Oral History of David Patterson

Interviewed by:
John Mashey

Recorded: September 13, 2007
Mountain View, California

CHM Reference number: X4150.2008

John Mashey: I'm John Mashey, a trustee at the Computer History Museum and I have the great pleasure to interview Dave Patterson today. Let's get on to it with no further adieu. Dave tell us about your early background and how on earth you got into computing.

David Patterson: I was the first member in my family to actually graduate from college. My dad was trying to go to college, trying to graduate. I beat him to it. At our high school, we had, you know, what would be, I guess today, called the AP math program which I was in and my high school teacher, we got calculus at the time, and he could teach anything. He always wanted to be an actuary and just thought that was just the grandest goal that he could think. So, I assume all of us thought the same thing, so as far as I know going to college I was going to be an actuary. I didn't quite know what it was, but he wanted to be that. So, I was a math major and then a course was canceled at UCLA [University of California, Los Angeles] in my junior year. I couldn't get in, so I had to fill a program. Took a class in computers. I'm sure I knew what computers were. They were popular but I have absolutely no interest anytime before I took that course, that I would be possibly interested in computers.

Mashey: So, when was that?

Patterson: That'd be 1967. So, I think it was one of those, we were in the quarter system. I had to look at my transcript, and my guess it was like the winter quarter, and took a Beginning Fortran class with the punch cards and all that stuff that people don't understand these days. And just, boy, this was it. Math had gotten more abstracted in the junior and senior years, and from it had a more practical way that they taught calculus at the time. And, wow, it was like the stuff Fred Brooks writes about. It's, you know, your thought, the magic of your thought. There was no computer science major so I just took as many computer classes as I could thereafter in computer science, which they'd recently formed, and in the business school. Just any place I could take a class, and I just thought this was the greatest thing ever. The next part of the story is Jean Lubert [???] who was a PhD student at UCLA at the time had just graduated, and he hadn't yet started his academic job. So, he hung around and taught a couple quarters.

So, I was supporting myself in downtown Los Angeles working in a factory while a student. My wife and I had recently gotten married right before my junior year, so I was paying the bills by working in a factory. And I took a compiler course from Jean Lubert [???] and I just kind of came to class and said, "Boy, I really love this stuff." I guess this was in my senior year, and I really loved this stuff, and wished I could do that instead of working in the factory. So, to my everlasting thankfulness he decided he would go around as kind of a senior grad student, go talk to a bunch of research groups to see if he could find a place for this undergraduate student who I guess did well in his class. He got me a job. I remember the story, it was like the Internet stuff was happening there and I talked to, I think, Steve Crocker and he was interested but [Leonard] Kleinrock was out of town. And so he tentatively said we could hire him, and Kleinrock came back, "And what are you doing making offers while I'm out of town?"

But that one went away but there was another guy there, Larry McNamee, he was doing CAD [Computer Aided Design]. Way back then it was the competitor to SPICE [Simulation Program with Integrated Circuit Emphasis] and he wanted to hire somebody, so I got on there and so I got to work, get paid as an undergrad student, paid as a faculty member and this was in senior year, and I thought at the time the job market wasn't all that good, and never had thought about graduate school. A Master's Degree seemed like a pretty cost efficient thing to do. I thought I would stick around for a Master's and then explained to

my wife, and then we had just about the same week that I graduated, got my Bachelor's Degree from UCLA. Our first son was born and so talked to my wife about whether it was okay to go to graduate school given we had a family. She was happy in UCLA's family housing. So, I thought I'd get a Master's Degree and then kind of everybody in my office was going on to PhD. They assumed I'd get a PhD so kind of talked to my wife, “I'm enjoying this, what do you think about me going onto a PhD?” She said, "Well, if you're smart enough to do it, go ahead." And so I’m an accidental PhD student, and surely, you know, for the kindness of Jean Lubert. If he hadn't taken the initiative to try and get this undergraduate student involved in research while an undergraduate, there's no way I would be where I am today. I don't think I would've gone to graduate school if he hadn't done that because I had no role models and no plans. Heck, for me getting a bachelor's degree would've been a big deal in my family. So, it was surprising that I ended up getting a PhD.

Mashey: So, then how did you end up at Berkeley [University of California, Berkeley]? You clearly by now having very long roots in Berkeley. Your family's there. You've clearly found the place.

Patterson: I come from a family of storytellers. So, just strap yourselves in. I decide it's time, you know, I've been there a long time, I have a second child, you know, supporting a family. Our younger siblings are starting to buy houses. It's getting time to graduate and the project I did, actually, was wrapping up, so started to interview. Alas at that time, or perhaps with my advisor, I didn't get any real advice how to interview, how the academic system worked. I was kind of learning it on my own. I applied at all the schools in the Pacific Eight Football Conference was one thing that I did. There along that time, I've gone to conferences. I was doing the microprogramming stuff, which was a precursor, in retrospect, of the languages and compliers for microprogramming. And so there was a SIG [Special Interest Group] Micro workshop and I remember going to one of those, and one of the guys at one of the schools in Virginia kind of latched on to me and said, I think it was Virginia Polytechnic or some school like that. We were out to dinner and I was there, you know, kind of on my own and a guy, it was John Wakerly, John Wakerly came by and said, "I really liked your talk. Be sure, you're going to graduate soon?" "Yeah, I plan to." "Be sure to apply at Stanford." And the guy I was with dinner with after the guy from Stanford left, possibly, to try and get me interested in Polytechnic says, "Stanford's not for people like you and I." That was kind of a reasonable assessment at the time, kind of, but somehow that went back in my brain.

I eventually did, after many years, I did complete my research project. It was microprogramming and I actually built a complier and I did a little head-to-head competition with a master's student’s where he wrote in assembly language. I wrote in high-level language. It compiled and I was able to optimize at a higher level. We both built emulators for the HP-2116 [Hewlett Packard] minicomputer and worked them in code, and, you know, a cool little thing. It just took a long time to get done. I started applying on my own and one of my friends was a professor at Maryland, and I came and gave my academic talk and it lasted about 20 minutes. That didn't go well at Maryland [Laughter]. He came and took me aside and said, "Dave, to do an academic talk you really have to do a longer talk." Okay, so I applied at the time, to Colorado because of John Denver, Colorado was kind of this mythic place to go live. And so I applied to Colorado State and the University of Colorado, and my wife came with me on the trip. And we were at Colorado State on Friday, and the University of Colorado on Monday. Colorado State on Friday it was, you know, Fort Collins is really a cow town. It's really not the Colorado of John Denver's imagination. We were walking up and down the streets and people in pickup trucks were yelling at us and stuff. Ah, it was so different. At the time the guy liked me and made me an offer on the spot, but it was going to expire. Well, I want to explore my options, so we had the weekend and were hiking around the hills of Colorado. My wife, Linda, who was born and raised in Albany, California, moved down and we met in high school

loved Northern California. And we were walking around, and hiking, and she goes, "You know, I just can't help but think we're going to end up at Berkeley. I just can't help but think that." And so I explained to her here's the academic hierarchy, UCLA is lower down than Berkeley and this is not likely. "Honey, you just don't understand how it works." Okay, so she's got this in her mind and we start getting some offers, not other offers and she says, "What about Berkeley?" Well, they haven't called me. This is too awkward. I have an offer from Bell Labs. It's kind of going slowly. [She says] “Call Berkeley!”

Mashey: When was this now?

Patterson: This is 1976. This is the spring of 1976. I'm going to finish soon, so I'm applying for jobs. Late in the interview season my wife insists that I call Berkeley, and I call up the chair of the department, Elwyn Berlekamp. So, Elwyn Berlekamp who's, you know, extraordinarily famous mathematician, computer science, and games and stuff like that, but he's a mumbler. After years of talking to him I have no problem communicating. If you're not used to his pattern and style, his patois, you have a hard time listening to him and over the phone, it's really hard. So, I'm trying to talk to him, "Hi, this is Dave Patterson. I've got offers from Bell Labs." Turns out Elwyn was at Bell Labs. He had gone out to Bell Labs, so that turned out to be a good thing to say. Berkeley's the only place I haven't heard from that would make a difference to me and I just thought of it so late. Basically I didn't tell them because my wife's been bugging me to do this, so I'm doing this extraordinarily humiliating experience just for my wife. He tells me, "Well, you know, you're top ten, not in the top five." I thought, that wasn't that bad. Now that I know the story, Elwyn says that to anybody who called, "You're the top ten but not the top five," it's a friendly thing to say. He decides to pull my resume out. At the time, Berkeley was trying to decide to grow in their hardware. They had hired Al Despain. They give Al Despain my resume. He looks through it. Looks pretty interesting. He's going to go down to San Diego because he's adjacent as it turns out, drive through UCLA. So, he stops by and we have a conversation, hit it off. He goes back, so I get invited for an interview in June or something: really late in the interview process. Give a good presentation. People like it and I get the job offer, and the story is: I get the phone call, "We're going to make you the offer." I get in the bus for married housing to come home to celebrate with my wife. I had called my wife, "Take our two sons, we're going to Berkeley." We're jumping around because she just went there, and I get off the bus and I start walking towards the house, because I remember the day, and I kind of jump up in the air and kind of high-five a tree, and I knew I was good enough for that job. [Laughter] So, somewhere in the back of my mind I was really pissed off that that guy said that you and I aren't good enough for jobs at places like Stanford. I didn't know it. "Yeah!"

