of the most complex set of instructions of any machine ever invented, and I've never seen one that comes close. First of all, it was a vector machine which was, I believe, the second attempt at vectors, but the attempt that succeeded was the Cray-1, but that was a machine that Cray hadn't invented yet. Ended up with the STAR-100 competing with the Cray-1 and losing big time for multiple reasons to do with some of the choices that were made for the STAR-100, and it turned into something that I've done much later over the last few years [which] was to find a way to go back and fix the architecture problems. So I think I know a new way to do vector systems that pretty much will outperform any vector system in the world. The gotcha is I've never found anybody I can talk to, to convince. But in any case, the result was that I ended up being on the design team for the STAR-100 as it went through a long and very difficult development process. Well, how I can say it? The machine, the STAR-100, was supposed to ship in 1970. We held the 1970 shipping party in 1974, which was...

Spicer: Not too good.

Resnick: Not really. Let me put it this way. The agreement for the STAR-100 was that they would deliver the Serial-1 system to California, to Livermore, with a delivery date, and if they delivered late they would pay $10,000 a month. They ended up owing Livermore more than several hundred thousand dollars at least. They did finally get a machine—Serial 1--but that just shows you some of the problems. The machine was four years late, and the development fix there was to put some design management, control and designs for portions of the machine that had not been complexly envisioned. The development program leadership was given to Neil Lincoln. You know Neil Lincoln from ETA?

Spicer: Yeah, yeah.

Resnick: Well, he was the guy that actually did the management control design and then put it ... had the vision on how to complete the machine. Neil Lincoln was one of the most intelligent people I ever met, first or second. And one of the things he did, for example, was when they had picked the instruction set for the STAR-100 but couldn't find a way to implement it, Neil told them how. He wrote a compiler for the STAR-100 and it was the only piece of code that was really fully vectorized and because the vector instruction set was so complete ... I mean for example, you could even do sparse arithmetic and there were no branch instructions in the code that he wrote, I didn't believe him but I looked and it was true. [An example of the capability was a Mask capability in the vector instructions. Could do a vector instruction to build a mask, and then use the mask to control the needed vector operation so that the vector operation would pass over data that did not fit the execution of the instruction.] The compiler was never really fully completed, but just the objective was the sort of thing that Neil Lincoln would attempt.

Spicer: Wow.

Resnick: So in any case, this is now probably late ’72 or something like that that Neil was put in charge, and the first thing he did was to stop everybody from working, which is because none of the individual designers were coordinating with the other designers and consequently things like the control unit wasn’t getting done. People were working on functional units, and Neil said, “Okay, here’s the plan. Here’s what we’re going to do. Here’s the block diagrams of what we’re going do,” and he made it happen. And so I became part of the design group and then helped to deliver the STAR-100, the first couple, and then remained as a designer with Control Data for quite a few years working for Neil and then working as part of the group that did, for example, some of the initial designs when Control Data picked up the first integrated circuits and made its own chips, which were gate arrays.

And so I did the logic design of the first IC prototype demonstration which was something like an adder, just as something to exercise the manufacturing tools and the test equipment and things like that, and did that for a long time and maybe that’s the end of that part of the story. During the time I had done, originally got the CDC 7600 training, I ended up being, as I said, on the team that was going to go to Livermore, to wherever the machine was delivered. Before I got on the STAR-100, I was on the CDC 7600 team, and so I ended up doing checkout for the CDC 7600s. The CDC 7600s were a whole bunch of modules with a plug back panel where Cray hired a local group to do some of the assembly work, to actually plug all the wires into the back panel. The gotcha was that there was no automatic way to check that all the wires were functional or plugged into the right place, and so the checkout was taking an initial set of modules, not all of which were good, and plug them into a new back panel and then debug the modules and debug the wiring, which is a wonderful, wonderful way to learn the machine because you’re in its low-level details. And so here I was...

Spicer: Sorry. Can I ask one question?

Resnick: Sure.

Spicer: The backplane that you’re talking about...

Resnick: Not a backplane.

Spicer: Oh, it’s not a backplane?

Resnick: It was a set of chassis in which modules were plugged in and so the back ... what became a backplane was just a whole set of plugs for the modules and so the back of that was places where individual wires were plugged and so you had a wire diagram. You go from this pin on this module to that pin on that module.

Spicer: See, that’s what I was wondering. Is it a wiring mat, such as...

Resnick: It is a wiring mat.

Spicer: ...you would see on a CDC 6600, for example, or the Cray-1...?

Resnick: The wiring mat for the CDC 6600 looked more or less like that of the CDC 7600, the difference was that they used different gauges of wires because they were designed for different impedances. Consequently the wires were thinner and easier to use but one of the results was that the pins were also more dense.

Spicer: Yeah. I have a quick question for you that is kind of an urban legend that I keep hearing which is ... and it kind of, it's plausible but the legend goes that Seymour found a computer one day that he ... customer couldn't get working and the fix was that one of the wire lengths was wrong.

Resnick: I was going to tell you that war story.

Spicer: And ... oh, please go ahead, then. I didn't even know it was true. I just heard it, you know, third-hand.

Resnick: The machine was not delivered to the customer. They knew who the customer was, and so this was one of the fairly early customers. The plan for Control Data was that for the CDC 7600, just like for the CDC 6600, they had a plant in Arden Hills that would be the manufacturing plant. But it took a while from when Seymour said, "Okay, here. This machine is done," to get the modules into production and have all the manufacturing set up so they produced the machine. And so the agreement was that the first 10 systems would come from Chippewa Falls, which then allowed the Arden Hills plant time to get into production and so this was a system that was intended for an aircraft company in Washington: Boeing.

It was intended for Boeing and the system was in checkout. Seymour typically didn't spend a lot of time at the shop. Seymour is an introvert and consequently pretty much worked by himself but he would typically come in at least once a day and look at the mail. A couple of times he came in at a specific time where an important customer was visiting and he was there to shake their hands but otherwise the area around Chippewa Falls was managed by Les Davis, who later on basically became head of Control Data when Seymour left. He was there until, we'll say, Control Data got in trouble for some reasons we haven't talked about yet.

But in any case, that war story is that several engineers were working on a problem in a CDC 7600 being checked out, and Seymour came in and saw them and continued with what he was doing at the time. About an hour later, he walked past again and they were still working on the problem. And so he poked his head in and said, "What's going on?" and they said, "Well, we've got this problem, and we can't figure it out." And so Seymour took the oscilloscope probe and had at it. In about 10 minutes he said, "Well, you've got a short path on this signal," The guys working on the problem saw they had all the signals but had not looked at the timing. They just didn't realize that one of them was early and consequently was getting there ahead of when it would be properly detected, and so Seymour picked up the wiring manual: "The wire is supposed to be this long," At the time all the wire lengths were color-coded. So in other words, if the wire was supposed to be 36 inches long it was supposed to be blue or whatever or pink or whatever, and so the guys ran back and looked at it. It was the wrong color, and so Seymour said, "Okay. Go cut us a new wire." Ten minutes passed. Nobody came back. Seymour knew what the problem was and he was also very impatient and so he thought about it, I was told, for a little bit and had an idea. He cut the wire, stripped off both ends, and clipped his scope probe between what were then the wire ends. The length of the scope probe added to the delay because now the wire was longer, turned the machine back on. It worked.

Spicer: Wow.

Resnick: That machine...

Spicer: That’s incredible.

Resnick: ...that wire, that scope probe was delivered to the customer.

Spicer: Really?

Resnick: I found out about it and heard that war story when ... now, a few of years later, they were shipping CDC 7600s, and that 6600 machine was brought back and installed within one of the memory ... excuse me ... one of the places where they were doing software development in Control Data Headquarters.

Spicer: Right, right.

Resnick: Okay? And so at the time, I was doing a little maintenance in the development center and got the direction: "Hey, Dave, go work on this problem," and I saw the scope probe, and I said, "What the hell?!" <laughter>

Spicer: So they just left it in there?

Resnick: They left it in there!

Spicer: Wow! That's incredible. <laughs>

Resnick: Okay. And so I looked at it and then ...

Spicer: That's so funny.

Resnick: Well, I mean, to me, a little bit of the magic was that the scope probe had a signal hook on it with a spring on one end , and multiple wire turns on the scope plug-in on the other end, so it hung onto the wire really well.

Spicer: Oh, yes! Right, right.

Resnick: But that looked like it was not going to come look loose either, so I left it.

Spicer: Wow! That's an amazing story!

Resnick: Yeah.

Spicer: Lovely! Thanks for telling us that.

Resnick: Sure. Other war story is I'm in training with basically the debug team for Serial 3, 7600. And as I said, Seymour would, at the time, was doing software development. And he developed the initial 6600 OS, that I think was actually the first OS that went with the early machines, before the software people at Control Data were able to put in their own version.