Mashey: If I hear all this right, it sounds like we owe a whole lot of this to an accident of a missing course and the fact that your wife goaded you.

Patterson: Oh yes, my wife plays an extraordinarily important role in my life. Because then we get the job offer. Well, at the time, the academic salaries… I think I still have my offer letter, but it's about $12,000 a year. And both of our younger sisters have houses. I say, "Linda, I've stayed in graduate school all this time and we've sacrificed economically for all this time. I perfectly understand if you would like me to get a job at a real company so we can get a house." She's asking the question, "If you turn this job down and go to industry, can you change your mind and go to Berkeley?" "Ah, no, very unlikely." "If you go to Berkeley and you don't like it and decide to go to industry, can you do that?" "Oh yeah, that's pretty easy to do." She says, "Okay, we'll be poor but proud." [Laughter] Absolutely, my wife of now 40 years plays

crucial roles at many points in my career, and had she made less unselfish decisions at many points, I'd be someplace else.

Mashey: In the one sense, it's a little sad. Of course, I was at Bell Labs right at that period, so we might have seen you over there, but that would have been New Jersey then.

Patterson: Well, we'd grown up in California. Whether we would have accepted a job in New Jersey, I just don't know. What was the other career path like? I don't know.

Mashey: But it's worked out really well.

Patterson: I was happy how it worked.

Mashey: Now that you're at Berkeley, talk about what sort of research you started doing and how you got into that and so forth.

Patterson: [I] show up at Berkeley in January of 1977. At that time there wasn't a lot of programs to help steer people the right way. I didn't even know where you ate lunch and there was nobody around. So, I wandered down University Avenue a mile from campus to a pizza parlor. Within a couple weeks, I got to get involved in a research project. Well, everybody wants me to do a parallelism, so I'll do a parallelism and that was kind of, my thought processes. And then I was looking for a mentor and Al Despain, who was an associate professor at the time played that role. We started talking about how we would do a multiprocessor project, around the era of the transputer, and we thought grand thoughts. Al had worked with some of the famous physicists because of his adjacent thing and they had this kind of model of, you know, thinking great thoughts in the afternoon. So, we had this very interesting architecture philosophy about how we would design things. We were going to design everything, a microprocessor, a programming language, and operating system, just soup to nuts: everything. And then Carlo Sequin who came from Bell Labs who was doing CC stuff at the time, and so we were just going to, these grand vision papers. We actually had no resources at all. We didn't have any funding. We didn't have any computers. We were going to do all this stuff kind of using the shared department PDP-11. I was kind of working with this group of people and that's what we did. And so we wrote papers, and technical reports, and stuff like that. Around that time, Sam Fuller called who used to be a professor at Carnegie Mellon. In fact, in that story after I got the job offer at Berkeley, he calls up to say, "Gee, with your microprogramming stuff, that would tie into some of the stuff they're doing in CMU [Carnegie Mellon University]", and would I be interested in a post-doc kind of position, I think, at CMU, or maybe a research associate position at CMU. But I had an offer at Berkeley. Well, he goes back to DEC [Digital Equipment Corporation] and follows up and says, "Gee, I like this microprogramming research. We're doing these VAXes with these giant microcodes, and bugs, and all kinds of problems there. How'd you like to come visit and transfer the technology?" And so I said I would. So, this was the fall of 1979 from,, wherever that is, September to December. We would spend it in Boxborough [Massachusetts] at one of the VAX minicomputer sites. They needed computer scientists and I was going to do this microprogramming stuff. So, they asked me to give a talk and the theme of this research project at Berkeley was XTREE and that it was going to be a tree topology to connect all these chips together. And the guy kind of introduced it, "Well, Patterson's going to talk about XTREE, whatever the hell that is," basically. So, I think, maybe not everybody knows what this is, but the break in my academic schedule made me stop and have time to

stop and think about how to do things: think about things I hadn't thought about. Well, what are the advantages and disadvantages of doing something in academia? What can you do there? What can't you? What resources do you need? How much resources do you have? And so it gave me a break to stop and think. Well, I guess I could see what industry was doing firsthand. I had worked at Hughes Aircraft while I was a graduate student, but this was a real computer company not an aerospace company. So, you kind of see what they could do versus what we could do, and pros and cons, but somehow it was a transformative experience and I came out of that being much more careful about how you pick a project, what the goals you set for a project in terms of in the environment that you're in, as opposed to abstractly.

Mashey: So, I'm curious then. You've done a long series of projects that have actually had relevance out in industry. Is this particular experience what caused that to happen? Because I've known a number of academics who've done a lot of interesting research that seems disconnected from the real world.

Patterson: I guess, because I didn't get a lot of guidance…,told what to do… I wasn't part of a big research project. I was self-funded because I was either working at Hughes Aircraft in a fellowship. I kind of had to figure stuff out on my own. I think as part of that… well the benefit of not having a lot of support as a graduate student is you have to kind of think from first principles. I suspect Hennessy was like this too but he had a thing from first principles and to me, because we were in architecture and there are people selling microprocessors, this was like the ground truth right? That I figured out that this is really great. It's not like philosophy or something where were argue endlessly. There's these ideas and then you have a place to test these ideas. So, it'll be important to continue to interact with industry to both test your ideas when you think you have a good idea. Plus, obviously they'll help you figure out what the problems are. They will help you with this difficult problem in academia to find out, is this an important problem or not? This is an intellectually interesting problem for sure. There's lots of those. How important are they if you solve them, right? And I always have used the crutch of interacting with industry to help figure out whether this problem's important or not.

Mashey: So, anyway then you came back to Berkeley…

Patterson: Yeah, and I came from Berkeley, these VAXes, these were giant microcodes, and these were just chock full of bugs. So, I simultaneously came from the environment, the last project, where clearly the microprocessor is the future of computers. It was like, what's the question here? This was very different at DEC, by the way. DEC at that time, microprocessors were toys. Microprocessors go on the Apple II's, you can't do anything with them. And so you need these really big computers and that just didn't make sense to me. So, I came back and I could see the real world experience in building production microcode and just how chock full… errors were there all the time when they were doing change orders. So, I wrote a paper almost immediately after coming back or maybe even before I came back, talked about for the future of VLSI [Very Large Scale Integration], which was obviously the future of the microprocessors, we were going to have to use microcode and we were going to have to have mechanisms that could patch… repair the bugs that were in the microcode. We had to do that. So I wrote this paper and sent it to IEEE Computer [Institute of Electrical and Electronic Engineers], and it was rejected. The bottom line was: This is a stupid way to design computers. It doesn't make sense to design microprocessors this way and, with this extra RAM, and cost, and patching and the field. This is nuts. So, I was confronted with two facts. You can't build a VAX without lots of bugs in the microcode and this is a stupid way to design microprocessors. These are both true statements, and so my

experience with DEC, my experience with lets try and do things that within the resources you have in the university, and the rejection by IEEE of this idea, led to the RISC [Reduced Instruction Set Computer] project. So that absolutely led to the RISC project and we just started off, I kind of came back with fire in the belly and I think that very first quarter that I was back in January we did the advanced graduate course. We investigated the RISC ideas. We did a series of papers and the students in that class were investigating those ideas. I think that came right out of that experience. I think kind of sometime in there John Cocke shows up, and I'm pretty sure it's after he did the class. The class got started and then he shows up. Or maybe it was before I took off on my leave, but I'm not really sure when he comes back, but that's why we called it RISC. This was this simplified instruction set to make it facilitate, and if it takes you ten years to develop it, it doesn't matter. You know, all that stuff was kind of, I think, pre John Cocke, and I don't think John particularly thought of that. I don't think he was that excited about the idea. He was excited about compilers and matching the instruction set, and then radical ways to build... Why do we have to worry about worst-case clock delays? Stuff like that. In ECL [Executive Control Language], he had this different mindset and I had this keep it simple to track the fast changing Moore's Law, and compilers don't use most of that stuff. And I think that was out of, in reaction to the VAX. But John clearly came through in that time and, he spent a day with us and talked to us, and was inspiring to talk to. And I think, the next day he went to visit [John] Hennessy and Gibson. I think John who is a little bit, started with Stanford a little bit later, and because of his background, I think more embraced the compiler perspective on the stuff, where ours was the thing there. So, I think that's when we start this series of courses, part of this: how do you get things done at a university? Well, I recognized that courses have deadlines, really serious deadlines where you give grades and if you don't make them, you know, students fail and stuff.

Mashey: Just like the real world where we have to shift things.

Patterson: You wouldn't think you have in academia, but you do. You have these deadlines that people will get grades, and if they don't make them, they get bad grades, and they care about grades. So, let's tap into that culture. So, we did that investigation course where we investigated the architecture ideas and simulated, and, the register windows ideas are in there, a whole bunch of other stuff… delayed branches. There's a whole bunch of things. You can read the individual papers where we collected that stuff. This is right when the VLSI Mead & Conway stuff was coming along, so the spring course was a take the Mead & Conway course and learn that, and then I think if I've got this right…

Mashey: So, this was spring 1980 now, is that right?

Patterson: Yes, and then the follow on course was build the microprocessor, and so the follow on course was the quarter, I guess, in the fall was, that's where we built RISC 1 really in the course, and then two other guys built RISC 2. They're called blue and gold, RISC Blue and RISC Gold, and we later called them RISC 1 and RISC 2. Two other guys, Manolis Katevenis and Bob Sherburne built, without the constraints of the course and without the deadlines of the course, built a much more elegant implementation of the ideas, and a much better implementation of the ideas. The other guys did it just as part of the coursework. Manolis and Bob did it kind of as part of a dissertation, stuff like that. But, I think the series of courses, so I think from coming back and starting the first course in January 1980, I think we had chips back and stuff a couple of years later. You know, you look back in retrospect and it was kind of amazing how at the time. You talk to modern guys, "Oh, it's only 40,000 transistors. I mean, how hard

was that?" [Laughter] And for us, this was the first hardware we'd ever built. Will it work? We were fighting against the titans of the world.

Mashey: Describe the environment a little bit in terms of what did you have for computer support? What did you have for CAD tools? What was the process of actually taping out and getting something back? I mean, when I look back at those times, it seems like there were periods when universities are actually close enough to the technology to do some pretty interesting leading edge things, and there's other times where the hill to climb is pretty hard. That seemed like a particularly good period for universities.

Patterson: I agree. I think the classical CAD industry of the time did absolutely not have this Mead/Conway approach that we could make optimizations at a high level. If you understood enough you could make optimizations at a high level that were much more important than getting the transistors right at the low level. Because of Carver Mead, and [Lynn] Conway's evangelism, we got the attention of smart computer scientists. Just about this time, a couple years after I arrived, John Ousterhout who was kind of a superstar graduate student at Carnegie Mellon, actually in operating systems, came to Berkeley and decided before he got here that he wanted to do VLSI CAD. That he's going to switch from operating systems to VLSI CAD just because he thought that would be an interesting thing to work on. I think one of Berkeley's rise from the masses to become a leading university, one of those major events was hiring Ousterhout. That kind of got everybody's attention because he was the number one draft choice at PARC [Palo Alto Research Center] and everywhere, and he went to Berkeley. It's like, wow, why'd that happen? So, I latched onto John and he was my next door neighbor in our building, the office next door, and I wanted to make him happy. Fortunately, he's just gifted in terms of his skill set, his taste, and how he attacks problems, and builds code that actually works. So, he built his first system called Caesar, which was a layout system and he was willing to fix it. So, we used it and then Rich Newton led the more traditional CAD efforts, and of course SPICE and things like that, and because there was a bunch of computer scientists doing their projects, people would fill in and build tools when they didn't exist. But we had this very synergistic relationship of somebody who was trying to build a ship with somebody who wanted to build CAD tools that work. And because we were taking this high-level perspective on the right way to design chips, that gave opportunities to do new types of CAD that industry absolutely wouldn't have done it that way. But it made sense from our perspective. So, I think those early years in the 1980s, and the other thing that I didn't mention is Bob Broderson was trying to get Berkeley back into DARPA [Defense Advanced Research Projects Agency] funding. Berkeley had had some missteps in DARPA funding before I'd gotten there which involved getting a lot of DARPA money and computers catching fire [Laughter]. So, I don't know, maybe that's the way you end a project. You know, God, I can't get the bugs out of this thing, what if it burned up? But supposedly somebody went away for the weekend and it burned up over the weekend. So, we had some recently not so good experiences and he wanted to get us back in DARPA, and it was right after I came back from, as I remember it now, just after I came back, maybe the week I came back from VAXes. And he said, "Look, I'm going to try and get some DARPA funding. What's your ideas?" And then I started off, like, in this old style of, “Well, we should do parallel processors, that's the right problem to work on. What do you want to do?” “Oh, what I want to do, I want to design a computer.” “Why don't you propose that?” So, I think the fact that we had a funding path, that Bob was willing so shoulder some of the work but give us an opportunity to pay students to do the work, which was a novel idea, and to buy computers. We used VAXes at the time, Berkeley UNIX and VAXes. So, if you look back at the RISC papers, the three RISC groups, the Cocke IBM ones, John at Stanford, and us, you know, the question is, “Well what language are you interested in? What's the operating system?” For us it was C, UNIX. What's the question? What else you going to do? And so, at IBM it was PL.8 and it was this strange ECL [Embedded Common Lisp] thing that didn't have, average clock timing, and stuff like that. We're microprocessors and the language is C, and the

operating system is UNIX. Why are you working on something else? And I think Hennessy may have had the same insight because I picked that one. DARPA might not have funded him, so he did Pascal or something. That was our competition was the VAX. The VAX was the machine to use. John, I think, did Pascal and the PDP-10 or something, and I assumed that was some grantsmanship thing. But, you know, when look back, you know, yeah, well obviously, UNIX, C, you know, doing better than the VAX. And I was interesting at the time too, it wasn't like the Intel 886 was competition. It was the VAX because the way computers were heading, you know, ideas for minicomputers were going to chips. Obviously, the chips were going to dominate. What we wanted to do was figure out how to build the future generation chips and that was just, you know, the environment we were in at Berkeley made that crystal clear and, you know, in retrospect it was the right decision. Did I answer that question?

Mashey: Yes, I think so. Actually, that may be an interesting thing. Your comment about C and UNIX there was that it seemed at that time that that was about the time people really started thinking about designing architecture for which C would actually be good, as opposed to putting C onto weird things that it was never really intended for C.

Patterson: Yeah, well there's this prejudice against C, this low level language and, you know, kind of pedestrian language that, you know, intelligent programmers wouldn't use, that Pascal was this other thing. And, you know, came out of this Ken Thompson, Dennis Richie school of, Ousterhout described it as kind of like a plaid work shirt. You know, it's the thing you put on. You go out in the yard. It's not going to protect you from anything, but if you know what you're doing, you get your work done. And there were these kind of language wars, well I don't know about wars, but language arguments or battle going on at the time, high level, low-level stuff. So, it was controversial to pick…

Mashey: To pick C at that point.

Patterson: For some people, right. For us, well, there was…, no there was…

Mashey: There was the Portable C Compiler.

Patterson: Portable C Compiler, called PCC. Portable C, we didn't have people who wanted to work on compilers in the project, the portable C compiler was there. We could generate code. We could run it, in simulators, all that stuff.

Mashey: So actually, talk about that a little bit more. Talk about the relationship between Berkeley and Bell Labs at that point, because I'm thinking of people that went one way or the other, or Chris Thompson was out visiting and Dave Ditzel ended up over there and so forth....

Patterson: Yes, we just had an anniversary party for Ingres, which had, I suppose, I don't know, it's fortieth, some terrible number, some really big number. And I was kind of asking, Ken Thompson came there and I thought it was Ken Thompson visiting there and then Bob Favery [???] did it. I think if you talk to Stonebreaker, the database guys had already decided that UNIX was clearly the right way to go, and that it wasn't Thompson's visit so much, it was that. Maybe that led to Thompson coming, but, I think their