Spicer: Do you remember the name of that OS? Was it Chippewa?

Resnick: It was Chippewa ... COS, Chippewa Operating System.

Spicer: Right.

Resnick: That was Seymour's. And the software people came in and did Compass and all the other stuff to make a real product out of it. But in any case, Seymour was doing that for the CDC 7600 side like he had done for the CDC 6600 side. And so what he would do is he would write at home, he was an introvert, as I say,

Spicer: Oh, yes.

Resnick: And he would come in, typically in the evening, and he would sit at the operators console and run the code that he'd written that day. Don't know exactly how he was able to do it. It meant that he had to do things like punch cards and things like that to get the binary. But in any case, he would do any debugging, or whatever was needed to be done, and then he would leave! One day, he comes in, and I happen to be in the CDC 7600 room doing some maintenance on some of the tape units, and Seymour was acting irritated. And he would type into something, he would look at the screen, he would type in something else in fairly rapid succession, and so he obviously had a problem of some sort. And he did that, I'm not sure exactly, but it was something on the order of three-quarters of an hour, and he got up and left. And the next day, I'm in the room again, and Seymour comes in and looks around, and I'm the only engineer around, and he said, "Dave, would you run diagnostics on the machine? I think the machine is broken."

And so I basically ran the machine's diagnostics and all the diagnostics worked. He said, "Dave, this Boolean instruction is getting the wrong answer. Set it up so that we can debug it!" And so I did that. Basically, what you want to do is to put the machine into a loop, so that you've generated high enough scope trace frequency, so that things lit up on the oscilloscope and you'd typically place, you know, somewhere between two and ten instructions in a loop, which meant that you were doing a couple hundred thousand instructions per second, and you got a good bright display, and so I set it up to include the instruction that generated the Boolean operands, the Boolean instruction, and the following instruction that tested the Boolean result.

If the result was zero, do this; if the result was not zero, go over there. And what Seymour said was, "The Boolean instruction should be non-zero, and so we should take the non-zero branch but it's taking the zero branch." And so we set it up, and I knew where all the test points were, and I knew where all the logic was, having worked on the 7600s for most of the year at that point. The Boolean instruction worked properly. And between the two of us, we showed that the test instruction consisted of a functional unit of three modules, and each module did about 20 bits of the result, and you then combined the result as needed into a finaly result as needed by the executing instruction. Basically, if you're doing a zero test, then you wanted zero result from all three modules. And a non-zero result was any of the three being non-zero. And it turned out that one of the modules was stuck equal to zero. In other words, it didn't matter what the operands were, you always got an "equals zero" result from the failing module

[Aside: Floating Point, integer, and Boolean data was 60 bits. So three 20-bit modules covered. The system had no character or byte-level capability. The character format used 6 bits, so 10 characters per word. 8-bit bytes had not become a fully accepted standard at the time,]

Spicer: Right, wow.

Resnick: And so those middle bits were non-zero, then the instruction went in the wrong direction, and Seymour was happy. "Dave, I got a new module," plugged it in, the machine worked. The next day, Les Davis comes to me and says, "Dave, Seymour was impressed with you last night. He said you knew what you were doing, and you helped him. Would you like to upgrade the diagnostic so that that problem would be found [next time]?" And it was pretty obvious that the diagnostic should have found it.

Spicer: Yeah, right, right.

Resnick: And so ...

Spicer: Wow, what a vote of confidence! That's wonderful!

Resnick: Yeah, and at the time, Les Davis was second in command to Neil.

Spicer: To Seymour.

Resnick: To Seymour. And so at the time, there was no compiler. Everything was done in assembler. And it turned out that within Chippewa Falls, everybody used Cray's assembler. And so I grabbed Cray's assembler and I looked at it. It has the most odd structure of any assembler I've ever seen before or since. And I looked and said, "Do I want to learn this? I'll never use it again." And it would have taken me about as long to learn to use the assembler properly as I wanted to take to write the code. And so I ended up writing the code in octal. In other words, I knew the machine instructions so ...

Spicer: Oh, my gosh! <laughs>

Resnick: ... well, enough, so I could write out the instructions in octal.