view was UNIX was clearly the right thing to happen. So, the database guys were on board with UNIX. Thompson had visited and was apparently, Thompson was one of our alumni. So there was these ties between the Bell Labs there, and then in those first few years they were doing the RISC stuff. Well, Dave Ditzel, we wrote a paper about what was called high-level language computer architecture at the time, about basically saying, rather than closing the semantic gap by building these very complex constructions and architectures, you could do something more sensible than that. And he went and joined Bell Labs. His first job was at Bell Labs designing microprocessors. We ended up writing the paper together. We had written this paper for the architecture conference, and that summer I could kind of see how things were going and it looked like things were really going to work. So, I wrote this draft of the paper, the case for reducing instruction set computers, and I sent him a draft, and he gave lots of comments back. “And well, Dave, should we author this together?” I mean they were really great comments. “Yes, sure.” But I kind of consciously, in a cold, calculating way, I kind of wanted to get the argument out there, and actually we had enough results that I thought, “Wow, this could work.” And if we can get the argument where people start talking about it going when our results come out, that'll help us. I thought it would help with the technology transfer somehow. So, what I did was, I wrote this paper and then Dave came back, and a lot of good comments so we made it a co-author. So, then I sent it to the friends I'd worked with at DEC. For some reason my nature is to get lots of feedback on everything I do, maybe because I wasn't a very good writer at the time. I gave the paper out and I sent it to Doug Clark who I'd worked with there, and I had met Bill Strecker, and he wrote back, "Oh, what are you thinking of doing this?" "We're going to put it in this computer architecture newsmagazine, no referring at all." And he says, "Well, I think we're going to write a rebuttal." So, sounds good. I kind of liked the open exchange of ideas, and so we wrote our paper. They wrote their rebuttal. That went into Computer Architecture News, and so you had this kind of very unusual he said/she said technical conversation between, who knows who this guy is from Berkeley, but boy, these are the guys who did the VAX. You know, this is the architect of the VAX, Bill Strecker, and Doug Clark, a well-respected guy. And,this back and forth that was going on. So, I thought, “Oh, controversy!, a fight!”, and so that a got a lot of people's attention to talk about these issues. All the technology changes and everything that was going on at the time. So, people started talking about these ideas. There was the 801 project was secretive. There is something called the 801, “What is it?” It was very hard to know even what they were doing. It was supposed to be very exciting, very breakthrough, but unless John Cocke came and told you what they were doing, you couldn't figure out anything about it. So, there was that kind of the mystery of the 801. There was this kind of straightforward competition with these papers around these ideas that what was going on, and the thing at the time was people were angry. We wrote this paper that the way to design computers, it was kind of more or less this quantitative approach and the way things have been going on is pretty stupid. People were yelling at each other. They were emotionally upset at the time, that what you're saying is dangerous ideas. You should stop that!. This is bad for the field that you're talking this way. It was really, in retrospect, not like… well here's two conflicting views and people were very upset. So, that led to a series of debates at conferences with these ideas came up where it would pack the room. We'd get both sides there. It would be filled up. It would go on for hours and, we did that kind of three times and every time people would show up because, I don't know, there's a fight, there's a controversy. So they were very volatile times. Let me squeeze one more thing in, and so who were the guys doing this? These were two assistant professors, not tenured faculty, who were making powerful companies very mad at them, maybe me more than John because of my personality. I think they got madder, John says they didn't get so mad at him. Now, he's president of the university. I'm still a professor. They got really, really mad at me.

Mashey: Well, actually you might be amused. There were actually echoes of that [at] other places, because inside Bell Labs, Ditzel was promoting these things. Okay, and of course there was a fairly large, well funded, Syskey chip effort inside there. I actually, at one point had to review Ditzel's proposal

to do this and bless that, before I left. But I mean it certainly was clear at that point that there were a lot of good things to be done there. So, anyway I think you may have caused controversy in more places than you realized. [Laughter]

Mashey: Okay, so that stirred up a lot of controversy. Take it forward over the next couple years. How did this turn into SPARC [Scalable Processor Architecture]? How did you get involved with that? Talk about the wars that went on then, I guess, in the mid-1980's.

Patterson: So what was happening at this time? We were building the chips and we were writing papers, using simulators, and talked about instructions and how many instructions it would take and how fast we thought it would run and stuff. Kind of impressively, the proof was in the pudding. We could build chips. So, RISC 1 worked basically, but very slowly.

Mashey: When did you fab [fabrication] support from that?

Patterson: It was through MOSIS [Metal Oxide Semiconductor Implementation Service].

Mashey: It was MOSIS.

Patterson: Yes, MOSIS, and so I don't actually remember whether it HP [Hewlett Packard] or not who actually fabbed it, but it was the MOSIS implementation server. So, the whole VLSI, Mead/Conway thing enabled all of this.. Otherwise, it got us interested. How could we actually design chips, courses were being done. Before it was something only electrical engineering people could do, but actually we're reaching out to computer science people. Mead/Conway thing and the MOSIS chip fabrication, maybe PARC [Palo Alto Research Center] even fabricated it in the early years. I'm not quite sure. So, we got RISC 1 and it kind of worked, but much slower than we claimed in the papers and so a bunch of people [said], "Sure, sure RISC was a great idea, look at the clock rate". So, fortunately the other group was doing RISC II and it worked great. I mean, it had like a 400, it wasn't megahertz, four megahertz, I think that was right, a four megahertz clock rate in some technology. We did a shrink and it went a third faster. This paper went to ISSCC [International Solid-State Circuits Conference] and it was kind of a remarkable thing. You know, ISSSC is the crown jewel of electrical engineering. It's like getting into Science or Nature.

Mashey: So, let me think. As I recall about that time, Motorola 6800's were maybe eight megahertz or something like that?

Patterson: Yes, that would be about right.

Mashey: So, basically it was pretty close, the clock rate.

Patterson: Yes, it was pretty close. Much worse technology, this was MOSIS. So, we got the spin in the papers at ISSCC and I still remember when Manolis Katevenis gave it and I was sitting here kind of

remarkably. Because even if we were at four or five megahertz and Intel was at eight, we had a much better instruction, we had a 32-bit instruction set with compilers and all that stuff. So, we outperformed them, and it was really amazing. A bunch of university students can build a microprocessor that is genuinely faster than what Intel is shipping, and it genuinely was. And we did it, we just took a completely different philosophy and could do that. And I remember when Manolis [Katevenis] presented the results at the conference, at ISSCC in a giant room, really, whatever, 1,000 people there. There was this buzz afterwards.

Mashey: So what year was this?

Patterson: I think it was 1983 but you'd have to check that. It's 1983. It's kind of when did we submit it, when was the conference held or something like that, but we did the work 1982. My guess is it's 1983. But there was a buzz in the room and as far as I knew, the buzz in the room was, “Ah, don't believe that, this isn't really true”. Or, ”You know, oh my God, they really can do it.”, but I think what that did was technically resolve the issue in most engineers’ minds. Now, there was a completely different argument which they referred to as the railroad argument. The tracks were a certain width even if you came up with a railroad, this was kind of an IBM argument, whatever it is, 40 inches. Suppose it was better at 45. Who cares? The whole railroad is set to 40 inches. You can't possibly make money with RISC. It's just a bad idea. Technically it's a good idea, but in the real world, you know, there's all the software. You can't do it. And so the argument shifted to technically was it superior to not, and mind you, I think that was a big milestone there. And later the guy's friends who worked at DEC wrote a paper comparing their own internal efforts to building the VAX and they actually a few years later gave, I think, the best quantitative argument about what this was all about and said the RISC had this kind of factor two or three advantage given the technologies.

Mashey: Was that the Clark Bondicher [???] paper? Okay.

Patterson: Yes. So, I think that was kind of, well, for the arguments we were having in the early 1980’s, that paper from the other side kind of laid it out to their credit, honestly, and it looked like there was really a difference for the assumptions that we made.

Mashey: So, then this was clearly sort of the earlier version of SPARC in some sense. So, maybe talk about that.

Patterson: Yes, so what happened was I was consulting, I had been rotating through consulting for companies and I was thinking about the next thing I wanted to do after the RISC stuff. And although this language, Prolog, was very popular at the time, I for some reason decided Smalltalk, this object oriented Smalltalk approach was a more interesting language. And I don't have any idea now how I came to that conclusion, but then I thought that, well, it was about productivity. It looks like a really good decision now. The place doing Smalltalk was Xerox PARC, so I kind of invited myself to consult with Xerox PARC, and I ended up in the Smalltalk group. And what I learned was, boy, you know, basically it was going to be really hard to influence industry from PARC. They were just so isolated from products that that was going to take a long time. So, what happened while I was consulting with PARC on this Smalltalk work, which we get later called Smalltalk on RISC, was Bill Joy, former Berkeley graduate student who is nothing if not well read, had read Manolis Katevenis' dissertation and found it very convincing. So, they're shipping,

making lots of money, 68000's. He comes to me and says, you know, and here's this little tiny company. I don't know, it had less than 100 employees and he came and invited me, "You know what we'd like to do, we would like to build a RISC chip at Sun [Microsystems, Inc.]" and asked me if I wanted to help, you know, lead the way. And so he invited me over and the last time I had seen him and Andy Bechtolsheim in kind of these arguments, I actually got them the early 68000. So, I visited him and Andy when they were in a garage somewhere and gave them the microprocessor. He plugged it in and it worked, and showed that the 68000 worked well. And so I visited next time and it was actually the first building at Sun. It was actually a two-story building that had, you know, I don't know, 50 or 80 people and it was like, wow, this is a real company. [Laughter] And so here is this chance to go get your ideas into a real company's product line, and so I was kind of the first one on the project and then I helped recruit Bob Gardner from PARC as a full-time guy. And then we helped recruit a bunch of other people on that first SPARC team, and of course, Sun was growing like crazy. It was the Google of its day and so, you know, they were shipping more products all the time and it took them a few years to build SPARC. But my advice was, look, instruction sets aren't that big a deal if you kind of start from scratch, from first principles this'll take you a long time. Like, what would HP do? Oh yeah, one guy there said let's start off with brand new principles and it took them a long time. You know, well, let's start with the RISC stuff and then my other piece of advice is, I think that's a good framework but on a lot of technical details if you're not sure what to do, do what the 68000 did. That's what you're shipping, and that I think was really good advice. Like, how should the condition cuts work? Well, how did they work on the 68000? Right, do that and that kind of framework there. The other thing that I always get blamed for is register windows, which, depending on the technological time of day it's a terrible idea or a really good one. It waxes and wanes over the years. Like Tensilica has register windows and that's from Marlan of MIPS and Stanford. So, I said a fairly simple thing to Steve Muchnick who was working on the problem. I said, "Look, you want to do the graph coloring algorithms to do the register allocation, like in the Stanford papers and the out/in papers." He said, "Oh no, I don't want to do that. You know, that's too hard. We're not going to do the graph coloring." I said, "Well, if you're not going to do the graph coloring, you better use register windows, right? If you're going to do the graph coloring, I don't think you need register windows, but that's what we did at Berkeley. We didn't have all that sophisticated stuff, so we did register windows, you know, but that's the decision point.