Spicer: Wow! And you thought assembler was too hard? That's funny. <laughs>

Resnick: Well, as I said, the instruction set of the CDC 7600 was really simple and straightforward. And so at least it was easy for me. In any case, I wrote the code, punched the cards, and about a week later, I walked up to Les Davis and said, "Here, this is the diagnostic." And he said, "Dave, does it work?" I said, "Well, of course it works!" And he says, "Follow me." And he unlocks Seymour's office and the big office, huge desk, nothing on the desk except there's a module on the corner.

Spicer: <laughs>

Resnick: He hands me the module.

Spicer: Oh, that's the bad module, okay.

Resnick: It's the bad module, which he very carefully did not send in for repair.

Spicer: That's smart, yeah. So you could put it back in and see if your diagnostic ...

Resnick: I put it back into the machine. The diagnostic found it. The diagnostic shipped with the machine. And after about three or four months, a programmer in California wrote a reverse assembler, and reversed my code back into Compass. And that diagnostic shipped ... I think became standard. But more ... if I was doing it, I would have simply gone back and inserted it into the previous test sequence in the existing diagnostic that included the test cases. But in any case, I never saw it again. So.

One more war story.

Spicer: Okay.

Resnick: About Neil Lincoln and ETA. As far as I know -- it's been stated multiple times, it's appeared in print a couple times -- the name for ETA Systems when Neil was starting the company, came from "ETAOIN SHRDLU," which was known as the "printer's devil,' basically the frequency of the letters in the English language in order.

And so Neil's son said, "Let's name it after ETAOIN SHRDLU," because they couldn't find another name." Okay, so ETA stuck. The ETAOIN SHRDLU, okay, comes from the time of Linotype machines, which was how newspapers were composed. The person using the Linotype machine had a single row of keys, and he had to use the keys with his left hand. And what he would do is he would enter all ... he's reading a page, which is the column of type that he's putting together, and the Linotype would would fill a column, with the characters/letters that he would type in. At the end of a single line of type he would hit the, in effect, the return key, saying, "That's the end of this line," then the machine would generate a lead slug that held that line of type. The person running the machine had to pick up the hot slug as it came out, and put it in a receptacle, in order, and so the left hand was busy typing, and in order to control that, in order to reduce the amount of hand motion the letters in that line of type were ETAOIN, the space character, and SHRDLU, in that order.

Spicer: Right.

Resnick: Okay, now the reason that that became the name of the rascal, okay, for the machine, was that occasionally, the Linotype would stick. One of the characters would stick, and consequently, all the letters after that were not entered. And so the people who ran the machine had a trick in that the stick was typically a very easy to recover from if you knew what the key was, you could just jiggle the key and it would pop back up. The person doing the entry did not want to look at the entry keys, as that would mean taking their eyes off the data page being read from and thus lose time in recovering from scanning line of keys. And so what they would do is they would take their thumb, upside down or something on that order and press it along the line of keys. And what it meant was that often ETAOIN SHRDLU, that whole line was entered as a to-be-printed line of type. And it wasn't intended. So you would read a story, and all of a sudden, you'd see the line printed in the paper! The people putting all the columns ot a page together were supposed to scan each page as it was being put together, but occasionally missed the ‘extra’ inserted line.

Spicer: Oh, interesting!

Resnick: And people said, "Well, that's just ETAOIN SHRDLU putting his name in the paper." <laughter>

Spicer: That's great.

Resnick: So in any case, that was ...

Spicer: There's some AI software, too, called SHRDLU, by Terry Winograd. Do you know about that?

Resnick: No.

Spicer: OK. I've got a couple questions for you. On the Cray 1, I've heard that it took six months to build and six months to debug. Is that about right?

Resnick: I never worked on the Cray 1.

Spicer: Okay. How about the CDC 7600 ... construction versus debugging?

Resnick: Oh, construction?

Spicer: Yeah, or assembly versus debugging.

Resnick: The first CDC 7600 took a long time.

Spicer: Yeah, sure.

Resnick: Because as I said, you were putting really not very well checked out modules, along with a un-debugged back panel, and so it was certainly months for the initial one. After serial number three ... I think starting with Serial , they came up with a way to plug in to the delivered back panels, plugs to check that the wiring, in fact, was correct. And to fix that before it was ever powered up. And the result was that the manufacturing time was reduced tremendously. And part of the speedup was also that module testing at assembly was upgraded if/as needed.

And you know, there's another war story, which was that they ended up with some reliability problems in that machines would generate random failures sitting in the field, and that turned out to be a grounding problem in the modules. It took well over a year to find out what those problems were. And ended up with Cray having a whole bunch of people looking for debug and paying expensive people to go and troubleshoot it. And I was sitting in ... onsite in Los Alamos, and we had a special highly-trained engineer come in just to try to work on our problem ... on this problem. The 7600 would crash randomly; it was not very often, but the crash left no clues as to the cause. And so we had done things like shock testing, temperature testing. We had changed the temperature in the room to see if there's a thermal problem in the machine. And so late one morning, we are just sitting there, and we're exhausted. We have no further ideas. And at that time, the CDC 7600 machine room had hallways for people going back and forth just doing their job. And a person pushing a cart, where the cart has a whole bunch of heavy paper on it, and so she's walking down, and there's a sharp right turn in the hallway and one of the I/O pieces for the CDC 7600 is close to the corner, and she runs into the piece of I/O equipment, and the machine crashes. And she just caused our failure.

Spicer: Yeah. What was the I/O equipment?

Resnick: It was a tape controller. Which means it had direct cable connections into the mainframe.

And so that turned out ultimately to be a grounding problem. In other words, you generate a shock, and the grounding went out of bounds with respect to the power supplies on the CDC 7600 itself, and it crashed.

Spicer: Wow.

Resnick: And so that's the sort of things that we would work on.

Spicer: Those give you gray hairs, those experiences.

Resnick: Oh, yeah, we worked on that goddam thing for a long time, and it would just fail randomly, and it turned out that people would come in and touch something.

Spicer: Oh, were you able to correlate it to people bumping into this controller?

Resnick: Basically, we changed the grounding.

Spicer: Right.

Resnick: And Los Alamos also said, "Well, what it means is that the asphalt that we've got down for the floor is generating a charge."

Spicer: Really?

Resnick: Right.

Spicer: Wow!

Resnick: In other words, the wheels rolling on the floor ...

Spicer: Really?

Resnick: ... generated a sizable static charge, they were hard rubber. And consequently would build up a charge, and that became the charge that caused the transient.

Spicer: Oh, my goodness! <laughs> That's complicated!

Resnick: Yes, of course! <laughter> Excuse me, that's what I got paid for, at least then.

Spicer: Yeah. Oh, my goodness.

Resnick: I mean, ultimately, you know, I got into doing things like system level architecture. I mean, for example, at when I left Micron helping to do the hybrid memory (HMC), I said, "Okay, well, at this point, I know how to build a extremely capable high-performance memory system all the way up to, let's say, something with ten million DRAMS that users will never see fail."

Spicer: Wow.

Resnick: Now, I didn't say the parts don't fail. I'm saying, the ... for roughly the cost, okay, that is done today, with a modified Hamming code, okay, for that same cost, I can give you a reliable memory system that, for example, allows maintenance while the operating system and the user's code is undisturbed.

Spicer: That's amazing.

Resnick: Yeah, I never found anybody to listen to do that. The gotcha is that it has to change some of the architectures and I've never been able to get some of the people in Micron interested because it's different.

Spicer: And they're competitors to Rambus, would you say? Or not really?

Resnick: Rambus had technology.

Spicer: They might be interested in your ...

Resnick: No, Rambus doesn't do memory anymore.

Spicer: Oh, really?

Resnick: And at the time, how we say, they talked Intel into adopting their plans for memory. And so they were going to do the memory parts, and Intel was going to do the interface. They were late. Okay, they were also going to charge the industry for using those parts. And a charge a premium that was above the DRAM costs at the time.

Spicer: Like a royalty.

Resnick: A royalty. And consequently the whole memory industry was really, really pissed off.

Spicer: At Rambus.

Resnick: At Rambus and at Intel which was adopting the technology. But in any case, they finally started production, and actually delivered some systems. One of the things that Rambus had done was to allow the basic capability to be extended. And so what happened was the first time that they put together a system that used that extended capability, the extension was basically connecting multiple modules, using the Intel I/O. And it failed. And it was because Intel had never properly designed it, and didn't adequately check it out to see that, in fact, it worked to full specification. And as a result they lost a year. The result of the loss of that year was that everybody's plan for using Rambus was deliberately killed.

Spicer: Oh, dear.

Resnick: And they went back to using the standard DRAM, and consequently Rambus became ... just something that you laughed at. Which is sort of too bad because some of their engineers were really, really good, and I knew several of the people.

Spicer: Can you briefly say the difference between a CDC 6600 and CDC 7600? Is the CDC 7600 just a faster version... or is it different in any way.