Mashey: So, to be fair then, what that sounds like is that not just the language influenced what were in these chips, but the particular level of compiler technology.

Patterson: Sure, and what happened you guys were starting up at MIPS and I think the different philosophy of these companies, the compiler group at Sun, their philosophy was there's a bunch of code out there written in C. We should translate it and do our best job, and I think the MIPS compiler team took a more, "look, this is not legal C and if it's not legal C, if we take all kinds of C we won't be able to do all the optimizations we would like to do, so we're just not going to do that". So, I think the Sun philosophy of, you know, being more inclusive and all programs out there, maybe because they had been shipping machines for awhile were a lot more inclusive and more worried about how well the compiler worked and stuff like that. Now, later they went ahead and did the graph coloring stuff.

Mashey: Anyway, right?

Patterson: Anyway, but at this critical point in time where we had to make some decisions they said, I didn't have huge emotional investment in register windows. It was just one of these technological tradeoffs that happens at the time.

Mashey: Well, as you know they have come and gone, and come and gone.

Patterson: Yes, I'm surprised that they keep coming back. I thought, well, everybody would agree that they don't like them but they come back.

Mashey: They come back, yes. So that sort of did, in some sense, the technology transfer of that.

Patterson: There's a couple of other stories in there. I realized I was approached early on to join MIPS in retrospect, that Skip Stritter was consulting at PARC, I think, and came in to talk to me about doing that. And I'd already had my little canned speech about I'm an academic, I like being in a university. I like to transfer the ideas but, I'm an academic at heart and so I gave him my canned speech, really seriously. And he was like, “Okay”. [Laughter]

Mashey: Sounds like Skip.

Patterson: And I didn't even pursue what he was talking about. Oh, it was going to be Hennessy and you. That might have been interesting but I gave him my automatic, no I'm an academic non-industrial speech. The other thing was interesting along the way is I could see my friend Hennessy, they were doing this MIPS stuff. We were doing a different one here and I kind of thought this would be bad. It'd be better for us to use the same instruction set, and so I went to the guys at Sun, and said, "It'd be better if we use the same instruction set. We should embrace MIPS and kill our own design." And it was really hard for me to say that, and Bill Joy said at the time, he says, "Look, we've talked to the MIPS guys. They have a terrible business model. They want to charge us for board space on the number of MIPS on the board, not on the cost model. We cannot sell products based on that." [Laughter] So, I was a little relieved that they said that, but, I did say, "This is not the right thing to have multiple. Oh man, what should we do? Oh, let's give up on our own design and embrace theirs." And it was a true at the time. I mean, unfortunately at that particular moment in time, it was a bad business model, which you later changed, I think? Right, it did change, but at the time they were making the decision if there'd been a more reasonable business model, I don't know what would have happened but, Sun was in the business for money. I don't know that they wouldn't have, changed it, but I remember that time. I’ll stop there.

Mashey: So, from there, so bring us forward a few more years. You seemed to go off of the microprocessor stuff into something completely different.

Patterson: Well, some of them looked like the RISC stuff was going to work. We did a series of RISC projects. One was on the Smalltalk. The third one was the multiprocessor project and, you know, and cache coherency and stuff like that. So, it look liked the processor stuff was really humming along. It was going to get really fast and so Randy Katz, who, a few years behind me. He graduated from Berkeley, went elsewhere and we hired him back. We all got Apple computers, and so he got an Apple

computer which had floppy disks, and he bought a spare disk. And he picks it up and he goes, "Wow, look at this disk in this little tiny," at that time, tiny format. It was probably 5 ¼ inch but it was a small format. "I wonder what we could do with these?" Right, and so Randy asking that question, I wonder what we could do with these things? It was kind of like, here's Mead/Conway. We can design our own chips. Here's a new technological opportunity, what's the implications of that? And a lot of the stuff we've done at Berkeley gets started with a question like that. So, Randy asked that question. So, we started talking and so Bullock, you know, we understand Humboldt's Law. The processor stuff's going to go really fast. How we going to have a storage system to keep up with these processors that are going to go shooting through the roof? Plus, we'd done multiprocessor stuff as well, besides the microprocessors. One of the microprocessor effects, you can put a bunch of them. So, it seems like here, what are we going to do with these small disks? It seems like we should work on [a way to] help us with the performance problem. So, we talked about it and we came up with a proposal, and we would just replace these single, giant IBM mainframe disks with a lot of these smaller disks. And we wrote a paper. I guess this is the story of my life here. We wrote a paper, sent it to some former Berkeley students, PhDs who were now at IBM, and the paper was just a tech report. It was more or less rejected. Yeah, great, you replace this one big disk with 40 of these small disks. Guess what? Your reliability's going to be 40 times worse, right? This is a terrible idea. So, we said, ”Hmm, okay, good point. We didn't think of that, I guess. I think we better come up with a scheme for reliability.” So, then we came up with first some kind of error correcting scheme, and then the parity scheme, and the RAID [Redundant Array of Inexpensive Disks] 5. And so then we started talking to people about these ideas here. Oh yeah, I know what that is. Yeah, Tandem does that. They were talking about mirroring. No, no, our idea is different than that. And then, oh we have the error correction. Oh, yeah, I know what that is. Thinking Machines has already done that. They have this data vault. No, that's not the idea. So, what we decided we had to do was to write a taxonomy talking about the evolution. To explain what we were doing and why it was different, we needed to write this taxonomy and, this was a paper that I ended up being the first author of and I just numbered the options, right. The first one was mirroring. We call that RAID 1. The second one was, using error correction like a memory, that's RAID 2. Then we'd have a parity but you can only do big transfers. That's RAID 3. And then here's this other idea that I think that could so some small accesses and big accesses, call that RAID 4. And here's what we thought was a new idea that rotated parity and helps with the writing and stuff like that. So, a part of that paper is the argument. This time it's called, "A Case for Redundant Arrays of Inexpensive Disks," another case for paper, and the first part of the paper is it's a technological argument, big disk versus small disk and all the possible advantages, power, potentially. And then how are we going to do the reliability? And so that paper, A Case for RISC, which we wrote the draft of that paper, a technical report, and what are we going to do with that? Well, Garth Gibson and I, who's the lead graduate student, he'd worked on the multiprocessor RISC project with us and then was looking for a dissertation topic. So, he was the first one on board when we were having these conversations. We decided to go to a tutorial that was done by Santa Clara, the guy who's the disk guy there was doing it. We talked to him and he said, "Well, I'm going to do this tutorial." So, we brought photocopies of this technical report with us that laid out the case for RISC. So, it must have been December--

Mashey: You mean the case for RAID?

Patterson: Case for RAID, right, December or January of I guess, 1987. December 1987, January of 1988. We talked to Stonebreaker at the time who we were working with in part of this grant project. What should we do with this? Well, you better publish this thing. Well, what's the next deadline? Well, the database conference, SIGMOD [Special Interest Group on Management of Data]. So, we submitted something to SIGMOD but we brought this technical report to this big disk meeting with the disk experts

and we handed these out. Well, it was like it was before the World Wide Web. This was like dumping a million copies in the stratosphere or something. Whatever. The people just faxed this thing, photocopied this report, and it just went everywhere. There was a guy who used to be at CDC [Control Data Corportation] and then became at Seagate, Dave Anderson who's still a well known guy. He told us he got five different copies of it. People sent him The Case for Reduced Instruction Set Computer. So, that particular one, whereas the RISC stuff was anger, and hostility, and my reputation was attacked, it was, it's like people talking about abortion today. I mean it was that level of anger at the time. We did the RAID paper which was IO, which people apparently don't care about as much. Kind of laid out, here's this technological trend, big disk, small disk. People say, "That sounds like a good idea." So, the paper went out there and people were pretty…

Mashey: Well, you were a little more established by this time, too. [Laughter]

Patterson: And people kind of embraced the idea. I think Randy and I, for many years, thought of this as a storage performance thing, and what clearly made it popular was the reliability piece. Because to our surprise, Byte Magazine wrote an article, picked up the paper and wrote an article on it. Well, why would Byte Magazine care? These PCs are so slow, IO's not a problem for them and they were arguing the reliability benefits. There was all kinds of benefits at RAID and one of them was much more reliable. And so surprisingly one of the first companies to do it was Compaq. They had a RAID 4 array that was out there, one of the early ones. Another one was Data General. They were early ones. We tried to get anybody to use these ideas and it's kind of one of the things that I've learned is the advantage we have at universities is that any company who uses our idea and starts making money at it, all the rest of the companies kind of go, "Oh, damn. They used that idea and now we're going to have to too." So, we as a university research lab, if anybody uses our idea that can get it started, where if you're a captured research lab like HP, you kind of have one customer. So, for us the ideas went out there, there's a bunch of technical issues that we worked on. We actually built both RAID I, which is what we call a software array, and RAID II was actually probably the first network attached storage. It was, we thought we were building the storage for diskless supercomputers and there was this gigabit network at the time whose name escapes me right now. And so we designed it to plug into this gigabit network, and this was supposed to be supercomputers over there and we'd have network attached storage device. And we build this 200 disk RAID system called RAID II with a hardware controller and crossbar, and all that stuff. But as Garth points out later, that was pretty surely the first network attached storage device.

Mashey: Well, of course, we were pretty pleased to have some of this old hardware.

Patterson: Yes, you have RAID II.

Mashey: That's right.

Patterson: You have RAID II here [at the Computer History Museum] and at the time I always thought that RAID I was kind of the first one, but in retrospect it is the first network. I don't know what it looks like on the display, or if it's being displayed.

Mashey: Oh, it is definitely being displayed. It has a place of honor.

Patterson: Yes. So, it's the first network attached storage: big deal today. The network attached storage industry is a…

Mashey: Very large industry.

Patterson: And the RAID stuff is a very large industry too. And the story there, we were broadcasting this to the winds, what happened at just this time, EMC [Corporation] in Boston, which was building cheap DRAM [Dynamic Random Access Memory] to plug into IBM mainframes. IBM had just screwed them by changing the interfaces so they couldn't sell DRAM anymore. So, they needed a new product and this paper lands on their desks, IBM mainframe disks, lots of small ones. So, they go into the disk array industry and just do great. IBM forced them out of the DRAM and got them into their storage industry, and EMC does just great.

Mashey: So, this was basically just a few years, as I recall, after the paper.

Patterson: Oh, yeah it was very fast. So, the technical report is in December of 1987. The SIGMOD paper is whatever, May or June of 1988. Boy, the next few years all kinds of stuff happens. Now, as we learned in retrospect IBM on the system, the strange system for offices, System 32..

Mashey: It's 36 perhaps.

Patterson: Yes, the people there came up with their own version of RAID 5, but this is this community that approximately never published anything. We never heard anything about it, but they actually, probably have the first RAID 5 patents. I think I have never filed a patent at Berkeley, but they are probably, as Randy looks chronologically, they probably came up with the RAID 5 idea before we did. We were in the same time frame but they were at least a few months ahead of us. But I always thought the RAID paper was about, it was like the case for RISC, right, this whole technological thing, right. Here's the trends. It seems radical but replace the big one by a lot of small ones, and if you add some extras you get reliability. And here's this new way of thinking about the way storage should be designed, especially with the technological trends of just getting smaller and smaller because of the new form factors it was about. And then we explained what we were doing. We laid out the RAID terminology and numbers like that.

Mashey: I, of course, followed this fairly closely. Particularly, I had a later colleague at SGI [Silicon Graphics, Inc.] who always used to say, CPU for show, IO for dough.

Patterson: That's pretty good.

Mashey: So, it fits this pretty well. So, as I recall, about that same time you and John [Hennessy] were starting to do books.

Patterson: Yes, that's right, exactly that time. So, he and I had gotten together over the years and we were frustrated by the books, the architecture textbooks at the time, which were kind of shopping catalogs. Like, here's a research project, here's a computer. It was often,descriptive like that and I guess when you got done you were familiar with all these different ways to do things. So, we were pretty frustrated by that. The RISC stuff was really in contrast to it. So, “We should write a bok, we should write a book.” I could see that I was about to become department chair at Berkeley starting in basically July 1st, 1990. Somebody suggested we should write a book but for me it was like another reason to talk to John. Look, this is it, John. I thought my life would be over as department chair. That was the last technical contribution I would ever make and we’ve got to do it. So, I was able to take a sabbatical. John took time off too and so we decided it was now or never, and what we thought is, I had actually had some experience of writing textbooks. I did a textbook on Smalltalk and then I got involved in computer literacy. So, I was an experienced textbook writer. John hadn't done one yet, but,we said, "Geez, we got to get this done." And so we treated it like a computer project and we figured we would need a place to hideout to talk, and so the DEC Western Research Labs were able to house us. So, we would show up three times a week and talk about this stuff. We had class notes and we thought, “Wow, we're almost there.” A common, foolish mistake of textbook writers. But, you know, we would just go through these ideas and how should we explain it, and what are we going to do. And, and a lot of the things that are in the book came about from those conversations. And I would drive up, for some reason I would have to drive, I'm not sure how that happened, but we would drive to Palo Alto and have these all day meetings. And in between we'd go work, and as John and I thought about it at the time, that was the hardest we'd worked since we'd finished our dissertations. Wow!, we put in a lot of time. Unfortunately, other people worked to cover for us and we got an alpha version of the draft done, and we treated it like a computer [project]: We had an alpha, beta, and full design. And then we found a publisher. Said look, we have an aggressive schedule. Publishers hear this all the time: authors saying that they have an aggressive schedule, but we went on an aggressive schedule. We have this plan of an alpha version, a beta version that we want to beta test to get the feedback from the marketplace, and then we'll finish the whole thing. Because I had this experience with publishers saying stuff and not delivering. I said, "We want a penalty clause. If you're going to work with us and if we deliver, you need to deliver a penalty clause." So, Morgan Kauffman, which at the time was a small publisher, mostly AI [Artificial Intelligence], had never done architecture, they signed up. If they didn't deliver, they would be willing to give money to one of our favorite charities. That's the way it would go, and they signed up. Bruce Spatz was working at the time, bought into we'll do a beta version. So, for them it meant they had to publish two books. So, it was a big deal for them. The dates are in the preface of the first edition, but it's pretty amazing is the elapsed time. It's about a year. Then I took off and did a sabbatical at Thinking Machines while it was beta tested. And then after the beta testing, we did a bunch of revisions. And it turned out, beta testing has some marketing advantages. You get all these faculty already kind of developing courses and stuff like that, and we got really good feedback on it. And then the book came out officially in 1990 and, it was kind of a phenomenon. You know, I had written what I thought, books that I called artistic successes, and this time rather than the first time I did a book, I said, "I know what the right thing to do is. Don't you damn publishers tell me what to do." I said, "Well, there's a bunch of stuff we care passionately about. There's a bunch of other stuff we don't care, right." So, we beta tested it and marketed it, and then we got feedback. And so the stuff we didn’t care, sure we didn't include that. The stuff we cared passionately about we weren't changing. So we presented it, I think I came up with the title, A Quantitative Approach, and the subtitle, Computer Architecture, and that was a purposeful title. It was a quantitative approach as opposed to, amazingly, emotional or intuitive approach, or emotional approaches, or philosophical approaches to computer design. And that was, we captured that in the design. And so performance was pretty early, how do you evaluate performance was pretty early in what we were doing, so that we could set the framework for the whole rest of the book.

Mashey: Well, clearly that book then, as I recall, instantly took over as the prime book for architecture.

Patterson: Yeah, at the time, I remember a guy said later, an HP designer, he said, "When that book came out, as an experienced computer designer, it had 90% of what I knew was in the pages of that book." It was so popular that Microsoft, they stocked it like stationary. I think I'll get some pens, a ruler, and Hennessy and Patterson, and it just, wasn't that many pages, and it was just a completely different perspective. In fact, I like quotes, so I came up, what's the quote for the first chapter. I picked the Monty Python quote, "And now for something completely different," and that was in the first chapter.

Mashey: So, let's follow that going forward. Now, you guys are now on your fourth edition as I recall.

Patterson: Yes, of that book.

Mashey: Where on earth do you get the time to keep doing this? It seems like about every four or five years you come out with another edition.

Patterson: It's probably a little longer than that. So, the question isn't so much Dave Patterson is a professor as John Hennessy is president of a university. So, John, to his everlasting credit has been willing to put the time in. First of all, I think John and I both have a pretty high bandwidth rate. I think both of us, it's something to do with our sleep patterns that I think both of us…

Mashey: You don't sleep. I knew this had to be some special feature.