Resnick: It's actually significantly different. The CDC 7600, first of all on an architecture point of view, Seymour realized that he couldn't put enough memory built out of the magnetic core DRAM available at the time that would meet his goal for size and speed.

Spicer: Right.

Resnick: Okay, and so what he did is he created another level of memory, which they called Large Core Memory (LCM). And that was made to be much, much denser, but simultaneously much slower.

But the small core memory was made so that you could things like fetch individual words. And the memory had lots of banks, so that you could be making multiple memory references in parallel with the instruction execution. In other words, he increased the level of parallelism in the CDC 7600 and I'll tell you how he did that in a little bit.

On the large core, you ended up having to transfer blocks ... I don't remember how long they were, but it was ... could have been something like 32 or 64 60-bit words. And it was up to the programmer, saying, "I'm running short of memory in small core," then it could transfer a block of memory back to the large-core memory, and transfer another LCM block back into the small core. And the idea was that hopefully you could do that while your code was running, and therefor overlap memory movements which made the system look like it had more memory.

The other thing that he did that was different is when he did the CDC 6600, the functional units were independent but not pipelined. And so in the CDC 7600, everything was pipelined. And so it meant that, for example, if you were doing a Boolean [operation], it took, I'm not sure exactly, but probably four or five clocks, you know? A clock to get it over there, a clock to get it back, etcetera, etcetera. And he pipelined them. So that you could be doing multiple instructions on the units in order, and consequently you became a lot less sensitive to the order of code. And the only thing that was really slow on that machine was the divide instruction because doing something like making a pipelined divide instruction was way too expensive.

Spicer: I had a question about the PPUs.

Resnick: Okay.

Spicer: Can you tell us a bit about those, and how they affected the way people programmed the machine?

Resnick: The PPUs were built very differently than on the CDC 6600. But otherwise, I think were really pretty instruction compatible. Same 12-bit data size and 4 Kbyte memory size. Single accumulator.

Spicer: Right. Sounds like a minicomputer.

Spicer: Hm?

Resnick: It's exactly a minicomputer! The gotcha was that in the CDC 6600 Cray did not have the logic space for multiple minis to do the IO and run the OS, but wanted that functionality to be part of the mainframe, and so he came up with an architecture that allowed him to get the performance of multiple minicomputers with one set of execution hardware. That was called the Barrel and Slot. Basically each PPU had its own core memory, and the core ran with, I believe, a ten-clock cycle. So from the time you gave it an address until the data it was back, if it was read, it was exactly ten clocks. So what he did was he had ten of those cores, and then had a single set of execution units, and he pipelined them. So that they executed every tenth clock, which was the performance they would have had if each processor had fully independent memory.

Spicer: Right.

Resnick: At that point, you had 10 functioning 10 PPUs, and they were all running full speed, with one set of hardware function units.

Spicer: Amazing.

Resnick: And that was called a barrel and slot. Think of the staves in a wooden barrel, with one of the staves being the execution function units. Each position held state for one of the executing programs and was responsible for interacting with one of the 10 PPU memories.

Spicer: Okay.

Resnick: And incidentally, I've tried to keep that idea alive, okay, and I've got something that's simpler than that, well, both simpler and less complicated, that allows that sort of parallelism to be put into some of today's machines and which could be another one of my architecture speedups.

PART II : Papers review

Spicer: Do you want to go through the papers maybe? Is that a good ... ?

Resnick: Then let's go through the papers.

Spicer: Yeah. That's a good idea. And we can, we'll just leave the recorder on while we go through these.

Resnick: There's a lot of paper here.

Resnick: Brochures. With ETA systems, one of the things I was still very proud of was to come up with test logic that allowed basically each chip to test itself and to run in parallel such that you could power up a brand new module and know it was good within ten minutes. Rather than having to run a whole bunch of diagnostics and do things like find and fix solder-joint failures or possible chips that failed in the assembly process.

Spicer: Did you work on both the liquid cooled and the air-cooled versions of the supercomputer? The Piper and ... ?

Resnick: I was the person responsible, okay, for getting Piper, the air-cooled version, designed. Basically, the plan was that some of the integrated circuits that went into the air-cooled version would otherwise have not gone into a liquid-cooled version. The machine was cooled to liquid nitrogen temperature. Some chips did not speed up enough, even at this extremely low temperature. And the idea, okay, those chips were way too expensive to just throw away, so come up with a machine that runs warm. Air cooled gives you a slower machine in which you can sell those other chips.

Spicer: And a much cheaper machine, too.

Resnick: Much cheaper machine.

Resnick: To me, that machine basically, excuse me, that paper, okay, is pretty much everything about how that system came to be designed, okay, done by Tony Vacca, okay, who was the project manager and Engineer In Charge who actually ran the mechanical development. He was in charge of the computer integration, right ... and the design of the chips, the physical design of the boards, getting the boards made and manufactured, getting the chassis built; all that stuff was Tony's. He worked with Neil Lincoln while Neil was in charge of the guys that did the logic design that went into the integrated circuits. And so between the two of them they made ETA work.

Tony is the head, okay, and then these, basically these were Doug Carlson did the mechanical design. Jim Kreilich did the, was mostly involved in the software, okay. Integrated circuit development was Randy Bach. Dave Frankel was mostly software type.

One day Tony Vacca, okay, came to me and said, "Electronic Magazine is here Dave, and I want you to tell them about the built-in self-test logic." And so I stood up and gave them a pitch and their mouths hung open a little bit, okay. Because this was ... this is unique, when you can build a board and the board says, "I'm fine, or I'm broken here," okay.

Spicer: That's amazing.

Resnick: Okay, that's ...

Spicer: Yeah.

Resnick: And so that was the ...

Spicer: Wow. Could you interrogate the boards in ...

Resnick: Oh, yes.

Spicer: Like, in the software and do, like, do a check on each board as well?

Resnick: Okay. And what it meant, for example, is if you had something that would go wrong, okay, you could run the self-test diagnostics on the floor, and say, "Well replace this board because the chip at location XY is bad,".

Spicer: Oh, really?

Resnick: Oh, yeah.

Spicer: You get down to the chip level. Amazing.

Resnick: In some cases, they would get down to the pin.

Spicer: Wow.

Resnick: Let's put it this way.

Spicer: That's incredible.

Spicer: Great. I had a question about the cooling system ... Whose idea was it to use liquid nitrogen and were there other cooling systems they considered?

Resnick: No, not at all.

Spicer: Like Freon or something?

Resnick: Well, this was basically Tony Vacca and Neil Lincoln putting their heads together. The highest perform circuit technology at the time, okay, was ECL. This is way before CMOS ...

Resnick:, TTL hadn't even been done at the time.

Spicer: Okay.

Resnick: And so, how could we come up and make a really faster machine, but that was also against it with respect to power limits.

Spicer: Right.

Resnick: Cooling was a big issue because ECL runs hot, and so that became a whole set of issues by itself.

Spicer: Yes.

Resnick: And Tony said, "Well," and just probably initially, as a joke, okay, "You know, CMOS, even though it's a hell of a lot slower than ECL, if you make it REALLY cold, it speeds up as it goes cold, okay."

Spicer: Right.

Resnick: "And if we actually ran this as liquid nitrogen, okay, then it would speed up by enough to be really, really interesting. And oh, by the way, liquid nitrogen is cheap enough so that we can buy a quart of it for as much as a gallon of milk."

Spicer: Wow.

Resnick: Okay. And so, Neil says, "Well, our density would go up by a tremendous factor," okay, because being a CMOS, even though it was not to the point of doing VLSI chip design; we were at the probably 2 or 3 micron level at the time. And so, nevertheless it was still much, much denser than ECL. And so Neil said, "We're interested... can it work?". And so their big issue, at least for Tony up front, was, "Can I make a board that doesn't turn into a potato chip when it goes cold?"

Spicer: Right. <laughs> Right.

Resnick: And so it turned out that Control Data was not using FR4 [a printed circuit board substrate material based on epoxy and fiberglass] although FR4 is still today the most common manufacturing material for boards,. The board material used had better dielectric properties so that lines could be longer and support needed impedances and support more wiring lines per inch. And so, it was the boards were not done in FR4, okay, and the metallization technology, okay, was also very different, okay. And so they took one of their boards and they put it in liquid nitrogen and it worked.

Spicer: So this is what I don't understand. Is liquid nitrogen not conductive? Like you can literally dip it into a liquid and ... ?

Resnick: There's no loose electrons and it doesn’t dissolve any board or wiring materials In other words, it's an insulator, [ph?] okay. Totally immersed.

Spicer: So they're immersed, just like a Cray-2 or whatever.

Resnick: Well, a little bit of a war story. I was working on a piece of equipment that had an ETA liquid nitrogen machine that was only partially done. It was basically in debug mode. The cryostat did not have the system in it, but was full of liquid nitrogen.