Patterson: I'm the West Coast surfer, California guy growing up and John's strictly an East Coast guy, and there's a lot of similarities between us. It's not only the way we grew up. I mean, I think he had an even bigger family than mine since his was Catholic. There are four brothers and us, but we both went to public universities. We both went to graduate school. I'm pretty sure, I don't know if he's the first member of his family to graduate from college, but likely the first one to go to graduate school. We just kind of had a common, even though, it's a common world view, and we kind of, the reason the writing went so well was we saw things the same way. We had the same value system and we didn't want to disappoint each other. John, I think, well in the first few editions, we were still active researchers, and people cared about the book, and we wanted to keep it going. After John became president, the first thing he said, "Look, I'm not going to give up the book." I think for him it was his way to stay in technically. Now, to his credit they talked us into doing an undergraduate book as well as this graduate book, and a lot of people don't know that. And so, John stuck through it seven times. So, the fourth edition, but he said, "You know, this was great, but for the undergraduate book, I wouldn't mind if I didn't have to work on that. But let me know for the next edition of the graduate book, because I have more financial incentives than John has." So, he's finally,decided to drop out on one of the two books. But I think he's willing to keep going on the graduate book, which makes me happy.

Mashey: Well, I must say I personally always thought this was one of the fine contributions from you guys. Of course, I have copies of all of these things.

Patterson: That's great. You wrote us a nice review at Amazon.

Mashey: Yes, yes, well got to take care of you.

Patterson: I think, "Why did it work so well?" Well, part of it was this completely different perspective. I think the other part is John and I took this really seriously. Wow, we're going to shape the minds of thousands of, not so much the industrial people but thousands of young people. This has got to be some of the best writing we ever did, and that was our attitude going in, "Wow!, this has got to be good stuff." And we were a little insecure, so we had some help with some professional editors, and we worked really hard, and as John said, "Boy, it's a big difference when your name's on the book in terms of reviewing it. When I wrote a chapter, geez, my name's on this. I'm really going to give Dave a lot of feedback on that, and vice versa." And we get along, so we just felt like, the standard had to be very high, and it's possible that not every, you know, author has that same, feeling here. And then in retrospect, gee, you know, I sure picked a good co-author. [Laughter] It helps. So, it was nice to have a really successful, brilliant co-author both to share the work and make us work on it, and in retrospect I don't think I could have picked anybody else to work with or had it work out as well.

Mashey: So, let's go back to some of the further research. There was a period there where, as I recall, you were doing networks of workstations. And actually that brings another question. So who is your acronym provider? You were associated with a number of well known acronyms.

Patterson: Well, first of all I think, from what I can tell on average academics are acronym challenged. I have colleagues who name projects after stars that I can't pronounce and, it's seven or eight letters and they don't remember what it stands for and stuff. So, and I think in some parallel universe I'm in a marketing department, because I think I've got a knack for this and, it just was common sense. Well, it's great if you can pick an acronym that's actually a word because then it kind of comes up. It's great if it's not too many letters and stuff like that.

Mashey: There is a place for you in marketing, you're right.

Patterson: Yes, fortunately I got to do technical stuff but I think I have some sense of what's a really bad name and what name will work. And people do occasionally ask me for that, but yeah, I can still do it. I mean, recent projects, I think, are pretty nicely named. So, one of my students, very successful student, Peter Chen at the University of Michigan, but he learned from me, he said, about a research project is to start projects with the name R. So, he's done Rio, and Rio Vista. So, he thinks start it with an R, but hopefully that's not the only secret.

Mashey: So, anyway the next one was NOW [Network of Workstations] as I recall.

Patterson: Right, didn't start with an R. I think another thing, as I look back on my career, is I often start talking about the next thing I'm going to work on before that. And sometimes I don't know if I'm a great person to hire on sabbatical because, went to DEC, kind of saw how VAXes were built and thought, boy, there's got to be a better way to do this. Well, I spent the sabbatical at Thinking Machines in 1990 and

saw all the effort they did to build these things. They're building stuff out of microprocessors. Here's one way to do it and it's kind of like building, certainly like building a mainframe. It's the same, lots of copies of the chips but there's many mainframe aspects to a minicomputer. Just how much better is it then a bunch of workstations? Because I could see the rates of innovation and it had some similarities to, like, the microprocessors versus minicomputers, right. You get the box. They're manufactured in the box. You can buy a lot of boxes. That's not the problem. Sun has a very high innovation rate because of the higher volumes. And, I could see how long it would take a company, a small company the size of Thinking Machines to put this all together with the software and stuff like that. And they have this wonderfully integrated network, and it could scale, and in fact it's designed to scale. In fact, one of the issues for Thinking Machines was, if you were way smaller than the scale point, it was not so economical. So, to be efficient you had to be close to the target, and it's just coming back from that to Berkeley after the sabbatical, it was, boy it just seems like how bad would it be to be able to build this. The networks were getting better. We ought to, and so that conversation, David Culler who worked on Thinking Machines, younger guy, Tom Anderson in operating systems who had joined, and that kind of technological…I videotaped my graduate courses and so people watch them. And they told me a couple years later, "Yes, you were talking about that." It kind of got to the point in lecture where there's LANs, there's workstations, they're kind of a multiprocessor. So, that led to the Network of Workstations [NOW] project and John and I have had these friendly rivalries here. So, when we were doing networks to workstations, he was doing DASH, the distributed shared, the address machines, which was a cache coherent supercomputer time rather than these clusters of workstations. So, these were kind of contrasting philosophies. The RISC stuff, I tried to join forces with IBM and Stanford because you could think of us having a common goal. The DASH effort and the network workstations were really contrasting, right. Both of them were trying to build large computers in a way that could scale with completely different technologies.

Mashey: So, then I think a bit later we're back to an R with RAMP [Research Accelerator for Multiple Processors] .

Patterson: With RAMP, okay, yes. Yes, so what I genuinely believe is like the biggest challenge, and John and I both, the amazing thing of where we are in the state of technology today is the standard computing industry has bet its future that we can solve the multiprocessing problem, which we try really hard every decade or so and give up. Make minor progress, push the rock a little bit on the hill and that's it, let's take ten years off. We've bet the industry on solving this problem. So, a group of us at several universities have said one of the obstacles is getting hardware in people's hands that can be at the scale of the future that we'd like to be able to change. So, RAMP stands for Research Accelerator for Multiple Processors, and the basic idea is now that FPGA's [Field-Programmable Gate Array] have grown with Moore's Law, we can fit several processors inside of one FPGA Rather than one processor, a lot of FPGAs. The latter problem is really hard, splitting a design across a lot. Putting a lot of simple processors into one FPGA, not so hard with the current tools. And so our bet is that it runs fast enough that we can run real software on it. You know, it's 100 megahertz, 200 megahertz. Now, that's slow relative to gigahertz rates today, but that's plenty fast enough to run software. All of us remember computers that fast, so that it would be flexible enough. You could tape out everyday. You could change it all the time, and ramp up the multiprocessing effort to really increase the rate of innovation. And so there's a bunch of us including several universities and some companies like Microsoft are helping manufacture the equipment. So, we see this as kind of an open source infrastructure for hardware people at a time when we desperately need innovation because the industry's future is precariously balanced on this ability to solve this really hard problem.

Mashey: So, does this in some sense represent a return to that early 1980's period when universities could actually do pretty interesting hardware?

Patterson: Yes, absolutely we can do interesting hardware that will run interesting software. We have to kind of fake the clock rate. Right, but that's not that hard. Because we're doing research, well it's really 100 megahertz. Let's set up the system as if it was two gigahertz. That's not that hard a thing to solve and as long as we can keep track of this accurately, which is some extra work, this gives us huge advantages. We don't actually need to take five years or more to build a chip. We can come to a design. We can get other designs. We can modify designs. We can share the hardware. We could actually do this radical idea. When MIT claims to have some breakthrough, we could actually FTP [File Transfer Protocol] the code over, run it on our version and see if it's true, see if those bastards are lying or oh my God, they're right. And so we think it could really change the rate of innovation, and what's interesting about it, this is a time, maybe ten years ago people wouldn't care, but right now, Intel, and Microsoft, and IBM, and Sun are betting their futures that we're going to be able to make parallel programs run well. Correct programs that run efficiently on these many core architectures, which is a hell of a bet and so we really need the innovation right now and I think RAMP will probably play an important role in helping find the solution.

Mashey: I think that's a good point.

Mashey: So back to research, what are you working on now?

Patterson: Right now, I'm working on two things. One is called the RAD Lab. I think, as I look back on my career, I've kind of done small things and big things, so microprocessors like RISC and then big systems like RAID and NOW. So the big one, RAD, stands for "Reliable Adaptive Distributed" computing systems. And what we're focusing on is new internet services, like a YouTube or an eBay, and how can we create the technology to allow one person to create it and build a giant system serving a million people without building giant companies. And a big part of the bet is using machine learning to give the sophistication to run that. And what we're focusing is on the data center itself, beside that's the leverage point. One of the phrases I think Louis Berozo [???] came up with is, "The data center is the computer." So if the data center is the computer, then the question we're asking ourselves, well what's CAD for the data center, and what's the operating system for the data center? So we're trying to build CAD for the data center and an operating system for the data center and using machine learning to give us this leverage of when all these things are going on how to make sense of this. And the other one that we're just starting is called the PAR Lab, for Parallel Computing Center, and it's let's try and help industry with this gigantic challenge that's facing it. And this will be the, I've counted, and I think the PAR Lab will be my tenth project in my career. It's possible that this will be the most important one, if we can do it.

Mashey: If you can do it.

Patterson: Yes, if you can solve this, right. It's just a remarkable time. A $250 billion industry has bet that somebody can solve this kind of open problem in computer science.

Mashey: Which is filled with bones and broken motorcycles.

Patterson: Yes, right, and what I try and explain to people is more or less every parallel startup went out of business, failed and went out of business; 10 or 15 I could come up with. So it's not like we haven't tried. We've broken our pick on this problem many times in the past, every decade, right. And wow, look where we are today. And so it's almost like a Manhattan Project from a university. It's got academics involved, and there's a time limit. Suppose we can solve this problem in 20 years. Well, the industry may go away or be completely changed. Suppose we can solve it in 10 years versus five. Every year counts, so there's this sense of desperation going in. So I'm leading up a team at Berkeley in this area, and we're off to a great start.

Mashey: So let's go to something a little orthogonal. You've spent a bunch of time as a department chair. And you actually took on the job of being ACM [Association for Computing Machinery] president for awhile.

Patterson: Yes.

Mashey: So talk about those experiences. Those are not research exactly.

Patterson: No. I think I've done a couple of things like that. I think the chair; I thought that was the end of my life, but the skills that you need to get a group of people to ship a computer kind of carry over to getting groups of people to be successful in academics. So you have to learn to delegate. To do these jobs, you need to be able to inspire people, lead people, and stuff like that. So I think that same skill set carries over into academics; so I did that. I later helped with the Computing Research Association where I was chairman of the board, or chair, I guess it's called. I did that for four years. That was a bunch of volunteers from all over the country trying to worry about the care and feeding of computer science research. And then more recently, I was president of ACM for a couple of years. You have to be elected, like the CRE thing, and leadership. And so that leadership skills, I know how to get people to do things, run meetings efficiently rather than wasting everybody's time. And I think there's a chance that the fact that my good friend, John Hennessy, was president of Stanford may have contributed to my willingness to be president of something. [Laughing] There is a chance that that contributed to my saying yes.

Mashey: So let's finish up with sort of a set of questions, just to sort of summarize all of this. So here, I'm afraid, we don't have the time for you to go off into all these lovely long stories, but hopefully they'll summarize this whole thing. So I guess the first one is, so put this altogether, what do you think your most important life lessons were from all this?

Patterson: I'm not sure I want to be old enough to be asked questions like this. [Laughing]

Mashey: I'm sorry, Sir, but we're both at that point.

Patterson: I think optimism has served me well. It's not unbridled optimism, cautious optimism, but there have been a lot of times in my life where it's been a 50/50, or maybe 60/40 or 40/60 or something, and I said, "Let's go ahead. Let's try." And that's served my really well. And I think had I been more pessimistic, I wouldn't have called Berkeley. There had been a lot of things I wouldn't have done, so I

think the cautious optimism there. I think the other thing is kind of following your heart. People ask for advice on what they should work on. And man, if you don't care passionately about it, it's just hard to imagine you'd be successful.

Mashey: So you've had a lot of great achievements here. What would you call your proudest moment, which could either be technical or something else?

Patterson: Oh geez, I think proudest moment is probably family things. I'm pretty sure of that. Boy, that's a tough one. I remember, my two sons, getting married. I'll get in trouble. [Laughing] Getting married to my high school sweetheart, having a child was kind of shocking, having a second son, for somehow I had enough insecurities that when my second son was born, hey I'm a real man. [Laughing] Getting the job at Berkeley was a pretty proud moment. I won the distinguished teaching award at Berkeley. I was kind of insecure. Geez, maybe they made a mistake giving me tenure. Maybe it was Peterson, not Patterson. I said, "Oh, I won the distinguished teaching award." Yes, I think those are some of the things. The fact that the wrist chip worked. Boy, that was-- it's just all magic, right? You put these bits in. You send tapes. The piece of silicon is going to come back. What's going to happen? Will it just burn up or will something actually happen?

Mashey: So how about in all of these, were there any particular turning points or inflection points that happened that sort of led you down this path? Can you pick a couple of those?

Patterson: So in terms of personal life history, Jean Lubert [???], being nice enough to help me get a job while an undergraduate. My wife being willing to sacrifice personal financial security for this. Clearly, in the terms of my academic is going to DEC and stopping and thinking about what makes sense in the environment and what's a sensible project in coming back. Following the advice I just gave. Follow your heart. What do you want to work on? Well, other people want me to work on this. And it's like, well what do you want to go do? I want to do this. Well, do that. [Laughing]

Mashey: So if these things hadn't happened, in an alternate world, what would you have been doing? Yes.

Patterson: What would I do? People have asked me that before. I did a lot of sports, even though I did an individual sport, wrestling. Somehow I was on good teams, and I learned how to bond people together and form a team and to be successful as a team. I assume I would be doing something that would take advantage of that skill. I enjoy it, too. I notice all my books are co-authored. Well, probably almost all my papers are co-authored. I think I like doing things with other people. So I would think some kind of team lead, something like that.

Mashey: So you've had a lot of contributions, so pick a couple that you think are your biggest contributions to the field.

Patterson: Wow, biggest contributions to the field, geez, that's a tough question.

Mashey: Or impacts, perhaps, of those contributions.

Patterson: Impacts, I think both the RISC ideas and the book kind of changed the way people thought about designing computers. There was a different mindset after that. This wasn't just me. The RAID stuff, as I think about it, it's kind of like there's all these books you read about how to design reliable systems, dependable systems that often are only used in very niches. But as I think about it, the ones that really everybody uses is ECC [Error Correcting Code] in memory. And now RAID in storage. RAID in storage is pretty standard. It's not controversial. Of all the ideas you read about in dependability, there's a few that everybody… that use a lot of ones people talk about, so those probably had big contributions.

Mashey: And almost finally, how about role models, who did you particularly follow, if anybody? Or who, perhaps, also helped you along particularly?

Patterson: Yes, certainly my father, I'm the oldest one in my family. And my father is a person who the way he lived his life: family-oriented, working hard for the family and stuff like that. He was a major influence. He always commuted to work an hour or so each way, and that inspired me to live within bicycling distance. [Laughing] But the family-oriented stuff, certainly my father. I was very influenced at Berkeley by Dick Karp and Manuel Blum. These are two theoreticians, very well known, two award-winning theoreticians. And they set a standard of teaching excellence, of being nice guys. They're absolutely not prima donnas. I don't know anybody more famous than those two guys, and nice people. And they set a model of behavior and approach, and it was just really great to have them around. I liked that. I like that. I'd like to believe I would act that way anyways, but it was really nice that you could be very successful and do the things that I wanted to do, work with other people, be a nice person, really be good at teaching, spend a lot of time and energy in teaching and not be a prima donna. So I think those were some of the people who influenced me.

Mashey: So given that emphasis on teaching, can you perhaps give any advice for the sort of current and next generations of folks in this field?

Patterson: You mean the academics or the people getting into the field?

Mashey: Both, yes.

Patterson: So there's all this enthusiasm about biology in this coming century, and I'm one of them. I think everybody thinks that's an exciting area. But I'm one of the people who thinks we have barely scratched the surface in information technology. We are going to be talking about the prehistoric era, like when you and I talk about punch cards. They don't even understand that changing a comma would take eight hours. That's not possible. Yes, so I think it won't be long before we're looking back at that. I mean, suppose we solve the parallelism problem, and we make programming productive. There are these huge things that could happen. And if our semiconnductor colleagues keep making more powerful chips and we can figure out better ways to use them, then it'll really raise the level of extraction. This is the phenomenal impact. You think about a billion cell phones a year. We have six billion. It's a billion new cell phones a year, 250 million PCs. We're really having the opportunity to impact the whole world. You think about how many people, including me, use Google all the time, just search every single day.

That's part of your head. So we're just starting to see the impact of this technology. The friends who do the search stuff say we've already figured out the interface to AI. We're going to type two or three words to this Oracle, kind of slipping things into the mouth of the Oracle, and expect to get wisdom back. There's hundreds of millions of people who are already thinking that way, and if we can make that technology grow. So it just seems like an extraordinarily interesting time right now, with the parallel processing stuff, data centers, computer people are transforming our societies around access to information and this unbounded number of opportunities in this field. And very likely, anything else you want to do that you're interested in, help economics, politics, anything you want to do, having the ability to create this technology and use it to its best advantage is multiplying advantage that you could have a much bigger impact and many other ways you can do that. So I think this is just a great time, a fantastic time, to be getting into this field.

Mashey: Well, that's great. So you think computer science is not done.

Patterson: Yes. Computer science is not done. [Laughing] It's like the guy closing the patent office in 1870. Man, there's so much, like when we talk about this parallelism stuff. The best software has not been written yet. Not a single line has been written yet. There's things I would like to have to carry around with me in these things every day [holds up a Personal Digital Assistiant] that no one's written yet. And it will make my life a lot better once they do.

Mashey: Super. Well, thanks so much, Dave.

Patterson: Thanks, John.

END OF INTERVIEW