Spicer: Okay.

Resnick: An open dewar was basically a big no-no. And so, one of the support people came in carrying, okay, an assembled CPU module to put in the cryostat. He assumed it was empty. So you're carrying this and his face is right against the assembled CPU and memory system , okay?

Spicer: Oh, my gosh.

Resnick: And your arms are right here, okay?

Spicer: Oh.

Resnick: And so your visibility is <laughs> extremely limited.

Spicer: That's a huge board, isn't it?

Resnick: It's a fairly sized ... He walks in, okay, and he knows roughly where the cryostat is. And he starts to lower the thing into the cryostat and the bottom of the modules,, touch the liquid. Instant sizzle. Panic. He drops it and it falls into the dewar full of LN. And so the rules at the time, were that when we want to take an assembled system that we've just put together, we cool the system down very slowly so then you could introduce the liquid nitrogen without thermal shock.

Anyway**,** so he dropped the machine and that was the first time, that they had ever really shocked the system, you know, in the space of a couple of milliseconds the system had gone from air temperature to liquid nitrogen temperature. And the amount of failures that then found as a result? Zero. Everything worked.

Spicer: Wow. Pretty robust ...

Resnick: Very robust. And then so for the Piper, which is the air-cooled version, I led the group and basically the idea was what differences, are there going to have to be in the system in order to run it in air? And it came down to one large issue and that is that the memory system itself, , was very carefully separated from the cryostat ... if you cooled the DRAM chips, that changes the sense level enough so that you don't have functioning memory any more.

Spicer: It changes the logic thresholds.

Resnick: Exactly, it changes the logic thresholds.

Spicer: I know Seymour Cray once said, or, no, it was Gordon Bell, that observing Seymour Cray and saying, you know, supercomputing, and Gordon Bell has designed supercomputers too, is about plumbing and packaging as he called it.

Resnick: Well, did you understand that the cooling issue was why he ended up leaving Control Data?

Spicer: No.

Resnick: He'd shipped the CDC 7600. He had done all the planning for the CDC 8600. He was going to grow the module size to enable more logic per board, and to finally get into a good level of integrated circuit because the CDC 7600 still used individual transistors.

Spicer: Right, in cordwood modules.

Resnick: What he wanted to do, was to make a bunch of larger modules which would look very similar to a larger version of the CDC 7600 modules. And, on the CDC 7600, basically you had a cold plate filled with Freon or coolant both above and below. And so the distances that the metal had to stay cold was only a couple of inches.

Resnick: The modules were much, much bigger for the CDC 8600. Because the cooling flow was at the edge of the modules the physical distance from the cooling flow to the logic being cooled gave a result was that the circuits, the integrated circuits or transistors that he was going to use, never achieved the cooling level that was expected. The result was that they built modules, powered it into the first prototype thing that they could power up and test, and things got hot enough that the transistors fell off the modules. Solder melted and some of the transistors fell to the bottom of the cabinet.

Spicer: Wow.

Resnick: The 8600 used a still fancier version of I think individual transistors. So I don't know that he had gone into integrated circuits then on the CDC 8600. I don't think he did. But in any case, he worked for a year to try to fix that problem.

Spicer: Right.

Resnick: And finally said, and incidentally, you should check this with Bill and probably even Les Davis.

Spicer: Okay.

Resnick: Okay. Because what I'm about to say, I've been told, I don't know it's a fact. He went to Control Data Headquarters and said, "This is not going to work. I'm going to start over again." And Control Data said, "Hey, we've already sold two systems. You can't quit." I mean, you can't start over again. And he said, "I'm out of here." I mean, that's cutting it short, of course,? But, but that (CDC 8600 cooling problem] was the reason that Cray left Control Data.

Spicer: Yes. Right. He wanted to start on something new. And I've seen a picture of the 8600, at least a prototype.

Resnick: Yeah.

Spicer: And it's like a Cray-1. It's circular.

Resnick: That's right.

Spicer: It's just lower profile, but.

Resnick: Yeah, that's right.

Spicer: You know, for the same reasons, I presume, keeping wire lengths short.

Resnick: Oh, very much.

Spicer: Right.

Resnick: Oh, yeah. Yeah, he had to keep the wire lengths short in order to get them out of traffic that he wanted it to be in a reasonable space.

Spicer: I'd like to thank you for chatting with us today about your experiences, Dave. It is much appreciated.

END OF THE INTERVIEW